Reactis®
Model Inspector

User's Guide
Version 2020.2

Reactive Systems, inc

How to Contact Reactive Systems:

Web	reactive-systems.com	
Email	`help@reactive-systems.com`	(Technical support)
	`sales@reactive-systems.com`	(Sales)
	`info@reactive-systems.com`	(General inquiries)
Phone	(+1) 919-324-3507	
Fax	(+1) 919-324-3508	
Mail	Reactive Systems, Inc. 341 Kilmayne Dr. Suite 101 Cary, NC 27511 USA	

Reactis Model Inspector User's Guide

Contents

Chapter 1

Introduction

The Reactis® Model Inspector offers a low-cost lightweight tool for viewing Simulink models and the testing artifacts created for them with Reactis. The tool is aimed at members of an engineering team who need to review Simulink models and testing artifacts but not generate tests or debug the model. Some capabilities include:

- Hierarchical browsing of models, including Simulink®, Stateflow®, C Code (included as S-Functions or Stateflow custom C code), and Embedded MATLAB® code.
- Tracing signals through the model.
- Text search of the model (including C code and Embedded MATLAB code).
- Browsing Reactis-constructed test suites (.rst files).
- Browsing (but not editing) Reactis Info Files (.rsi files).

Support and Feedback

RSI welcomes user feedback and questions. To ask questions, make suggestions, or report suspected bugs, you may call RSI's help line at (+1) 919-324-3507 or send e-mail to:

`help@reactive-systems.com`

When sending e-mail, you are encouraged to include the "System Info" information for your Reactis installation; this can be obtained by selecting the Help → About menu item from the top-level Reactis Model Inspector window, then clicking Copy To Clipboard and pasting the information into the e-mail message.

Chapter 2

Installing Reactis Model Inspector

Reactis Model Inspector is available on Windows®[1] 10. To run Reactis Model Inspector, a system should satisfy the following minimum requirements:

- At least 512 MB RAM (more is required for large models)
- At least 40 MB free disk space
- An installed Ethernet card
- 64-bit Reactis Model Inspector requires 64-bit Windows

There are different methods for installing Reactis Model Inspector, depending on how you will access the license(s) you have purchased:

1. via a Reactis License File residing on your computer, or
2. via a Reactis License Manager running on a remote server.

In the latter case, you (or someone in your organization) will also need to install the Reactis License Manager. The Reactis Model Inspector installers may be downloaded from the Reactis User Pages:

`https://reactive-systems.com/login.msp`

2.1 Installing with Local License File

To install Reactis Model Inspector, perform the following steps:

1. Execute the self-installing executable

   `reactismi-setup-V2020.2.exe`

   and follow the instructions. The 64-bit Reactis Model Inspector installer is named

   `reactismi-setup-win64-V2020.2.exe`

[1] Windows® is a registered trademark of Microsoft Corporation.

but otherwise installing 64-bit Reactis Model Inspector proceeds exactly as the 32-bit install. If you obtained your release on a DVD, inserting the DVD into your DVD drive should initiate the execution of this executable automatically. If it does not, manually run the program `reactismi-setup-V2020.2.exe` that is included on the DVD. Alternatively, you may download the latest version of the installer from the Reactive Systems web site listed below. Note, that an updated version of the installer will have a name of the form `reactismi-setup-V2020.2.2.n.exe` where *n* is a patch release number.

2. To obtain a Reactis License File, select

   Start → All Programs → Reactis Model Inspector V2020.2 → License

   from the Windows Start menu. This will open a text editor showing the (incomplete) license file `rsilicense.dat`. Copy the information you find there into an e-mail message and send this message to Reactive Systems at `help@reactive-systems.com`. You will receive a response containing the completed license as an attachment. When you receive this e-mail, detach the license file and save it to a file named `rsilicense.dat` in the folder where Reactis Model Inspector is installed. If you installed in the default location the license should be saved to:

   ```
   C:\Program Files\Reactis Model Inspector V2020.2\rsilicense.dat
   ```

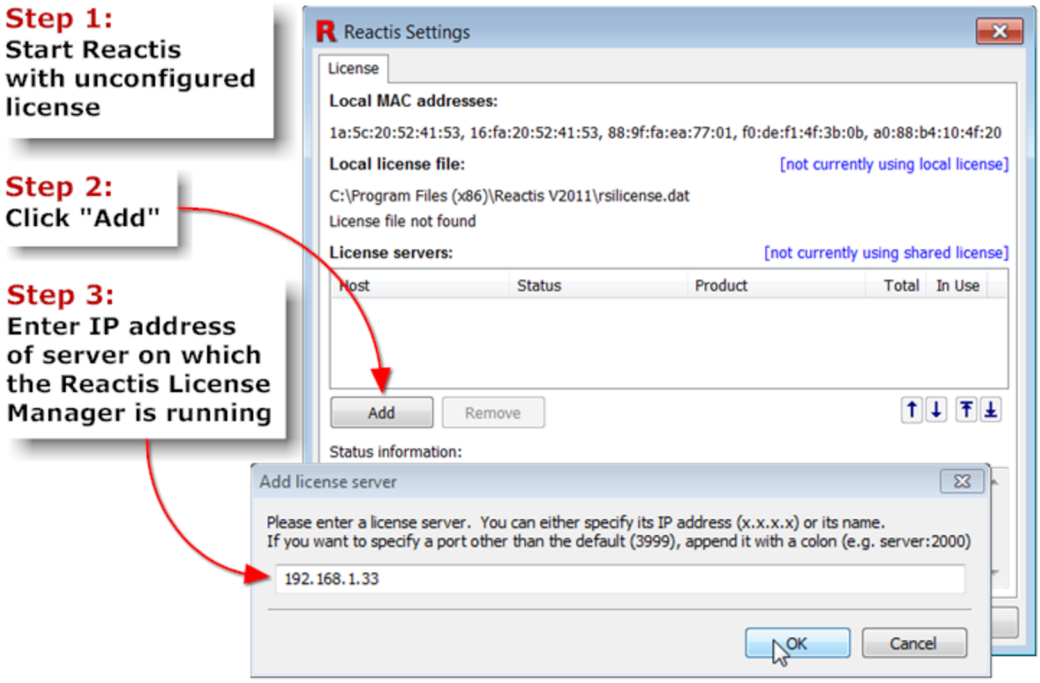

Figure 2.1: Configuring Reactis Model Inspector to access a Reactis License Manager running on a remote server.

2.2 Installing with Remote License Manager

If your organization already has a server running the Reactis License Manager, then you will need the name or IP address of the license server. The steps required to install Reactis Model Inspector on your computer (the client) are as follows:

1. On the client machine, run the installer (see step 1 of Section 2.1).

2. Invoke Reactis Model Inspector by selecting

   Start → All Programs → Reactis Model Inspector V2020.2 → Reactis Model Inspector

   The dialog shown in Figure 2.1 will appear. Click the Add button and enter the IP address or name of a server running the Reactis License Manager.

Step 2 can be repeated multiple times if there is more than one server.

2.3 Installing the Reactis License Manager

Please see the Reactis License Manager User's Guide:
`https://reactive-systems.com/doc/reactis-license-manager/reactis-license-manager-toc.html`

2.4 Performing a Silent Install

The Reactis Model Inspector installer supports a silent install (installing Reactis Model Inspector from the DOS command line in an automated fashion). The following command line switches control a silent install.

- `/SILENT`

  Instructs the installer to be silent. When the installer is silent the wizard and the background window are not displayed but the installation progress window and error messages are displayed. The default settings (which can be overridden using command line arguments) are used for the install.

- `/VERYSILENT` With this switch the installer will display error messages, but not the wizard, background window, or progress window.

- `/NOCANCEL` Prevents the user from canceling during the installation process by disabling the Cancel button and ignoring clicks on the close button. This switch is useful in conjunction with /SILENT.

- `/DIR="x:\dirname"` Overrides the default directory name displayed on the Select Destination Location wizard page. A fully qualified path name must be specified.

- `/GROUP="folder name"` Overrides the default folder name displayed on the Select Start Menu Folder wizard page.

- `/COMPONENTS="comma separated list of component names"` Overrides the default components settings. By default all components are installed. Possible components are:

main	:	Reactis Model Inspector GUI
lm	:	License Manager
lmo	:	License Monitor
help	:	Help files
examples	:	Example files

- `/COPYSETTINGS` Instructs the installer to copy the personal settings and license information to the new installation if running in SILENT or VERYSILENT mode and a previous existing Reactis installation is found. This is the default.

- `/NOCOPYSETTINGS` Do NOT copy personal settings and license information from a previous existing installation.

- `/SAVEINF="x:\filename"` Saves information entered during the install in a file. That file can later be used to specify default parameters via LOADINF. The following information is saved:

  - Install directory name
  - Program group name
  - Components
  - Should settings and license information be copied from existing versions of Reactis Model Inspector?

- `/LOADINF="x:\filename"` Load default information from a file created with the SAVEINF option. This is useful in combination with a silent install, i.e.

  ```
  C:\> reactismi-setup-V2010.exe /SILENT /LOADINF="mysetup.inf"
  ```

  will install Reactis Model Inspector without any user prompts using the values saved in mysetup.inf Options given on the command line will override parameters stored in the information file.

- `/HELP or /? or /H` Display help information for the silent install command line switches.

Chapter 3

Reactis Model Inspector Top-Level Window

This chapter concentrates on the functionality available in the Reactis Model Inspector top-level window.

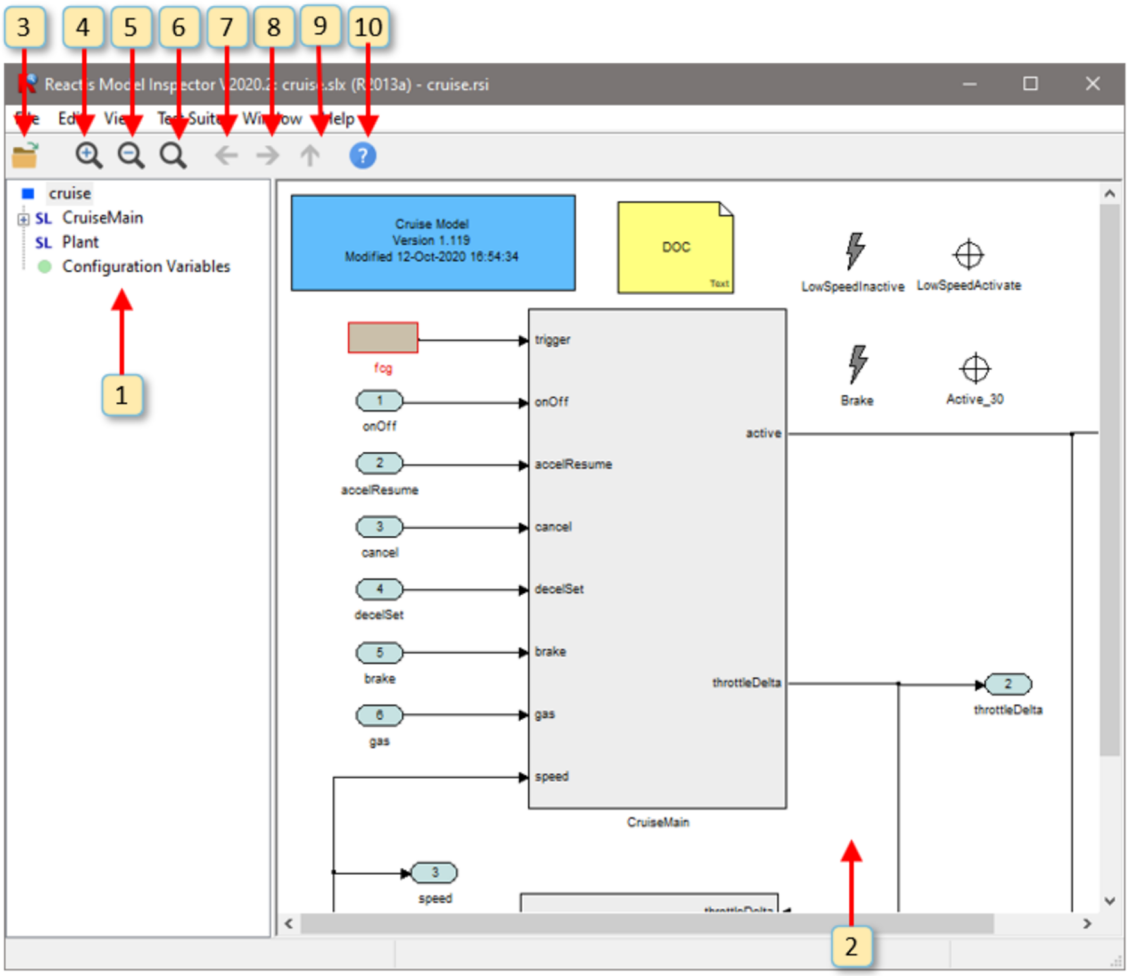

Figure 3.1: The Reactis Model Inspector top-level window.

3.1 Labeled Window Items

An annotated screen shot of the Reactis Model Inspector top-level window may be found in Figure 3.1. This section describes the functionality of the numbered items in this figure, while the section following discusses the workings of the pull-down menus.

The numbers below refer to the labels in Figure 3.1.

1. The model hierarchy panel shows the subsystems in the model and how they are related. Clicking the + to the left an item displays the subsystems of the item. Clicking on an item causes the diagram for the item to be displayed in the main panel (window item 2). Pressing the "F2" key causes the parent of the currently displayed system to be displayed. Hovering in the hierarchy panel over a child of the currently displayed system causes the child to be highlighted in the main panel.

   Right-clicking on an item in the hierarchy panel causes a pop-up menu to be displayed with an entry Copy System Path. Selecting the entry causes the path of the selected subsystem to be copied to the clipboard. The path can then be easily pasted into another document such as a spreadsheet, text file, or email. The copied system path can also help you easily navigate to the appropriate subsystem in the Simulink editor if you decide to modify a portion of the model currently displayed in Reactis Model Inspector. To do so:

   (a) In the hierarchy panel, right-click on the subsystem of interest and select Copy System Path

   (b) Start MATLAB and open your model in Simulink

   (c) At the MATLAB prompt type `open_system('`

   (d) Paste in the system path from the clipboard

   (e) Type `');` and hit return

   (f) The subsystem of interest will be displayed in the Simulink editor

2. The main panel displays the currently selected Simulink subsystem, Stateflow diagram, C code, or Embedded MATLAB code. You may interact with the diagram in a number of different ways using the mouse including hovering over model items, double-clicking on items, or right-clicking in various parts of the panel. The following mouse operations are available:

   Hovering...

   - over a From or Goto block will cause it and its associated block(s) to be highlighted in yellow.
   - over a Data Store Read, Data Store Write or Data Store block will cause it and its associated block(s) to be highlighted in yellow.
   - over a Validator objective will cause its wiring information to be drawn in blue arrows from the data items monitored (or controlled in the case of a virtual source) to the objective.
   - over a top-level input port that is controlled by a virtual source will show a blue arrow indicating the virtual source block that is controlling this input port.
   - over a Stateflow transition segment will cause the segment and its label to be highlighted in yellow.

Clicking...

- on a signal line in a Simulink system highlights the signal in yellow. This makes it easy to trace a signal back to the point where it was generated and forwards to the point where it is consumed (used by a block to compute a new value). The highlighting flows through (backward and forward) the following blocks that do not modify a signal value: Inport, Outport, Subsystem, From, Goto, Data Store Write, Data Store Read.
- in empty space removes signal highlighting.

Double Clicking...

- on a Simulink block will display the block's parameters.
- on a Simulink subsystem will cause the subsystem diagram to be displayed in the main panel.
- on a Stateflow state will cause the state's diagram to be displayed in the main panel.
- on a Simulink S-Function block with an associated Reactis Makefile (`.rsm` file) will cause the `.rsm` file to be displayed in the main panel.
- on a From, Goto or Goto Tag Visibility block will open a dialog listing all matching From and Goto blocks in your model (see Section 3.3).
- on a Data Store Read/Write/Memory block will open a dialog listing all matching blocks in your model (see Section 3.3).
- on a top-level input port will bring up the port type editor (in read-only mode) to inspect the constraint associated with the inport.
- on a configuration variable in the Configuration Variable Panel will bring up a type editor dialog (in read-only mode) to inspect that variable's type constraint.
- on a Validator expression objective will open the parameter dialog for that objective (in read-only mode).

Right Clicking...

Causes different pop-up menus to be displayed. The contents of the menu vary based on where the click occurs. A summary of the menu items follows.

Right-Click Location	Menu Entries
Validator objective (i.e. user-defined target, assertion, or virtual source)	**View Properties...** View properties of the objective.
Simulink block	**View Block Parameters...** Display Simulink block parameters.
Top-level input port or configuration variable in Configuration Variable Panel	**View Type...** Inspect type of top-level input port or configuration variable.
Top-level output port or test point Variable Panel	**View Tolerance** Inspect the tolerance used when comparing the actual value of the top-level output port or test point to its expected value.

When a diagram is too big to display completely in the main panel, scroll bars appear

for repositioning the diagram. Alternatively the diagram may be repositioned by left-clicking and dragging in the panel or by using the cursor keys. If your mouse includes a scroll wheel, then you may scroll vertically by clicking in the main panel and then using the scroll wheel.

Left-clicking and dragging in the panel while holding down the control key defines a "print region" that can be used for printing parts of a model. The print region is represented as a shaded blue box; it may be removed using the File → Remove Print Region menu entry or pressing the escape key.

Finally, pressing the F2 key within this panel causes the parent of the currently displayed subsystem to be displayed, pressing F3 zooms in, and pressing F4 zooms out.

3. Load a new model into Reactis Model Inspector.
4. Zoom in.
5. Zoom out.
6. Fit to page.
7. Go back in the history of displayed subsystems.
8. Go forward in the history of displayed subsystems.
9. Display the parent of the currently displayed subsystem.
10. Open Reactis Model Inspector help window.

3.2 Menus

This section describes the top-level menu items available in Reactis Model Inspector. Some menu entries also have keyboard shortcuts that enable the relevant operations to be invoked from the keyboard. These shortcuts are displayed to the right of the relevant entries in the menus.

File menu. The file menu contains the following entries.

Open Model... Load a new model into Reactis Model Inspector.

Close Model. Close the currently displayed model.

Reload Model. Reload the currently displayed model.

Select Info File... Specify a Reactis Info File (`.rsi` file) to be used with the current model. See Chapter 4 for a discussion of editing `.rsi` files with the Reactis Info File Editor. `.rsi` files store information that Reactis associates with a model including inport constraints, configuration variable settings, Validator objectives, outport tolerances, and other settings.

Extract Info File... Extract an `.rsi` file from an `.rtp` file . Reactis Tester may be configured to store launch parameters and the `.rsi` file used for a given run in a *Reactis Tester Parameter file (*`.rtp` *file)*. Selecting this menu item retrieves the `.rsi` file from the `.rtp` file.

Print... Open a print dialog for model printing. Section 3.5 explains this feature in more detail.

Remove Print Region. Clear the selected printing region in the main panel. You may select a region of a model for printing by left-clicking and dragging in the panel while holding down the control key. The resulting selection is highlighted within a blue box. Selecting this menu item removes the blue box.

Global Settings... Opens dialog to adjust Reactis global settings. Section 3.6 describes the use of this dialog.

Exit Reactis Model Inspector. Exit Reactis Model Inspector.

Edit menu. This menu includes entries used to view `.rsi` files (open the Reactis Info File Editor in read-only mode) and an entry to launch a model search function. `.rsi` files contain constraints on the values assumed by top-level inports, details related to Validator objectives, and other model information maintained by Reactis.

Find... Perform a text search of your model for strings matching a pattern you specify. Two types of patterns are currently supported. If the search pattern includes a colon (:), then the text before the colon corresponds to a Simulink block parameter name and the text after the colon corresponds to a value for that parameter. For example the pattern `BlockType:Inport` will initiate a search for all input ports in the model. If the search string contains no colon, then the search will examine Simulink block names, Stateflow state names and actions, Validator objective names, configuration variable names, and C code.

The scope of the search (the parts of the model examined for a match) is determined by the subsystem displayed when the search is launched. The scope consists of the currently displayed subsystem and all of its descendants (child subsystems, their children and so forth). Note that the search scope does not change if the search pattern is modified. The scope changes only when the search dialog is dismissed and a new search is launched. To search the entire model select the top-level and launch a search.

The following twelve entries invoke the Info File Editor in read-only mode to the tab specified by the menu entry.

Inport Types... Constrain the values generated for top-level inports during test generation.

Configuration Variables... Specify workspace data items that may change in between tests but not during a test.

Test Points... Manipulate test points for observing model behavior. Test points are internal data items that Reactis treats as *virtual outputs*; specifically, the tool records values for test points in test suites and, when executing a test suite, Reactis Simulator flags any differences between the values computed by a model for a test point and those stored in a test suite.

Outport Tolerances... Specify a tolerance for each outport of a model. When executing a test suite on the model in Reactis Simulator, an outport tolerance specifies the maximum acceptable difference between the value computed by the model for the outport and the value stored in a test suite for the outport.

General... Specify model-specific settings related to how a model executes (e.g. conditional input branch execution and short circuiting).

Error Checking... Specify the set of error checks Reactis will employ (e.g. flagging overflows or NaN values).

Coverage Metrics... Specify the set of coverage metrics to be measured when working with a model in Reactis. If a metric is disabled:

- the metric will not be targeted by Tester when generating tests, and
- Simulator will not include the targets of the metric in the Coverage Summary dialog, the Coverage Report Browser, and the highlighting in the main panel.

Excluded Coverage Targets... Displays a list of all targets which have been excluded from coverage and allows you to monitor the state or change the coverage tracking setting for each excluded target.

Validator Objectives... Displays a centralized list of all Validator objectives in your model and allows you to monitor, edit, and remove them.

C Code... Displays a list of all locations in your model where C code is used.

Callbacks... Specify fragments of MATLAB code to execute before and/or after a model is loaded. Note that these operations are distinct from the similar Simulink callbacks.

Search Path... Specify model-specific search path.

Dependencies... Specify files on which a model depends.

View menu. The view menu contains the following entries:

Back. Go back in the history of displayed subsystems.

Forward. Go forward in the history of displayed subsystems.

Go to Parent. Cause the parent of the currently displayed subsystem to be displayed in the main panel.

Zoom In. Zoom in the display of the model in the main panel.

Zoom Out. Zoom out the display of the model in the main panel.

Zoom to Fit. Fit to page; same as labeled window item 6.

Expand Tree. Causes the entire tree in the model hierarchy panel to be expanded.

Collapse Tree. Causes the entire tree in the model hierarchy panel to be collapsed.

Select Label Font... Select font for labels in Simulink / Stateflow diagrams.

Increase Label Font Size... Increase size of font for labels in Simulink / Stateflow diagrams.

Decrease Label Font Size... Decrease size of font for labels in Simulink / Stateflow diagrams.

Select C Source Font... Select font for displaying C source code in the main panel when using Reactis for C Plugin.

Select Line Styles... Select styles and colors for drawing various Simulink / Stateflow diagram items.

Test Suite menu. The following menu entry is available:

Browse... Launches the Test-Suite Browser by first opening a file-selection dialog to allow you to indicate which test suite is to be browsed. See Chapter 5 for details.

Window menu. Enables user to switch between different models currently loaded in Reactis Model Inspector.

Help menu. The Help menu contains the following entries.

Contents. Go to the table of contents in the documentation.

Index. Go to the index in the documentation.

Frequently Asked Questions. Go to the Frequently Asked Questions section of the documentation.

Release Notes. Display the release notes for the current Reactis Model Inspector version.

Top Level Window. Go to the section of the documentation that describes the Reactis Model Inspector top-level window.

Check for Updates... Query the Reactive Systems website to check if a newer version of Reactis Model Inspector is available.

About. Open a dialog displaying the Reactis Model Inspector version and other configuration information. The dialog includes a Copy To Clipboard button to transfer the information to the Windows Clipboard. When requesting assistance, sending this information to Reactive Systems via email often facilitates the efficient delivery of support.

3.3 List of Matching From/Goto or Data Store Blocks

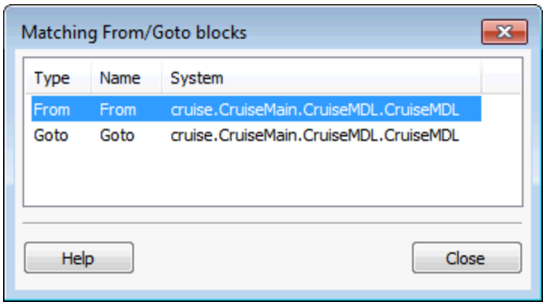

Figure 3.2: The list of matching From/Goto or Data Store blocks.

This dialog comes up after double-clicking on a From, Goto, Goto Tag Visibility or Data Store Read/Write/Memory block in Reactis Model Inspector. It lists all other such blocks in your model that match the double-clicked block.

In this dialog you can:

- Double-click on a row to have Reactis highlight the block in the main panel. If the block is located in a different subsystem than the one currently displayed, Reactis will switch to that subsystem.

- Click on a column header to re-sort the table by the values of that column.
- Click on the Close button to close the dialog.

3.4 Configuration Variable Panel

If the model you are inspecting has configuration variables, Reactis Model Inspector will include an entry Configuration Variables in the top-level of the hierarchy panel. Clicking on this entry displays a block in the main panel for each currently defined configuration variable. This panel allows you to view configuration-variable information. Double-clicking on a configuration variable opens the type editor dialog in read-only mode to inspect a constraint for the configuration variable.

3.5 Printing Models

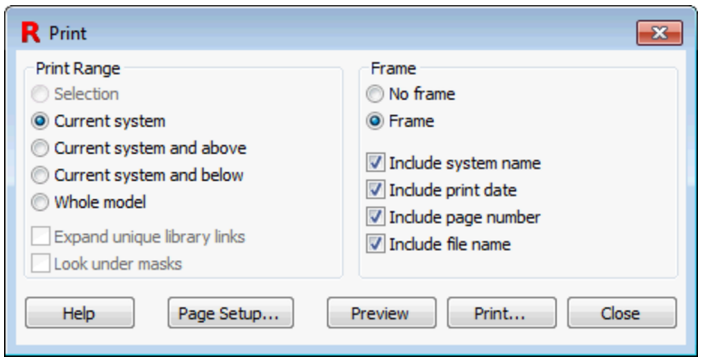

Figure 3.3: The Print dialog.

Reactis Model Inspector includes a flexible facility for printing models. Upon selecting menu item File → Print... the print dialog shown in Figure 3.3 appears. The radio buttons and check-boxes in the Print Range section of the dialog specify which portions of the model should be printed as follows.

Selection. Prints only the portion of the model located within the current print region in the main panel. If no print region is defined, this entry is disabled. A print region is selected by left-clicking and dragging in the main panel while holding down the control key. The print region is cleared by selecting menu item File → Remove Print Region.

Current system. Prints only the system currently displayed in the main panel.

Current system and above. Prints the system currently displayed in the main panel and all systems between the current system and the top-level system in the current model. Each such system is printed on a separate sheet of paper.

Current system and below. Prints the system currently displayed together with all its subsystems, sub-subsystems, etc. Each system is printed on a separate sheet of paper.

Whole model. Prints the whole model.

Expand unique library links. When checked, library blocks referenced by the model will be printed. Note that each library block is printed only once, even though some blocks might be referenced multiple times by a model.

Look under masks. When checked, masked subsystems will be printed.

The following radio buttons and check-boxes in the Frame section specify whether a frame should be printed on each page, and if so what content should be included in the frame.

No frame. When checked, no frame is printed.

Frame. When checked, a frame is printed.

Include system name. When checked, the name of the system is printed in the upper left part of the page frame.

Include print date. When checked, the date the model was printed is included in the bottom left corner of the frame on each page.

Include page number When checked, the page number is printed in the bottom right corner of the frame on each page.

Include file name. When checked, the name of the `.mdl` file containing the model and the folder containing the file is printed in the bottom left corner of each page.

The remaining buttons in the print dialog work as follows.

Help. Display print dialog help.

Page Setup... Invokes a dialog that allows the user to specify paper size and margins, and whether printing should be portrait or landscape.

Preview. Open a viewer to display what will be printed.

Print... Begin printing.

Close. Close the dialog (and cancel printing).

3.6 Reactis Global Settings

Selecting File → Global Settings... invokes the Reactis Model Inspector Global Settings dialog, which allows you to adjust some aspects of the way Reactis Model Inspector operates. The global settings are partitioned into four tabbed panes each described in detail below: General, Path, User Info, and License.

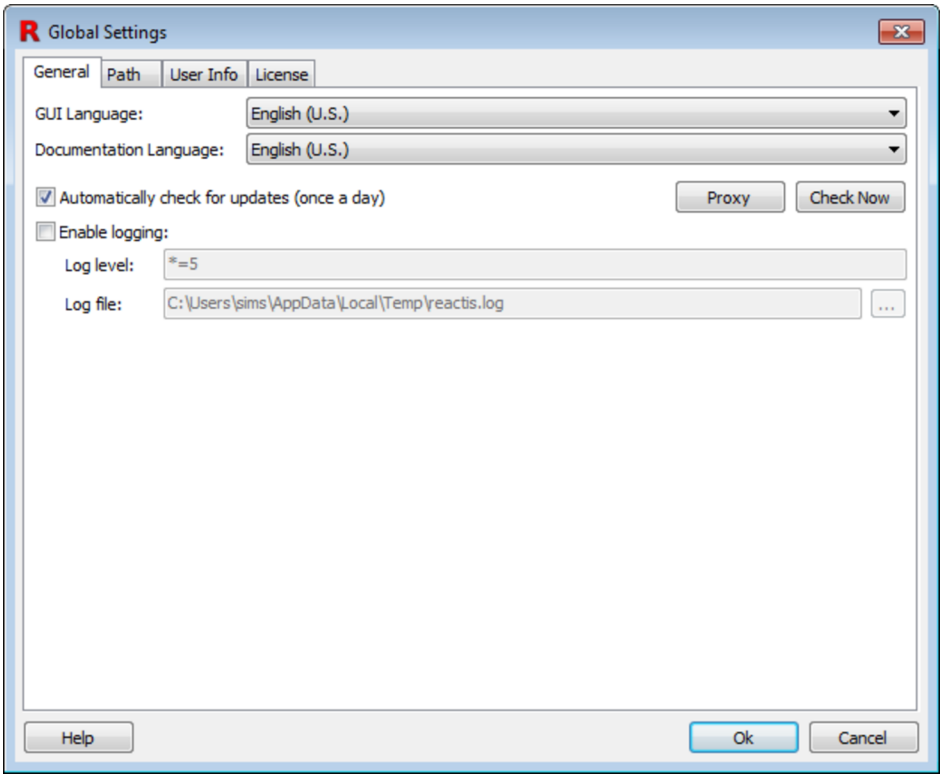

Figure 3.4: The Global Settings dialog with General tab selected.

3.6.1 General Settings

The General global settings tab shown in Figure 3.4 includes the following items.

GUI Language. Language used in the Reactis Model Inspector GUI.

Documentation Language. Language used for the in-tool Reactis Model Inspector help.

Automatically check for updates (once a day). Instructs Reactis Model Inspector to check once per day whether updates to Reactis Model Inspector are available for download. If updates are found you will be asked if you would like to download and install the patch. *Note: this feature can be disabled at install time; in which case this checkbox will not appear in the dialog.*

Enable logging. Enable logging, specify a log level, and indicate the file to which the log should be written. Note that logging degrades performance and can create very large log files; therefore, it is typically only used to diagnose problems. The log level string will be provided by the Reactive Systems support team if you are asked to create a log file.

3.6.2 Path Global Settings

The Path tab of the Global Settings dialog, shown in Figure 3.5, enables you to specify the list of folders in which Reactis Model Inspector will search for files such as Simulink model

libraries (.mdl). The order in which folders are listed in the dialog specifies the search order (from top to bottom).

Note that Reactis also gives users the capability to define model-specific search paths which consist of a list of folders to be searched when loading a given model. The model-specific path can be viewed in Reactis Model Inspector by selecting Edit → Search Path.... When searching for files, the complete search path is constructed by prepending the model-specific path to the global path.

The buttons labeled in the figure work as follows.

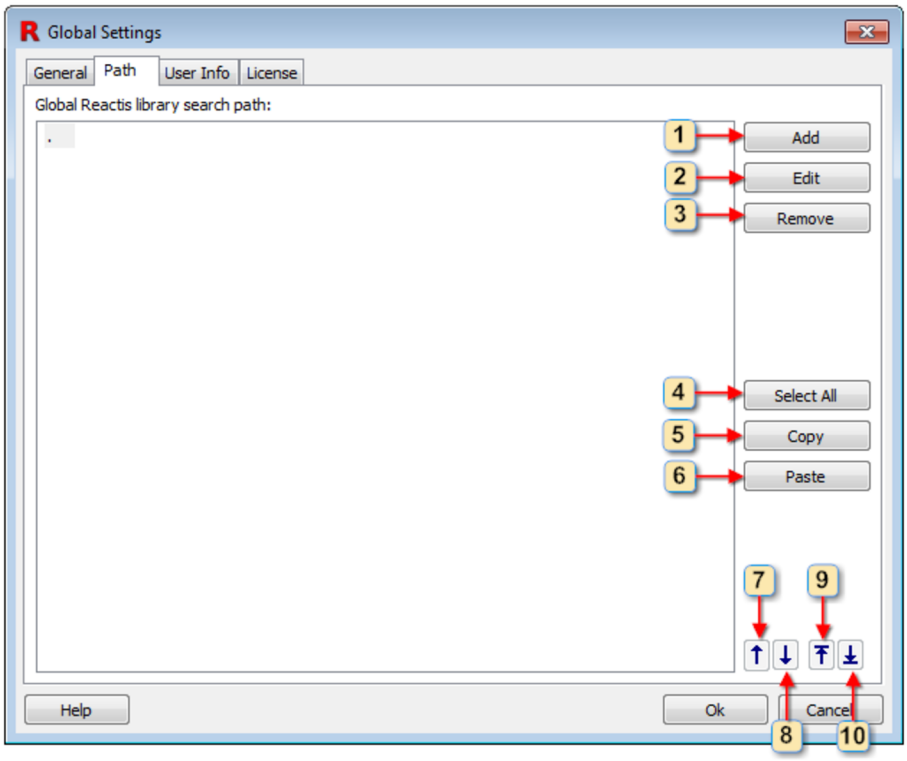

Figure 3.5: The Settings dialog with Path tab selected.

1. Add a new folder to the list.
2. Open a dialog to edit the currently selected folder.
3. Remove the currently selected folder(s) from the list.
4. Select all folders in the list.
5. Copy the currently selected folder(s) to the clipboard.
6. Paste from the clipboard to the list.
7. Move the currently selected folder up one spot in the list.
8. Move the currently selected folder down one spot in the list.

9. Move the currently selected folder to the top of the list.
10. Move the currently selected folder to the bottom of the list.

3.6.3 User Info Settings

The User Info settings tab is shown in Figure 3.6. When Reactis Model Inspector is configured to use a remote license server as described in the next section, information contained in this panel is submitted to the server when Reactis Model Inspector is started, and available to all users who have access to the server.

The list of users occupying licenses at a given time may be obtained using the License tab of the Settings dialog as described below, or using the standalone License Monitor utility included in the Reactis Model Inspector distribution. This utility may be invoked by selecting Reactis Model Inspector V2020.2 → License Manager → License Monitor from the Windows Start menu.

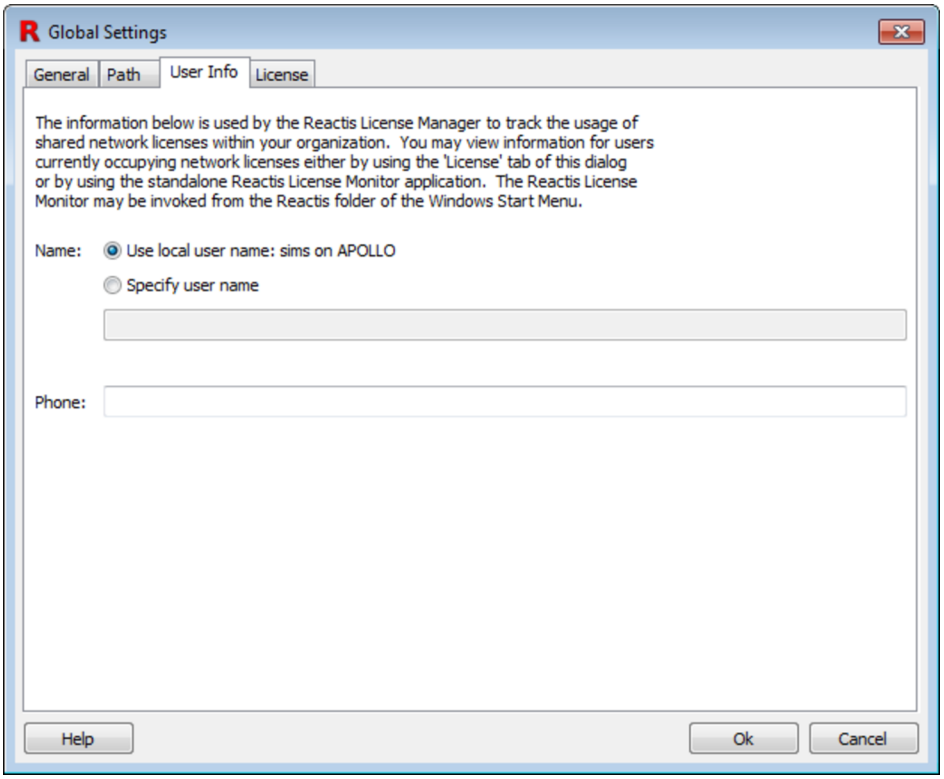

Figure 3.6: The Settings dialog with User Info tab selected.

3.6.4 License Settings

The License tab, shown in Figure 3.7, enables you to query and specify license configuration information. The first two sections display the MAC address of the machine on which Reactis Model Inspector is running and the location of a local license file if one is in use.

The third section of the tab displays a list of servers running the Reactis License Manager. When Reactis Model Inspector is invoked, this list will be searched from top to bottom for an

available license. The lowest portion of the tab displays a list of users currently using licenses for the License Manager/product currently selected in the License servers list.

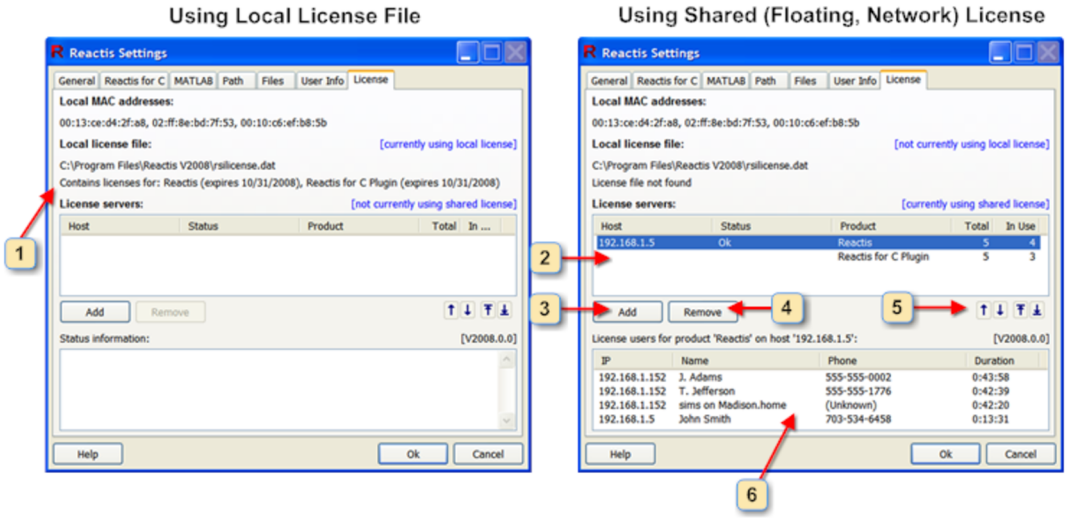

Figure 3.7: The Settings dialog with License tab selected.

Each of the window items labeled in Figure 3.7 is interpreted and used as follows.

1. Information about the contents of the local license file. If there is a problem with the license file, then a description of the error condition is listed here. If no problem exists, then a list of licensed products and their expiration dates is given.

2. This is the list of servers running the Reactis License Manager. Each entry in the list includes the following:

   Host The name or IP address of the server running the License Manager.

   Status The status of the connection to the License Manager.

   For each product managed by the server:

   Product Name of the product (Reactis, Reactis for C Plugin, Reactis Model Inspector, Reactis for C).

   Total The total number of licenses for the product.

   In Use The number of currently occupied licenses for the product.

3. Add a new License Manager to the list.

4. Remove the currently selected License Manager from the list.

5. Move the currently selected License Manager up one spot in the list, down one spot in the list, to the top of the list, or to the bottom of the list.

6. Information regarding the currently selected License Manager is displayed here. If there is a problem with the connection to the License Manager, then a description of the error condition is listed here. If no problem exists, then for each license currently occupied, this section lists:

IP Address. The IP address of the computer on which the Reactis application occupying the license is running.

Name. The contents of the Name field in the User Info settings tab of the person occupying the license.

Phone. The contents of the Phone field in the User Info settings tab of the person occupying the license.

Duration. The length of the time this computer has been holding the license.

Chapter 4

The Reactis Info File Editor

Reactis does not modify the `.mdl` file for a model. Instead the tool stores model-specific information that it requires in a `.rsi` file. The primary way for you to view and edit the data in these files is via the Reactis Info File Editor. In the Reactis Model Inspector, the Info File Editor can be opened in read-only mode to inspect `.rsi` files created in Reactis. The directives contained in the `.rsi` file are grouped into twelve categories. The settings of each category are viewed from a different pane of the Info File Editor. Those panes are as follows:

Harness. This panel contains information and settings for the current test harness.

Inport Types. Constrain the values that the inports assume during simulation, test generation, and validation.

Configuration Variables. Tag certain workspace data items as *configuration variables*.

Test Points. Manipulate model data items designated as *test points*.

Outport Tolerances. Specify tolerance for Simulator to use when comparing an outport value computed by the model against that stored in a test suite.

General. Adjust various settings that determine how a model executes in Reactis (e.g. conditional input branch execution, short circuiting, etc.).

Error Checking. Specify how Reactis should respond to various types of errors (e.g. overflow, NaN, etc.).

Coverage Metrics. Specify the coverage metrics to be used when working with a model.

Excluded Coverage Targets. View and modify the list of targets which have been excluded from coverage.

Validator Objectives. Manipulate *Validator objectives* (assertions, user-defined targets, virtual sources) .

C Code. If using the Reactis for C Plugin, view a list of places in a model that reference C code (S-Functions and Stateflow custom code) and manipulate the settings for white-box analysis of each S-Function.

External EML Functions. This panel lists `.m` files which contain Embedded MATLAB functions called from the model (e.g. from MATLAB Function blocks, Stateflow, Truth Tables).

Callbacks. Specify callbacks to be executed before and/or after a model is loaded.

Search Path. Specify a model-specific search path.

File Dependencies. Specify files on which the model depends.

Although not necessary, the default naming convention assumes that the `.mdl` file and `.rsi` file of a model share the same base name; for example, if the model file is named `cruise.mdl`, then the name of the associated `.rsi` file is assumed to be `cruise.rsi`. A `.rsi` file named differently may be associated with a model by loading the model in Reactis Model Inspector and selecting File → Select Info File....

For detailed descriptions of the contents of each of the twelve panes, please see Chapter 5 of the Reactis User's Guide:

https://reactive-systems.com/doc/reactis-for-simulink/user005.html

Chapter 5

The Reactis Test-Suite Browser

The Test-Suite Browser allows you to view test suites. Figure 5.1 contains an annotated screen shot of the Test-Suite Browser, which displays the test selected in the test selector button (window item 12). The body of the window consists of three tabs:

Test Data Displays a matrix whose first column lists the input ports, test points, and output ports of the model and whose subsequent columns each represent a simulation step. Within each of these latter columns there is a value for each inport, test point, and outport.

Test History Displays history information logged by Reactis for the currently selected test.

Suite History Displays history information logged by Reactis for the test suite.

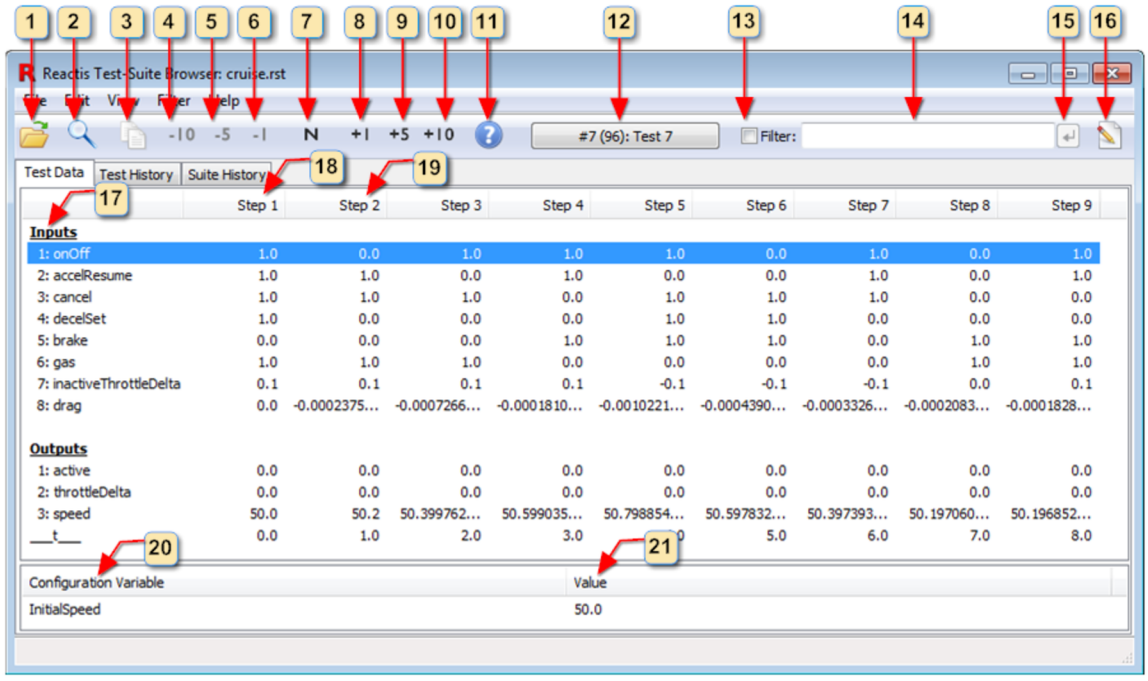

Figure 5.1: The Reactis Tester test-suite browser.

5.1 Labeled Window Items

The following items are labeled in Figure 5.1:

1. Display a file-selection dialog to specify an `.rst` file for loading into the browser.
2. Open a *distribution scope* to display the distribution of values the selected port assumes in the test suite, as shown in Figure 5.2. Alternatively, it is possible to view the values assumed during a single test or group of tests, rather than during the entire test suite. This may be done by right-clicking on the selected port or by using the relevant items in the View menu; see Section 5.2 for details. The button is disabled when a filter is active.
3. If either of the Test History or Suite History tabs are selected, copy any of the selected log information to the clipboard.
4. Go back 10 steps in the current test.
5. Go back 5 steps in the current test.
6. Go back 1 step in the current test.
7. Specify a step in the current test to view.
8. Go forward 1 step in the current test.
9. Go forward 5 steps in the current test.
10. Go forward 10 steps in the current test.
11. Display Reactis Test-Suite Browser help.
12. This pull-down menu enables you to select a test from the currently loaded test suite to view in the browser. The face of the button shows a message of the form #*tnum* (*tsteps*): *tname* where *tnum* is the test number, *tsteps* is the number of steps in the test, and *tname* is the test name. When a filter is active, the text in parentheses changes to *matching* of *tsteps* where *matching* is the number of steps in the test that match the filter.
13. Checking this box activates the filter. When a filter is active only steps for which the filter expression evaluates to true will be displayed.
14. A filter expression may be entered in this text entry box. When a filter is active, the main browser panel shows only test steps for which the expression evaluates to a non-zero value. You activate a filter by:

    - pressing the return key in the entry box, or
    - checking the checkbox to the left of the entry box, or
    - clicking the enter button to the right of the entry box.

    For example, to display the steps in which the brake is active enter `'brake == 1'` and press return.

    The syntax for filter expressions is ANSI C. The variables in the expression may be the names of inports, outports, or test points from the test suite. The elements of a vector

item in the test suite can be accessed using the array indexing notation of the C language (note that indexing is 0-based). The elements of a bus may be accessed using the C structure notation, e.g. *X.Y* if *Y* is an element of bus *X*. Some examples demonstrate the most common operators:

Filter Expression	Result
`(brake == 1) && (active == 1)`	All steps in which the brake is pressed and the cruise control remains active (indicating a serious error)
`(speed < 30) && (active == 1)`	All steps in which cruise is active at a speed less than 30
`(cancel == 1) || (decelSet == 1)`	All steps in which either *cancel* or *decelSet* is pressed
`(inp[0] > 0) && (inp[1] > 0)`	All steps in which the first two elements of vector input *inp* are positive

15. Clicking this button activates the filter. When a filter is active only steps for which the filter expression evaluates to true will be displayed.

16. Clicking this button opens the Filter Editor dialog shown in Figure 5.3. The editor lets you formulate more complicated filters.

17. This column lists the inports, test points, and outports of the model.

18. Values of inports, test points, and outports in step 1 of the current test.

19. Values of inports, test points, and outports in step 2 of the current test.

20. This column lists the name of each configuration variable.

21. This column lists the value of each configuration variable in the current test.

5.2 Menus

File menu. The File menu contains the following entries:

Open... Launches a file-selection dialog for choosing a new `.rst` file to browse.

Close. Close the current `.rst` file.

Exit. Exit the Test-Suite Browser.

Edit menu. The Edit menu contains the following entries:

Copy. If either of the Test History or Suite History tabs are selected, copy any of the selected log information to the clipboard.

Select All. Select all log information in the currently displayed tab.

View menu. The View menu contains the following entries:

10 Steps Back. Go back 10 steps in the current test.

5 Steps Back. Go back 5 steps in the current test.

1 Step Back. Go back 1 step in the current test.

Go to Step... Specify a step in the current test to view.

1 Step Forward. Go forward 1 step in the current test.

5 Steps Forward. Go forward 5 steps in the current test.

10 Steps Forward. Go forward 10 steps in the current test.

Significant Digits (Current Port)... Specify how many significant digits should be used when displaying the values flowing over the currently selected port. If '-1' is specified then the default number of significant digits will be displayed (16 for doubles and 8 for singles).

Significant Digits (All Items)... Specify how many significant digits should be used when displaying the values flowing over all ports in the model. If '-1' is specified then the default number of significant digits will be displayed (16 for doubles and 8 for singles).

Visible Steps... Set the number of test steps to be shown in the main panel of the browser window.

Open Distribution Scope (Whole Suite). For the currently selected port, open a distribution scope to display (as shown in Figure 5.2) the values the port assumes when executing the test suite.

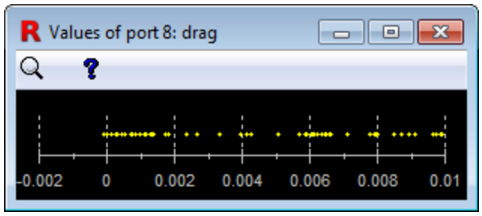

Figure 5.2: The values assumed by port "drag". You may zoom in by clicking in the body of the dialog. Clicking the button in the tool bar zooms to fit.

Open Distribution Scope (Current Test). For the currently selected port, open a distribution scope to display the values the port assumes when executing the currently selected test.

Open Distribution Scope (Selected Tests)... Open a dialog to select a subset of tests from the current test suite; then, for the currently selected port, open a distribution scope to display the values the port assumes when executing the tests in the selected subset.

The dialog for selecting a subset of tests works as follows. Holding down the control key and clicking on different tests enables you to select a group of tests. For example test 1, test 3, and test 5 may be selected by holding down the control key and clicking on each of those three tests. Alternatively, holding down the shift key and clicking on two different tests will add the two tests along with all tests between them to the group. This approach may be used to select tests 1-4 by holding down the shift key and clicking on test 1 and then test 4.

Filter menu. The filter menu lets you display only a subset of rows and columns in the main panel.

Edit Column Filter... Open the Test Step Filter editor shown in Figure 5.3 and described in the subsequent text. The editor lets you create a filter to identify test steps in the test suite that satisfy a particular condition. For example, all test steps in which the cruise control is active.

Enable Column Filter. This menu entry toggles whether or not the column filter is active. When active, only steps satisfying the filter expression are displayed in the main panel.

Show Hidden Columns. You can toggle this entry to cause Reactis to display steps for which the filter expression is false as well as those for which it is true. Steps for which the filter expression evaluates to false will have their values and step number displayed within parentheses.

Hide Selected Rows. Hide the currently selected rows. Note you may select a range of rows by left-clicking on a row, then holding down the shift key and left-clicking on a second row. All rows from the first row selected to the second row selected (inclusively) will be selected. Alternatively, a group of rows can be selected by holding down the control key and left-clicking on each row to be included in the group.

Unhide Selected Rows. Unhide the currently selected rows. Note, to select hidden rows you must first toggle the Show Hidden Rows menu entry.

Show Hidden Rows. Display hidden rows in the main panel. Note, in this mode, when a row is hidden, the item name and all values will be grayed out.

Help menu. The Help menu contains the following entries:

Contents. Go to the table of contents in the documentation.

Index. Go to the index in the on-line documentation.

Test-Suite Browser. Display Test-Suite Browser help.

5.3 Test Step Filter Editor

The Test Step Filter Editor, shown in Figure 5.3, lets you formulate filters to search for test steps in a suite that satisfy a given condition.

The labeled items in the figure work as follows.

1. Enable or disable the filter.

2. Enter a filter expression in this entry box. When a filter is active, the Test Suite Browser displays only test steps for which the expression evaluates to a non-zero value. The syntax for the expression is ANSI C. The variables in the expression take their values from inports, outports, or test points from the test suite. The elements of a vector item in the test suite can be accessed using the array indexing notation of the C language (note that indexing is 0-based). Some examples demonstrate the most common operators:

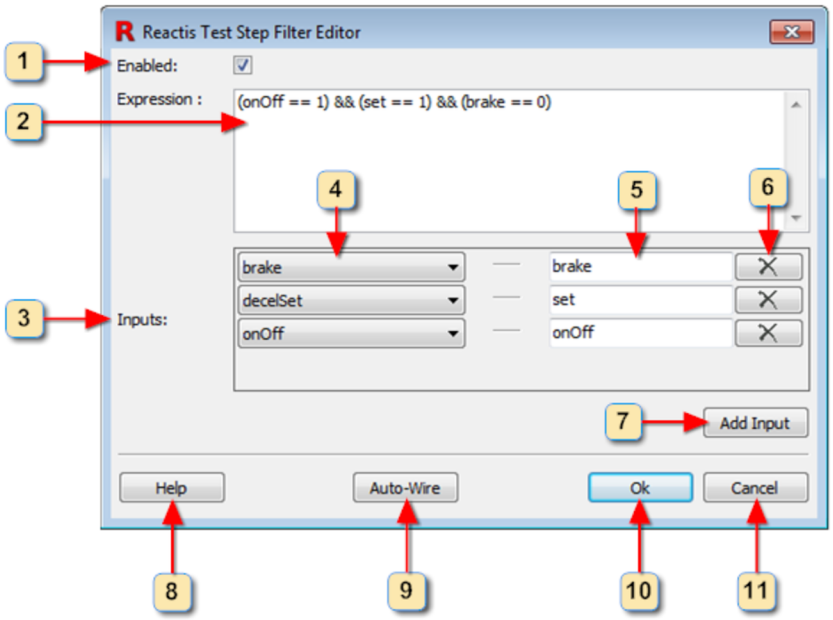

Figure 5.3: The Filter Editor lets you craft more complex filters to search test suites for steps satisfying a particular condition.

Filter Expression	**Result**
`(brake == 1) && (active == 1)`	All steps in which the brake is pressed and the cruise control remains active (indicating a serious error)
`(speed < 30) && (active == 1)`	All steps in which cruise is active at a speed less than 30
`(cancel == 1) || (decelSet == 1)`	All steps in which either *cancel* or *decelSet* is pressed
`(inp[0] > 0) && (inp[1] > 0)`	All steps in which the first two elements of vector input *inp* are positive

3. The variables used in the expression are defined in this section. They are termed inputs because the filter expression can be viewed as the body of a function and the inputs are the formal input parameters of the function. To apply the filter function to a given test step, values from the test step serve as actual parameters fed into the filter function for evaluation of the filter expression. The elements of a vector item in the test suite can be accessed using the array indexing notation of the C language (note indexing is 0-based). For example, if an input *inp* is a vector of length 3, then we can check for steps in which elements of *inp* are positive with the filter expression:

   ```
   (inp[0] > 0) && (inp[1] > 0) && (inp[2] > 0)
   ```

   The elements of a bus may be accessed using the C structure notation, e.g. $X.Y$ if Y is an element of bus X.

4. Each pulldown in this column includes an entry for each input, output, test point, and sample time in the test suite. The item selected will be fed into the input named in the

next column when evaluating the filter expression.

5. Each entry box in this column names an input/variable to be used in the filter expression.
6. Clicking this button deletes the filter input.
7. Clicking this button adds a new filter input.
8. Open help for the Filter Editor.
9. Clicking this button causes Reactis to attempt to automatically compute the entries in the inputs section. For each variable in the filter expression an input row will be added if it does not already exist. Furthermore the input will be wired (pulldown in the first column selected) to the input, output, or test point of the same name if one exists in the test suite.
10. Save the changes made in the Filter Editor and close the dialog.
11. Discard the changes made in the Filter Editor and close the dialog.

Chapter 6

Glossary of Reactis Concepts

In addition to browsing the model itself, Reactis Model Inspector lets you examine various testing and validation artifacts created in Reactis. Reactis is a separate product of Reactive Systems that supports automated testing and validation of Simulink models. This chapter provides a brief description of Reactis artifacts that can be examined in Reactis Model Inspector. For more complete descriptions, please see the *Reactis User's Guide*, available at:

https://reactive-systems.com/doc/reactis-for-simulink/

Throughout this chapter, the notation *RUG:9* denotes Chapter 9 and the notation *RUG:7.2* denotes Section 7.2 of the *Reactis User's Guide*.

Configuration Variable. A *configuration variable* is a workspace variable tagged in Reactis to be changed in between tests, but not during a test. See RUG:5.4.

Inport Type. Reactis lets you assign a type or constraint to a top-level inport that will limit the values that will be used for the inport for testing and validation. For example, you can limit values for an inport to a range by specifying a minimum and maximum value. Type constraints are manipulated from the Inport Types tab of the Reactis Info File Editor and may be viewed from that tab in Reactis Model Inspector. See RUG:5.3.

Reactis Makefile. A Reactis Makefile (.rsm file) lists the C files that comprise a library that implements an S-Function or is used by an S-Function or Stateflow custom code. See RUG:16.

Reactis Simulator. Simulator is a component of Reactis that supports interactive simulation and debug of models. See RUG:7.

Reactis Tester Component of Reactis that generates comprehensive yet compact test suites from models. See RUG:8.

Reactis Validator Component of Reactis that checks if a model satisfies its requirements. See RUG:9.

Test Point A Test point is an internal data item, such as a Simulink block, Simulink signal, or Stateflow variable, that Reactis treats as a *virtual output*. Reactis Tester records the values of test points in test suites and Reactis Simulator flags any differences between the values computed by a model for a test point and those stored in a test suite. See RUG:5.5.

Test Suite Test suites are the most important artifact created with Reactis. A test suite is comprised of a set of tests. Each test in the suite corresponds to a simulation run of the model that captures the top-level inputs and outputs at each simulation step. Comprehensive yet compact test suites are generated automatically by Reactis Tester. Reactis Simulator lets you execute tests in order to debug your model. When Reactis Validator finds a violation of a requirement it returns a test that you can execute in Simulator in order to diagnose the problem. Reactis Model Inspector lets you inspect tests using the Test Suite Browser.

Validator Objective Using Reactis Validator you can instrument your model with Validator Objectives in order to check that your model satisfies its requirements. Validator Objectives include assertions, user-define targets, and virtual sources. See RUG:9.

Virtual Source A *virtual source* is a Validator objective that lets you control a top-level inport of your model. See RUG:9.

Chapter 7

Revision History

Different versions of Reactis Model Inspector are labeled as shown in Figure 7.1 and described below.

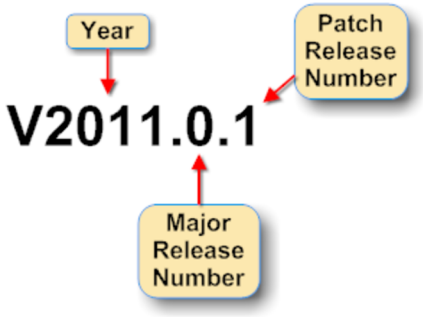

Figure 7.1: Version labels begin with a "V" and include three parts: a year, a major release number, and a patch release number. The parts are numbers separated by decimal points. By convention, trailing zeros are omitted.

Major Releases. A new version of Reactis Model Inspector is released each year and labeled by a "V" followed by the four-digit year, for example V2012. Each label for an intra-year release includes a suffix consisting of a decimal point followed by a *major release number*; for example V2012.1, V2012.2, etc. will label the releases during 2012 that follow V2012.

Beta Releases. RSI sometimes makes beta releases available to customers interested in evaluating the newest features of Reactis Model Inspector. Beta releases do not undergo as much testing as major releases do. By convention, beta releases have odd numbered major release numbers. For example, V2012.1, V2012.3, ... denote beta releases.

Patch Releases. Stable releases may be "patched." The label for a patch release is constructed by extending the label for the major release to be patched with a suffix that includes a second decimal point and a *patch release number*. Some examples clarify the scheme:

V2015.0.1	denotes	the first patch release for V2015
V2014.2.3	denotes	the third patch release for V2014.2

7.1 Patches Mailing List Archive

Patches to the Reactis product family are posted to the Reactive Systems website every four to six weeks on average. To view a summary recent changes, please view the archives of the Reactis Model Inspector Patches mailing list available at:

https://reactive-systems.com/mlists.msp?lid=2

7.2 V2020.2 (18 December 2020)

The V2020 release of Reactis Model Inspector adds the following new features:

- Support for MATLAB R2020b
- Improved block parameter display (right-click on block and select View Block Parameters)
- Improved GUI appearance on high-resolution displays. Automatically adjust if screen resolution changes.

7.3 Previous Major Release Dates

Major releases of Reactis Model Inspector prior to V2020 have occurred on the following dates:

Version	Release Date		
V2020	17	July	2020
V2019.2	20	December	2019
V2019	28	June	2019
V2018.2	20	December	2018
V2018	27	June	2018
V2017.2	22	December	2017
V2017	7	July	2017
V2016.2	22	December	2016
V2016	20	June	2016
V2015.2	18	December	2015
V2015	30	June	2015
V2014.2	19	December	2014
V2014	25	June	2014
V2013.2	17	December	2013
V2013	9	August	2013
V2012	26	November	2012

Index