RTI Monitor

User's Manual

Version 7.2.0

Trademarks

Copy and Use Restrictions

Third-Party Software

Notices

Deprecations and Removals

Any deprecations or removals noted in this document serve as notice under the Real-Time Innovations, Inc. Maintenance Policy #4220 and/or any other agreements by and between RTI and customer regarding maintenance and support of RTI's software.

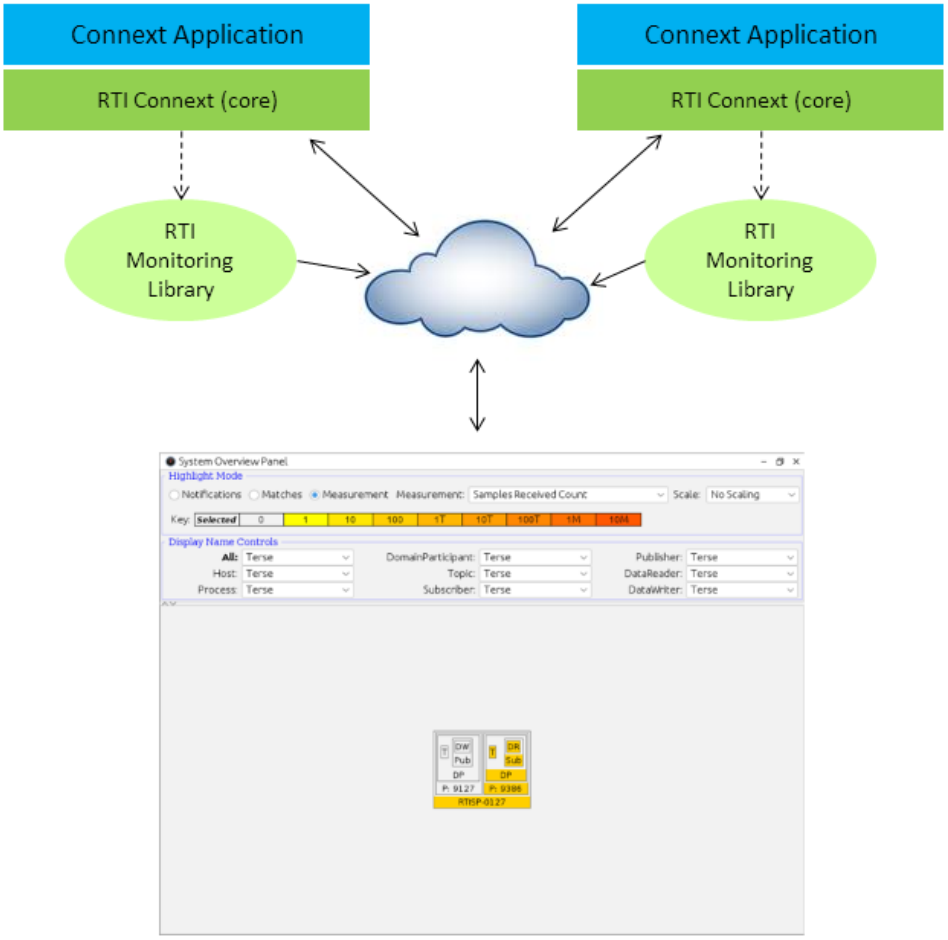
Deprecated means that the item is still supported in the release, but will be removed in a future release. *Removed* means that the item is discontinued or no longer supported. By specifying that an item is deprecated in a release, RTI hereby provides customer notice that RTI reserves the right after one year from the date of such release and, with or without further notice, to immediately terminate maintenance (including without limitation, providing updates and upgrades) for the item, and no longer support the item, in a future release.

Technical Support
Real-Time Innovations, Inc.
232 E. Java Drive
Sunnyvale, CA 94089
Phone: (408) 990-7444
Email: support@rti.com
Website: https://support.rti.com/

Contents

Chapter 1 Welcome

RTI® Monitor is a graphical tool that displays monitoring data from *RTI Connext®* applications.



RTI Monitor

Monitor will help you:

- **Understand your system** with an easy-to-use graphical view into your entire *Connext* application.
- **Verify your design** by making sure the entities in your *Connext* applications are communicating as expected.
- **Tune performance** by providing deep statistics on every aspect of the middleware's operation.
- **Optimize integration** with detailed information on every entity in your system.
- **Monitor real-time operation** with a dashboard of tools to see traffic patterns, errors, lost samples, and more.

You can run *Monitor* on the same host as the *Connext* application or on a different host.

To enable a *Connext* application to provide monitoring data to *Monitor*, the application needs to use the *RTI Monitoring Library* plug-in included with *Connext*.

Monitoring is enabled in the application by setting values in the DomainParticipant's PropertyQosPolicy (programmatically or through an XML QoS profile). See the Monitoring Library section of the RTI Connext Core Libraries User's Manual.

Connext notifies *Monitoring Library* every time an entity is created/deleted or a QoS is changed. *Monitoring Library* also periodically queries the status of all entities. *Monitoring Library* sends all the data to *Monitor* once it gets the data from the *Connext* application.

1.1 Paths Mentioned in Documentation

The documentation refers to:

- **<NDDSHOME>**

  This refers to the installation directory for *RTI® Connext®*. The default installation paths are:
  - macOS® systems:
    /Applications/rti_connext_dds-7.2.0
  - Linux systems, non-*root* user:
    /home/<*your user name*>/rti_connext_dds-7.2.0
  - Linux systems, *root* user:
    /opt/rti_connext_dds-7.2.0
  - Windows® systems, user without Administrator privileges:
    <*your home directory*>\rti_connext_dds-7.2.0
  - Windows systems, user with Administrator privileges:
    C:\Program Files\rti_connext_dds-7.2.0

You may also see **$NDDSHOME** or **%NDDSHOME%**, which refers to an environment variable set to the installation path.

Wherever you see **<NDDSHOME>** used in a path, replace it with your installation path.

Note for Windows Users: When using a command prompt to enter a command that includes the path **C:\Program Files** (or any directory name that has a space), enclose the path in quotation marks. For example:

```
"C:\Program Files\rti_connext_dds-7.2.0\bin\rtiddsgen"
```

Or if you have defined the **NDDSHOME** environment variable:

```
"%NDDSHOME%\bin\rtiddsgen"
```

- ***<path to examples>***

By default, examples are copied into your home directory the first time you run *RTI Launcher* or any script in **<NDDSHOME>/bin**. This document refers to the location of the copied examples as ***<path to examples>***.

Wherever you see ***<path to examples>***, replace it with the appropriate path.

Default path to the examples:

- macOS systems: **/Users/<*your user name*>/rti_workspace/7.2.0/examples**
- Linux systems: **/home/<*your user name*>/rti_workspace/7.2.0/examples**
- Windows systems: ***<your Windows documents folder>*\rti_workspace\7.2.0\examples**

  Where 'your Windows documents folder' depends on your version of Windows. For example, on Windows 10, the folder is **C:\Users\<*your user name*>\Documents**.

Note: You can specify a different location for **rti_workspace**. You can also specify that you do not want the examples copied to the workspace. For details, see *Controlling Location for RTI Workspace and Copying of Examples* in the *RTI Connext Installation Guide*.

Chapter 2 Starting Monitor

There are two ways to start *Monitor:*

- From *RTI Launcher's* **Tools** tab, click on the **Monitor** icon.
- From a command prompt:
  - On Linux and macOS systems:

```
cd <NDDSHOME>/bin
./rtimonitor
```

  - On Windows systems:

```
cd <NDDSHOME>\bin
rtimonitor
```

Monitor accepts the command-line options in Table 2.1 Command-line Options.

Table 2.1 Command-line Options

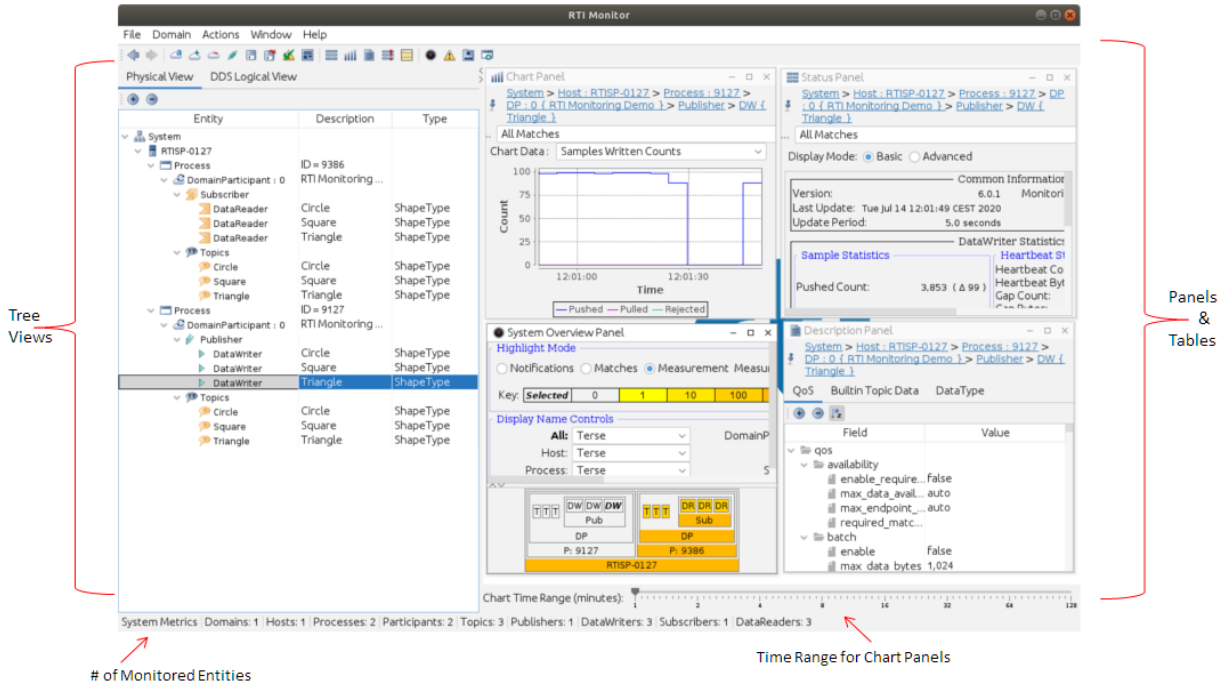
Option	Description
-aggregationPeriodSeconds <seconds>	*Monitor* periodically goes through all the monitored entities in the system (this information is saved in its own database) to calculate aggregated statistics and states. This value controls that minimum period (specified in seconds). Default: 5 seconds
-help	Displays all command-line options.
-historyDepth <value>	*Monitor* saves some statistics' history, so it can be displayed in the charts. This option controls how much historical data (number of samples) is saved per monitoring topic. Default: 12 samples
-ignoreTypeConflicts	Instructs *Monitor* to ignore any type conflicts. In *Monitor*, type conflicts are based on type-code equality rather than type compatibility. This command-line option can be useful if you have types that have different type-code but are compatible. Default: Not specified (do not ignore type conflicts)

Table 2.1 Command-line Options

Option	Description
-initialDomainIds <domain_id_list>	Specifies which domains *Monitor* will join when it starts up. <domain_id_list> is a list of domain IDs, each separated by a comma. To specify multiple domain IDs on a Windows system, enclose the comma-separated IDs in quotation marks. For example: -initialDomainIds “31, 32”. Default: If not specified, you will be prompted to enter a domain ID when *Monitor* starts.
-matchRefreshPeriodSeconds <seconds>	Specifies the period at which to refresh the system overview panel's matches. Default: 5 seconds
-notificationHistoryDepth <value>	Specifies the number of notifications to keep per entity. Default: 12 notifications
-pruneDeadObjectsPeriodSeconds <seconds>	Sets the period at which *Monitor* should clean up user-interface objects (such as the Host, and Process nodes in the tree views) that are no longer current (have no more children nodes in the tree view). This value should be increased when dealing with very large systems where the time to complete discovery is longer than the default value of 3 seconds. Default: 3 seconds
-spawnReadThreads	Instructs *Monitor* to use multiple threads (according to the number of cores on the host) to retrieve data from its DataReaders (which contain monitoring data). This is typically only needed for very large systems. Default: Not specified (use a single read thread to retrieve data at a period of 1 second)
-verbosity <value>	Sets the verbosity for *Monitor* and *Connext*. • 0: silent (both *Connext* and *Monitor*) • 1: errors (both *Connext* and *Monitor*) • 2: warnings (*Monitor* only) • 3: warnings (both *Connext* and *Monitor*) • 4: information (*Monitor* only) • 5: tracing (*Monitor* only) • 6: tracing (both *Connext* and *Monitor*) Default: 1

Chapter 3 Using Monitor

Monitor consists primarily of tree views and panels. There is also a toolbar for easy access to the most commonly used commands. This chapter provides more details on *Monitor*'s components.

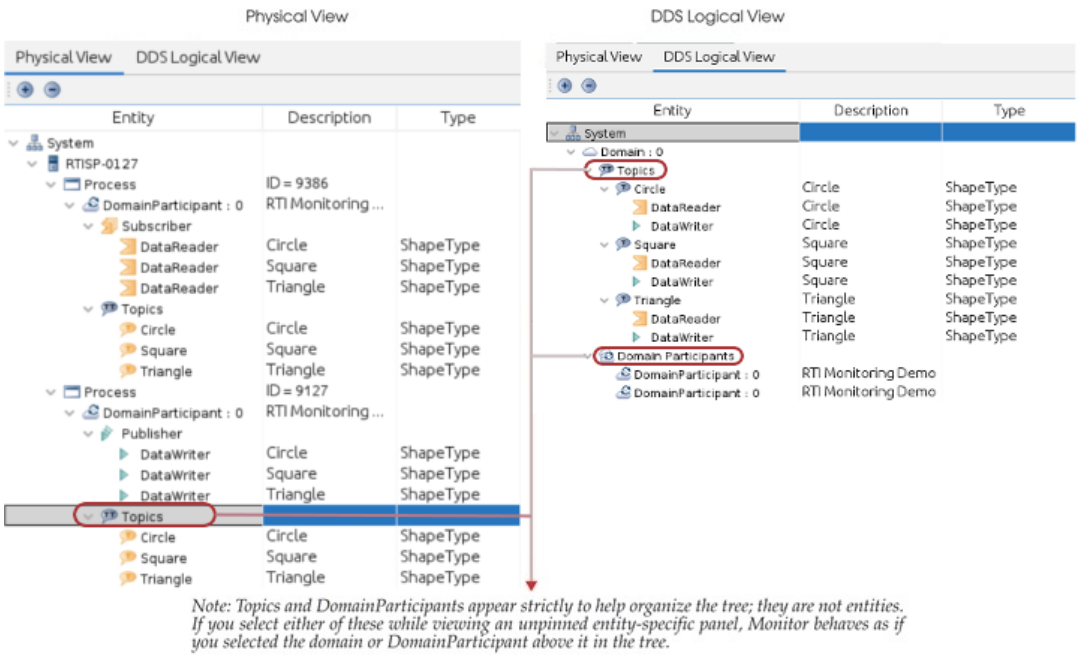


3.1 Tree Views

There are two tree views on the left:

- The **Physical View** displays all entities, arranged by their physical containment.
- The **DDS Logical View** displays all the entities, arranged by domains and topics.

Figure 3.1: Physical and DDS Logical Views

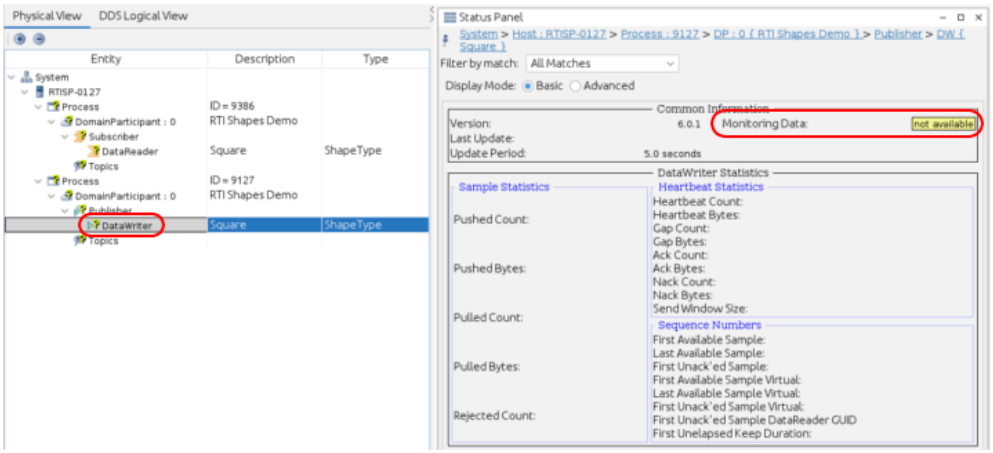


When you select an entity in one of the panels, it becomes selected in the tree.

Similarly, when you select an entity in a tree, any entity-specific panels are updated to display information for the newly selected entity. One exception to this is if you use the **pin** button in the upper-left corner of the panel. When a panel is pinned to an entity, it will periodically receive updated data for the pinned entity—even when another entity is selected in the tree.

Note: You may see a yellow question mark icon on entities in the tree views:

- The question mark always appears on DomainParticipants and the 'Topics' under them that are created with the *Monitoring Library* option **rti.monitor.config.new_participant_domain_id**. These DomainParticipants are created solely to publish monitoring topic data; they do not publish monitoring data about themselves.
- For other types of entities, the question mark means monitoring data is not available. For information on enabling *Monitoring Library* in a *Connext* application, see the RTI Connext Core Libraries User's Manual.



3.2 Working with Monitor's Panels

Monitor has several panels that display monitoring data in graphical and tabular form. Some panels show data for a specific selected entity, while others show system-wide information:

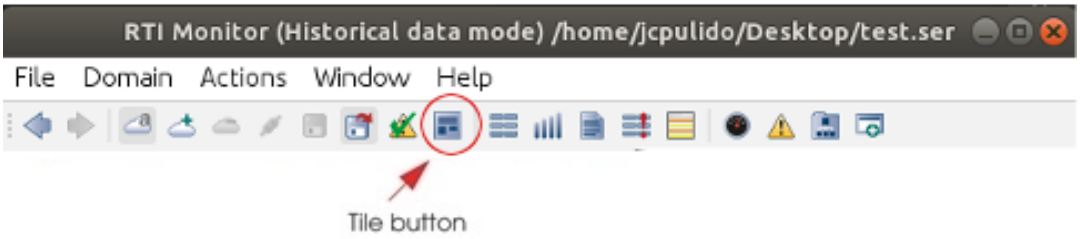
- Entity-Specific Panels
  - 3.3.1 Status Panel on the next page
  - 3.3.2 Chart Panel on page 13
  - 3.3.3 Description Panel on page 14
  - 3.3.4 Notifications Panel on page 17
  - 3.3.5 Distributed Logger Panel on page 21
- System-wide Panels and Tables
  - 3.4.1 System Overview Panel on page 25
  - 3.4.2 All Notifications Table on page 28
  - 3.4.3 System Types Table on page 30
  - 3.4.4 Processes Table on page 30

You can create these panels by:

- Using the **Window, Create Panel** option from the menu
- Right-clicking an entity and selecting from the pop-up menu (entity-specific panels only)
- Clicking a button on the toolbar

You may have multiple panels of each type open at the same time.

Panels can be arranged by various options in the **Window** menu. There is also a **Tile** button in the toolbar.



3.3 Entity-Specific Panels

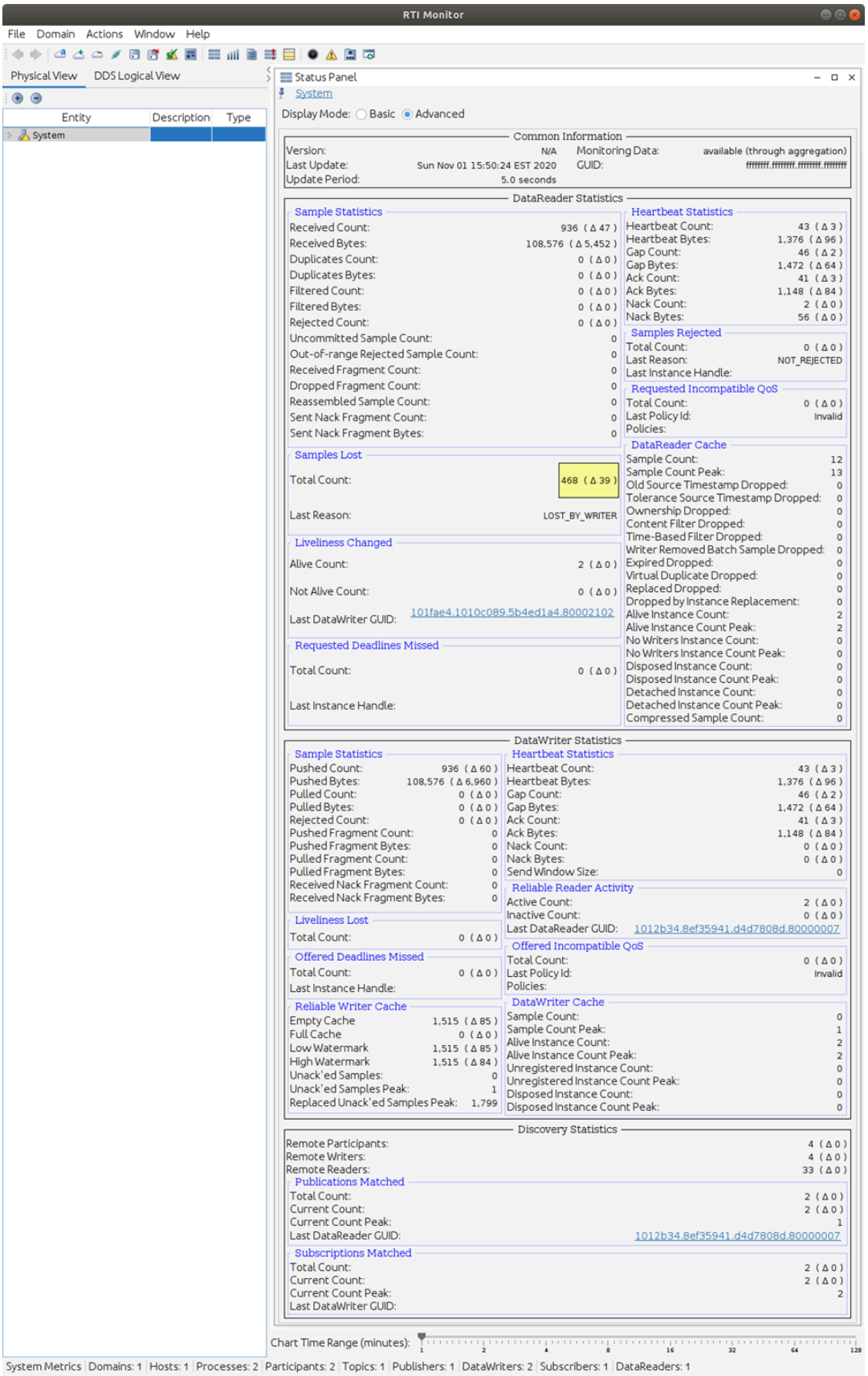
The contents for entity-specific panels change to show whatever entity is currently selected in the tree view. You can, however, 'pin' a panel to an entity to prevent it from switching contents; to do so, use the **pin** button  in the upper-left corner of the panel. When a panel is pinned to an entity (you will see the pin button changed to ), it will periodically receive updated data for the same pinned entity—even when another entity is selected in the tree. The entity for the entity-specific panel (pinned or unpinned) is indicated by the entity hierarchy list at the top of the panel.

The **backward**  and **forward**  buttons in the toolbar can be used navigate through the entity-selection history.

3.3.1 Status Panel

The Status panel  displays real-time statistics for the selected entity.

Figure 3.2: Status Panel


RTI Monitor
File Domain Actions Window Help
Physical View
DDS Logical View
Entity
Description
Type
System
Status Panel
System
Display Mode: Basic Advanced
Common Information
Version: N/A
Monitoring Data: available (through aggregation)
Last Update: Sun Nov 01 15:50:24 EST 2020
GUID: ffffffff.ffffffff.ffffffff.ffffffff
Update Period: 5.0 seconds
DataReader Statistics
Sample Statistics
Received Count: 936 (Δ 47)
Received Bytes: 108,576 (Δ 5,452)
Duplicates Count: 0 (Δ 0)
Duplicates Bytes: 0 (Δ 0)
Filtered Count: 0 (Δ 0)
Filtered Bytes: 0 (Δ 0)
Rejected Count: 0 (Δ 0)
Uncommitted Sample Count: 0
Out-of-range Rejected Sample Count: 0
Received Fragment Count: 0
Dropped Fragment Count: 0
Reassembled Sample Count: 0
Sent Nack Fragment Count: 0
Sent Nack Fragment Bytes: 0
Heartbeat Statistics
Heartbeat Count: 43 (Δ 3)
Heartbeat Bytes: 1,376 (Δ 96)
Gap Count: 46 (Δ 2)
Gap Bytes: 1,472 (Δ 64)
Ack Count: 41 (Δ 3)
Ack Bytes: 1,148 (Δ 84)
Nack Count: 2 (Δ 0)
Nack Bytes: 56 (Δ 0)
Samples Rejected
Total Count: 0 (Δ 0)
Last Reason: NOT_REJECTED
Last Instance Handle:
Requested Incompatible QoS
Total Count: 0 (Δ 0)
Last Policy Id: Invalid
Policies:
Samples Lost
Total Count: 468 (Δ 39)
Last Reason: LOST_BY_WRITER
DataReader Cache
Sample Count: 12
Sample Count Peak: 13
Old Source Timestamp Dropped: 0
Tolerance Source Timestamp Dropped: 0
Ownership Dropped: 0
Content Filter Dropped: 0
Time-Based Filter Dropped: 0
Writer Removed Batch Sample Dropped: 0
Expired Dropped: 0
Virtual Duplicate Dropped: 0
Replaced Dropped: 0
Dropped by Instance Replacement: 0
Alive Instance Count: 2
Alive Instance Count Peak: 2
No Writers Instance Count: 0
No Writers Instance Count Peak: 0
Disposed Instance Count: 0
Disposed Instance Count Peak: 0
Detached Instance Count: 0
Detached Instance Count Peak: 0
Compressed Sample Count: 0
Liveliness Changed
Alive Count: 2 (Δ 0)
Not Alive Count: 0 (Δ 0)
Last DataWriter GUID: 101fae4.1010c089.5b4ed1a4.80002102
Requested Deadlines Missed
Total Count: 0 (Δ 0)
Last Instance Handle:
DataWriter Statistics
Sample Statistics
Pushed Count: 936 (Δ 60)
Pushed Bytes: 108,576 (Δ 6,960)
Pulled Count: 0 (Δ 0)
Pulled Bytes: 0 (Δ 0)
Rejected Count: 0 (Δ 0)
Pushed Fragment Count: 0
Pushed Fragment Bytes: 0
Pulled Fragment Count: 0
Pulled Fragment Bytes: 0
Received Nack Fragment Count: 0
Received Nack Fragment Bytes: 0
Heartbeat Statistics
Heartbeat Count: 43 (Δ 3)
Heartbeat Bytes: 1,376 (Δ 96)
Gap Count: 46 (Δ 2)
Gap Bytes: 1,472 (Δ 64)
Ack Count: 41 (Δ 3)
Ack Bytes: 1,148 (Δ 84)
Nack Count: 0 (Δ 0)
Nack Bytes: 0 (Δ 0)
Send Window Size: 0
Reliable Reader Activity
Active Count: 2 (Δ 0)
Inactive Count: 0 (Δ 0)
Last DataReader GUID: 1012b34.8ef35941.d4d7808d.80000007
Liveliness Lost
Total Count: 0 (Δ 0)
Offered Deadlines Missed
Total Count: 0 (Δ 0)
Last Instance Handle:
Offered Incompatible QoS
Total Count: 0 (Δ 0)
Last Policy Id: Invalid
Policies:
Reliable Writer Cache
Empty Cache 1,515 (Δ 85)
Full Cache 0 (Δ 0)
Low Watermark 1,515 (Δ 85)
High Watermark 1,515 (Δ 84)
Unack'ed Samples: 0
Unack'ed Samples Peak: 1
Replaced Unack'ed Samples Peak: 1,799
DataWriter Cache
Sample Count: 0
Sample Count Peak: 1
Alive Instance Count: 2
Alive Instance Count Peak: 2
Unregistered Instance Count: 0
Unregistered Instance Count Peak: 0
Disposed Instance Count: 0
Disposed Instance Count Peak: 0
Discovery Statistics
Remote Participants: 4 (Δ 0)
Remote Writers: 4 (Δ 0)
Remote Readers: 33 (Δ 0)
Publications Matched
Total Count: 2 (Δ 0)
Current Count: 2 (Δ 0)
Current Count Peak: 1
Last DataReader GUID: 1012b34.8ef35941.d4d7808d.80000007
Subscriptions Matched
Total Count: 2 (Δ 0)
Current Count: 2 (Δ 0)
Current Count Peak: 2
Last DataWriter GUID:
Chart Time Range (minutes): 1 2 4 8 16 32 64 128
System Metrics | Domains: 1 | Hosts: 1 | Processes: 2 | Participants: 2 | Topics: 1 | Publishers: 1 | DataWriters: 2 | Subscribers: 1 | DataReaders: 1

It displays statuses of the selected entity, or an aggregation of all the statuses of all the entities that belong to that selected item. For example, if you select a DataWriter, the statuses are just for that entity. If you select a Publisher, the statuses are an aggregation of those for all DataWriters that belong to that Publisher. Aggregation calculation period can be controlled by the **-aggregationPeriodSeconds** command-line parameter (see Table 2.1 Command-line Options for details).

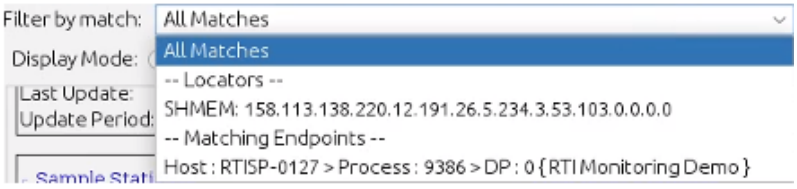
Basic and **Advanced** options are provided. **Basic** data only includes Sample Statistics and Heartbeat Statistics (for reliable readers or writers only). The **Advanced** option shows all the available statuses for the entity (some of the data is only available for reliable readers or writers).

Warnings and Error Statuses

Warnings and errors are checked for some of the statuses; warnings are highlighted in yellow, errors are in red. See 3.3.4 Notifications Panel on page 17 for details on which statuses are checked for warnings or errors. To clear the warnings and errors status of ALL entities in the system, select the  button from the toolbar or **Actions, Clear All Notifications** from the menu.

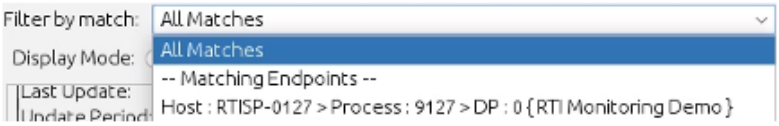
The **Common Information** section shows you general information about the entity—the GUID for the entity for this panel, when the data was last updated, and the current update period. The version of the remote entity is also provided (except for System, Host, and Process) in this panel. If the version of the remote entity is incompatible with *Monitor*, it will be marked as a warning (the background will be yellow). In this situation, very little data will be available about the remote entity (generally, just the information discovered about the DomainParticipant).

Filter by match only appears for DataWriters and DataReaders. If anything other than **All matches** is selected, the data shown in the Status panel will only include data that belongs to the matching kind that you have selected—a subset of the data for the entire entity.



Above: Example filter options for a Data
The locators are for the transport
matching endpoint is a DataReader.

Below: Example filter options for a Data
The matching endpoint is a DataWriter.



Filter options for DataWriters

- All matches
- A selected locator
- A selected matching endpoint (DataReader)

Filter options for DataReaders:

- All matches
- A selected matching endpoint (DataWriter)

Note regarding negative delta values:

When a DDS entity, such as a DataReader, leaves the network, associated counts such as the Topic samples received counts may show negative delta values. For example:

Support we have a Topic, foo, which receives 10 samples per second and there are two DataReaders. At time 0, DataReader 1 has received 1,000 samples and DataReader 2 has received 100 samples, for a total of 1,100 samples. Five seconds later (the default publishing period of the Monitoring Library), an update for both DataReaders is available and shows 100 more samples received, for a total of 1,200 (two DataReaders getting 10 samples per second over the course of five seconds).

Now at time 5, we have 1,200 total samples received and we delete DataReader 1. At the next update (an aggregation within *Monitor's* UI), the total number of samples received has gone from 1,200 to 250 (DataReader 1's removal results in -1,000 samples and DataReader 2 has received 50 more samples during that period). The aggregation calculation then reports 250 - 1200 => -950 samples received delta.

3.3.2 Chart Panel

The Chart panel  graphs the selected statistics (on the Y axis) over time (the X axis) for the selected entity.

You can control the time range with the slider at the bottom of the main window, and other chart properties by right-clicking within the chart area.

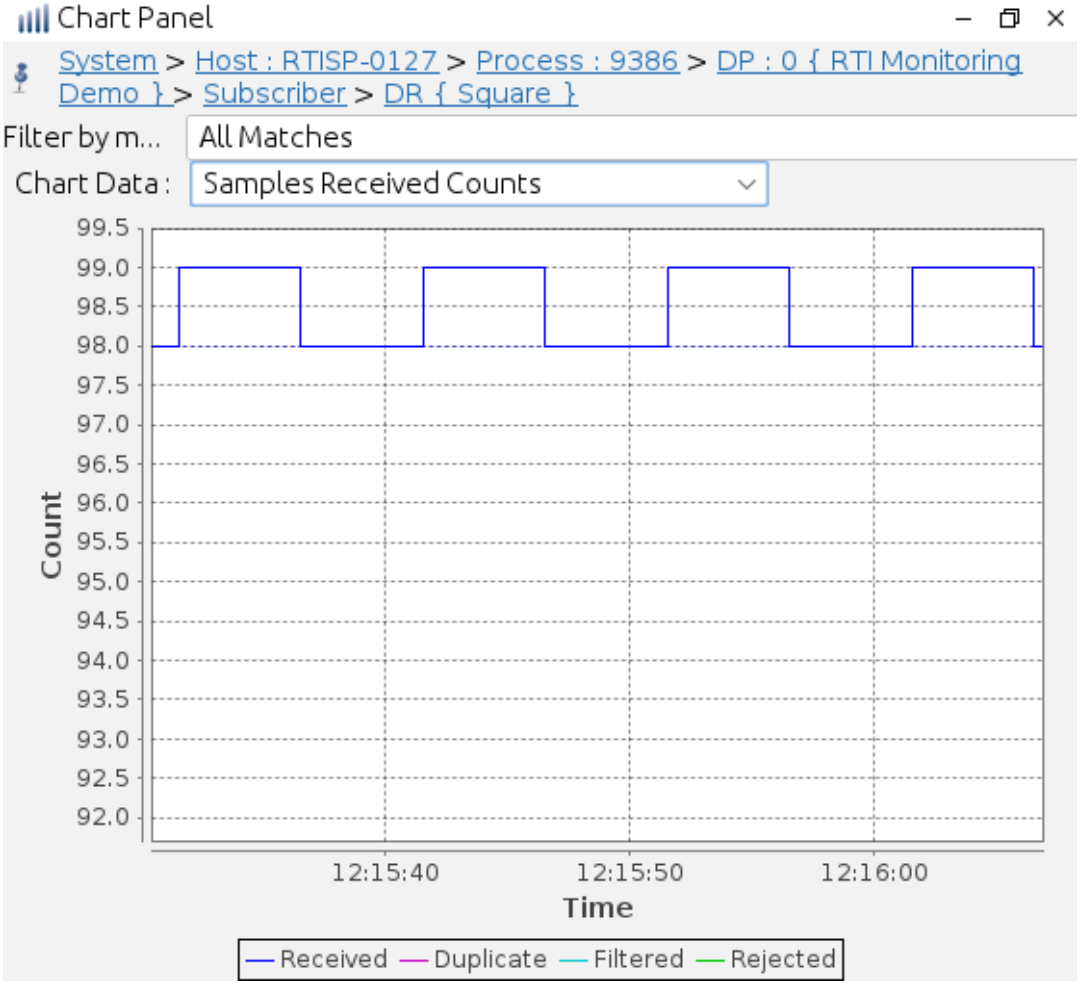
Filter by match only appears for DataWriters and DataReaders—3.3.1 Status Panel on page 9 describes the choices.

The **Chart Data** options depend on the type of the selected entity.

The number of samples that can be displayed in the chart is controlled by the **-historyDepth** command-line option (see Table 2.1 Command-line Options for details).

To plot multiple chart data for the same entity at the same time, create multiple Chart Panels.

Figure 3.3: Chart Panel

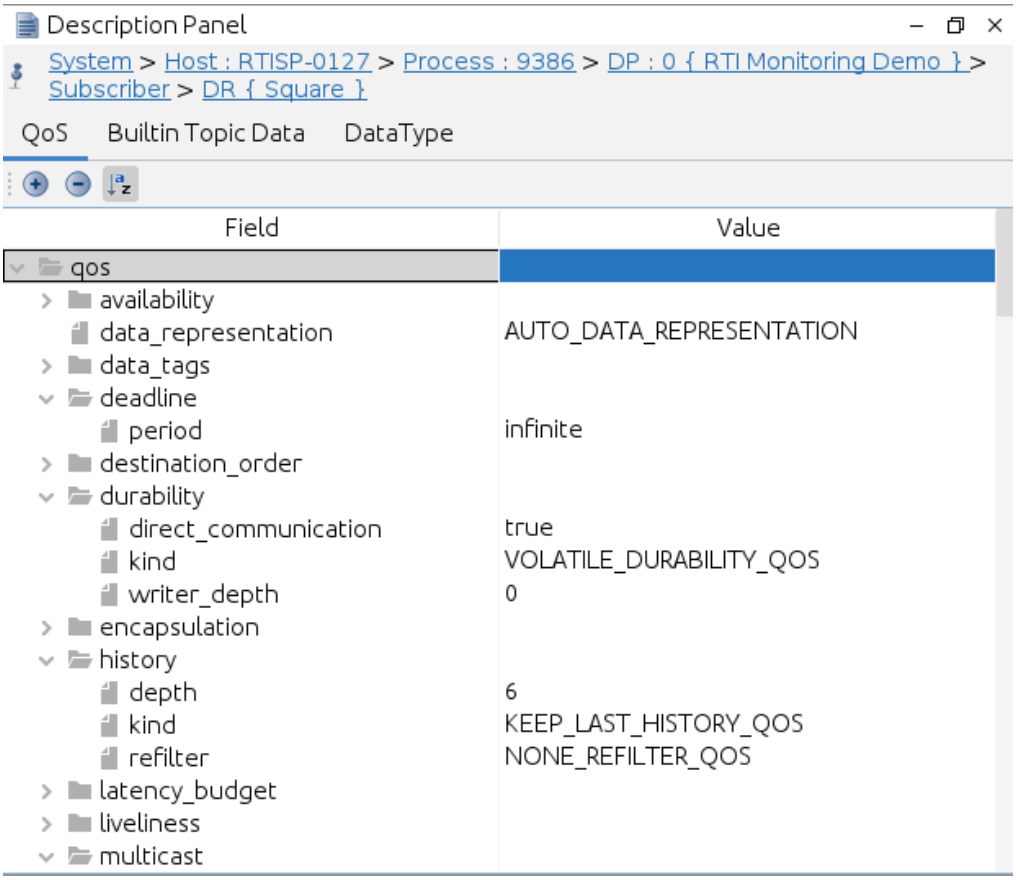


3.3.3 Description Panel

The Description panel's contents depend on what is selected in the tree view. There are three tabs which may appear:

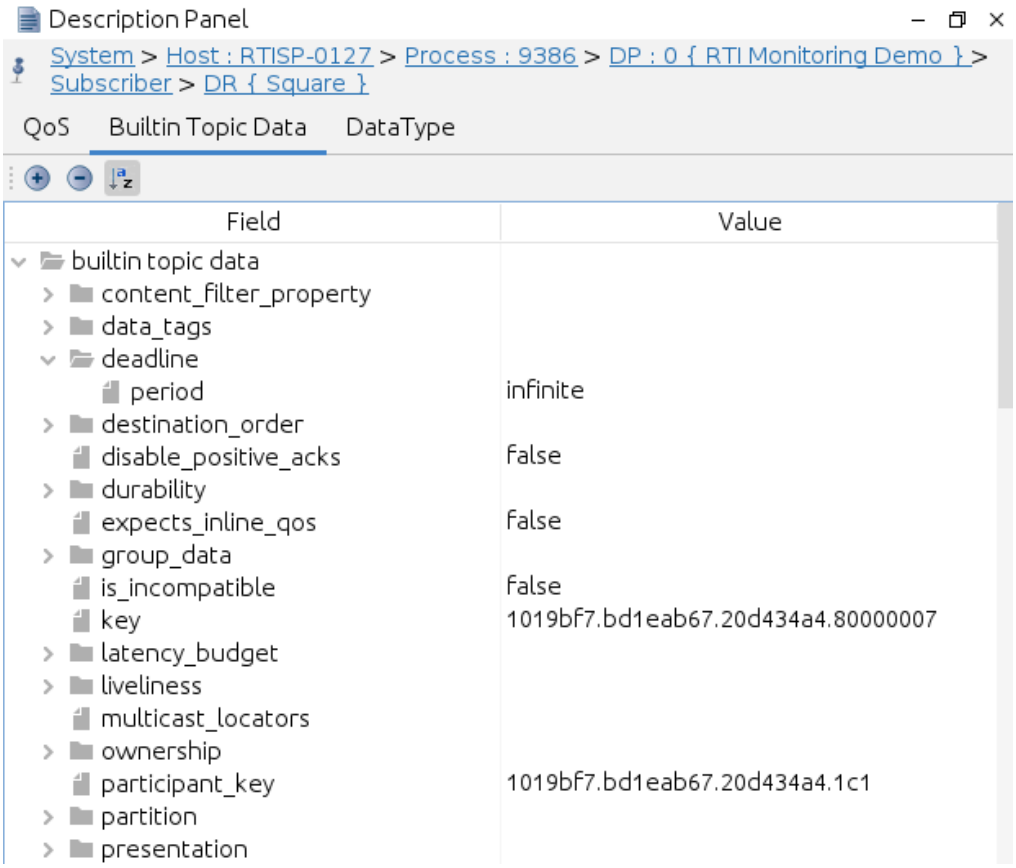
- **QoS**
  (for all *Connext* entities)—Shows the QoS settings for the selected entity.

Figure 3.4: Description Panel's QoS Tab



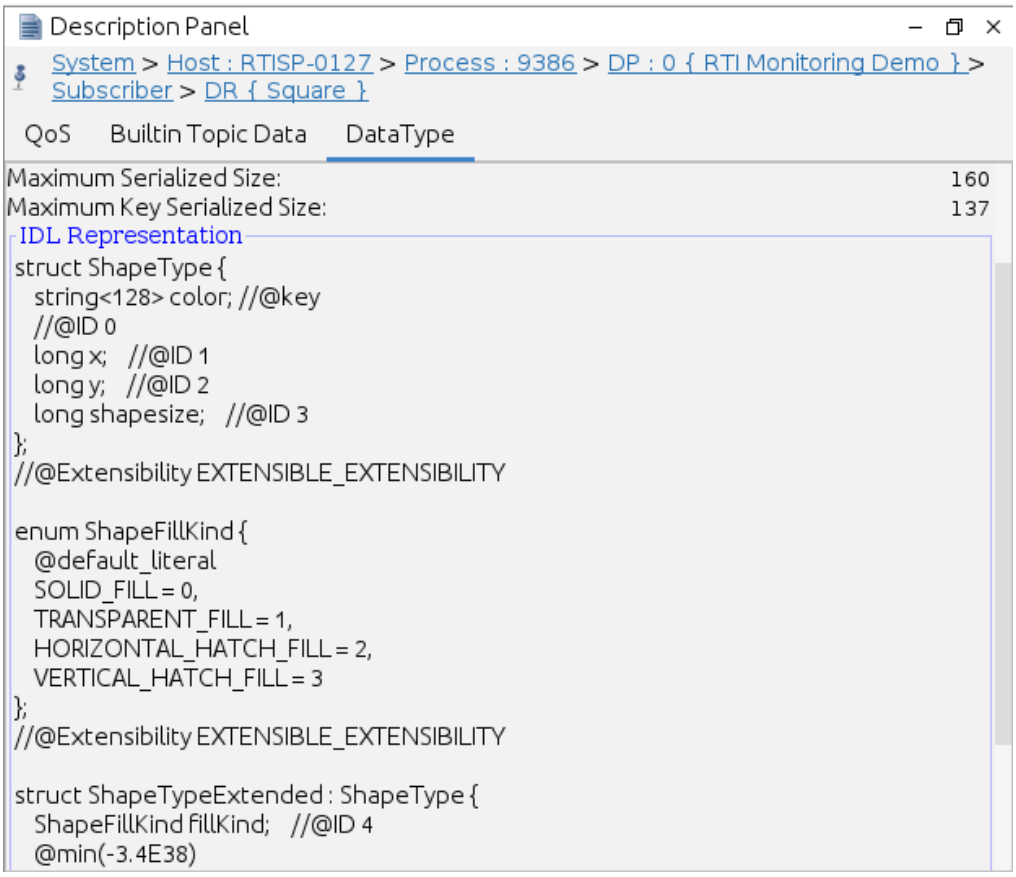
- **Builtin Topic Data**
  (for DomainParticipants, DataWriters, and DataReaders)—Shows the propagated QoS in the builtin topic for the selected entity.

Figure 3.5: Description Panel's Builtin Topic Data Tab



- **Data Type**
  (for DataWriters and DataReaders)—Shows the type name and ID; type code serialized size; and IDL representation of the associated data type:

Figure 3.6: Description Panel's Data Type Tab

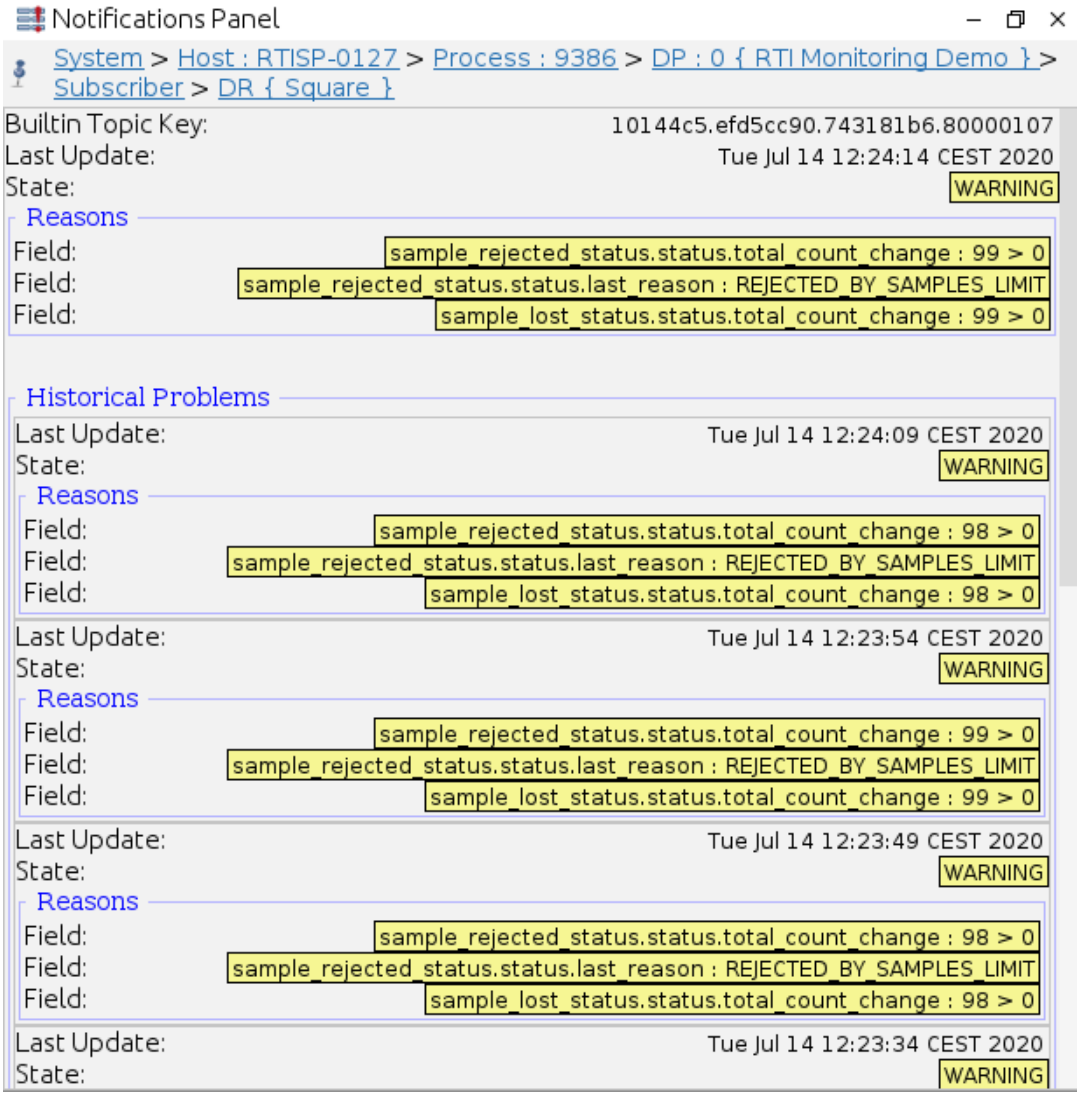


The Description panel is not applicable when a system, host, or process is selected, since they are not DDS entities.

3.3.4 Notifications Panel

The Notifications panel displays the selected entity's current status (normal, warning or error) and a historical list of all related alarm statuses. Warnings are highlighted in yellow, errors are in red.

Figure 3.7: Notifications Panel



Clearing Notifications

To clear the warnings and error status of ALL entities in the system, select the button from the toolbar or **Actions, Clear All Notifications** from the menu.

Historical statuses will never be cleared. The number of saved historical statuses is controlled by the **-notificationHistoryDepth** command-line option (see for details).

Table 3.1 Warning and Error Conditions lists the conditions that are considered warnings or errors.

Table 3.1 Warning and Error Conditions

Entity	Conditions	Warning or Error
DataReader	Type Inconsistency (see 3.3.4.2 Understanding Type Consistency on the next page)	Error
	Incompatible QoS	Error
	Samples rejected	Error
	Deadlines missed	Warning
	Liveliness lost	Warning
	Samples lost	Warning
DataWriter	Using push_on_write = false with best-effort reliability or an asynchronous publisher	Error
	Type conflicts (equality comparison)	Error
	Incompatible QoS	Error
	Inactivated DataReaders	Warning
	Liveliness lost	Warning
	Deadlines missed	Warning
DomainParticipant	On same host as another DomainParticipant that does not agree on using shared memory	Error
Topic	Inconsistent topic status	Error

Type conflicts might be ignored if the **-ignoreTypeConflicts** command-line option is used (see Table 2.1 Command-line Options).

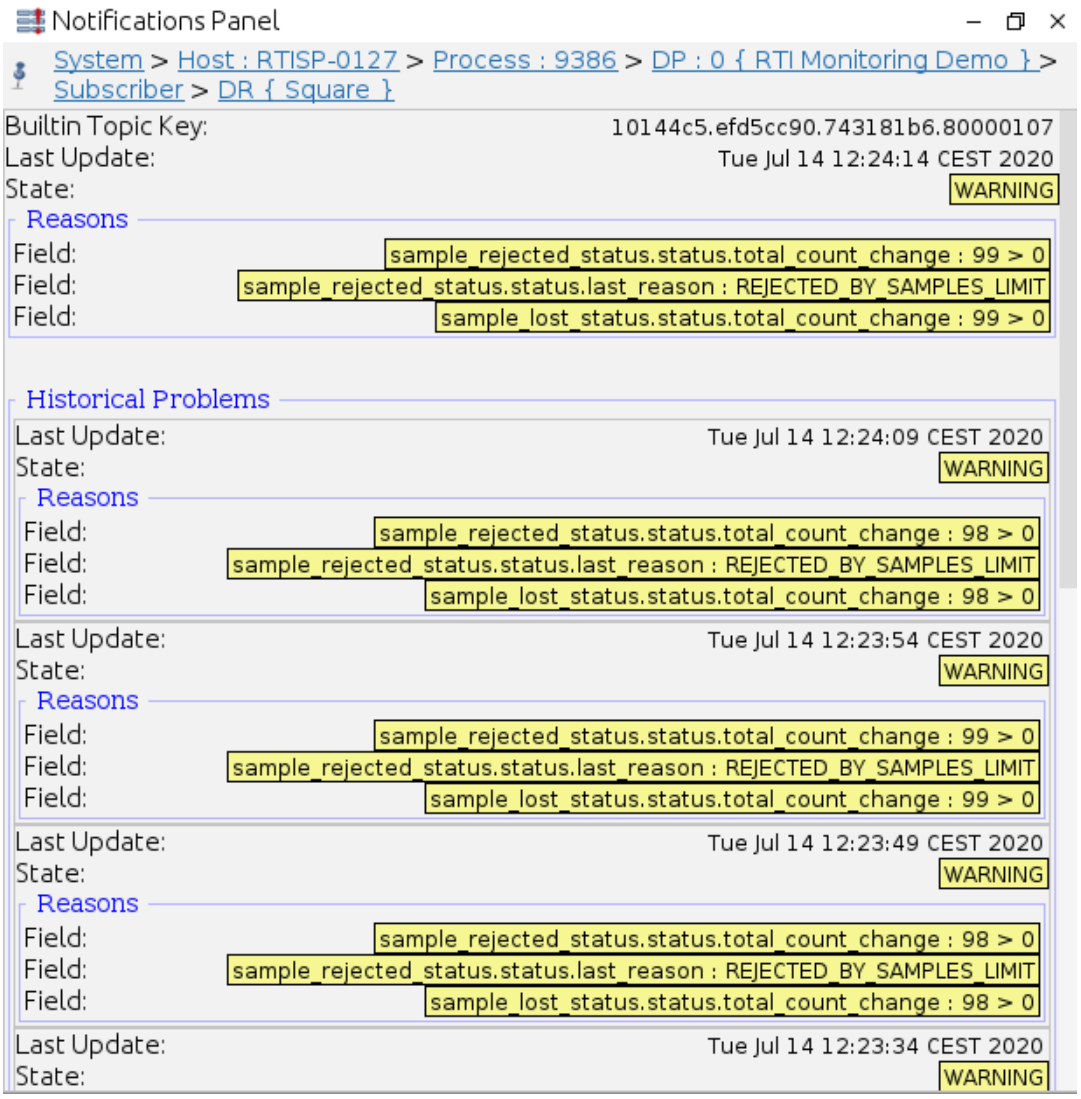
3.3.4.1 Additional Information from RTI Distributed Logger

RTI Distributed Logger is a library that enables applications to publish log messages to *Connext*. It is part of *Connext*.

If a *Connext* application uses *Distributed Logger*, the log messages it sends to *Monitor* are used as an integral part of the entity state kept for the associated Process. You can see this in Figure 3.8: Notifications Panel: Integrating Log Message Level with Entity State on the next page, where log messages have changed the Process' state to **Error** because there are error-level log messages. This is a simple but powerful way to monitor the health of a distributed system with minimal integration work. If the application already tracks its state, it can write updates to the state for Warning and Error to the log. Those will be picked up by *Monitor* and reflected in the display.

See also: 3.3.5 Distributed Logger Panel on page 21.

Figure 3.8: Notifications Panel: Integrating Log Message Level with Entity State



Note: *Monitor* cannot downgrade entity state from Error to Warning to Normal. You can make this change explicitly with the **Clear All Notifications** button on the toolbar.

3.3.4.2 Understanding Type Consistency

Monitor includes partial support for the OMG 'Extensible and Dynamic Topic Types for DDS' specification, version 1.3. This section assumes that you are familiar with Extensible Types and you have read the RTI Connext Core Libraries Extensible Types Guide.

Monitor proactively checks for type consistency. This is done as new types are discovered from DataWriters and DataReaders. If there are type inconsistencies, they will be added to the notifications for the DataWriter and DataReader. The type-consistency checking considers TypeObject information described in the OMG specification.

How Monitor determines type consistency:

1. If the TypeObject information is missing for either the DataWriter or DataReader:
    a. If the registered type names are different, then they are inconsistent.
    b. If the registered type names are equal:
        - If the TypeCode information is missing for either the DataWriter or DataReader, or if the TypeCodes are equal, they are consistent.
        - Otherwise, they are inconsistent.
2. If the TypeObject information is present for both the DataWriter and DataReader:
    a. If the DataReader's TypeConsistencyEnforcementQosPolicy's **kind** is set to DISALLOW_TYPE_COERCION:
        - If the types are not structurally identical, then they are inconsistent.
        - Otherwise, they are consistent.
    b. If the DataReader's TypeConsistencyEnforcementQosPolicy's **kind** is set to ALLOW_TYPE_COERCION:
        - If the DataReader type is not assignable to the DataWriter type, then they are inconsistent.
        - Otherwise, they are consistent.

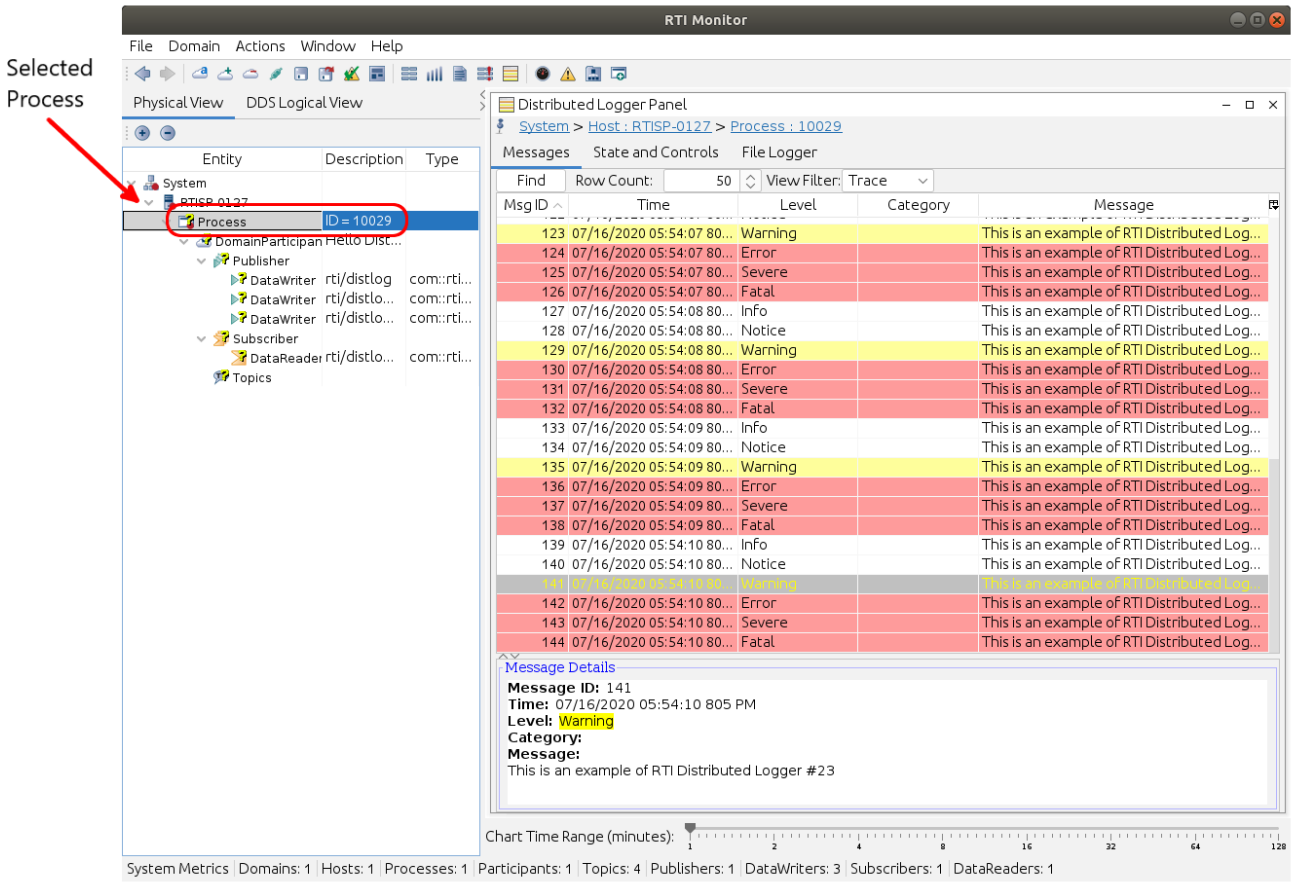
3.3.5 Distributed Logger Panel

Monitor's Distributed Logger panel  allows you to see messages from *Distributed Logger*, a separate library that can be included with *Connext* applications. You can also use this panel to control *Distributed Logger*.

Distributed Logger is integrated into *Monitor's* entity state (normal, warning, and error) tracking feature. Therefore error and warning messages logged to an application's logging system and sent to *Monitor* through *Distributed Logger* will change a process' status to Error and Warning, respectively.

The Distributed Logger panel is associated with Process entities. Therefore it only displays information when a Process is selected from the Physical View and that Process is running an instance of *Distributed Logger*.

Figure 3.9: Distributed Logger Panel



As seen in Figure 3.9: Distributed Logger Panel above, the Distributed Logger panel has three tabs:

- 3.3.5.1 Messages Tab below
- 3.3.5.2 State and Controls Tab on the next page
- 3.3.5.3 File Logger Tab on page 24

When a Process containing an instance of *Distributed Logger* is selected, the cached log messages populate the table in the **Messages** tab, and the **State and Controls** and **File Logger** tabs are updated to reflect the state of the *Distributed Logger* instance.

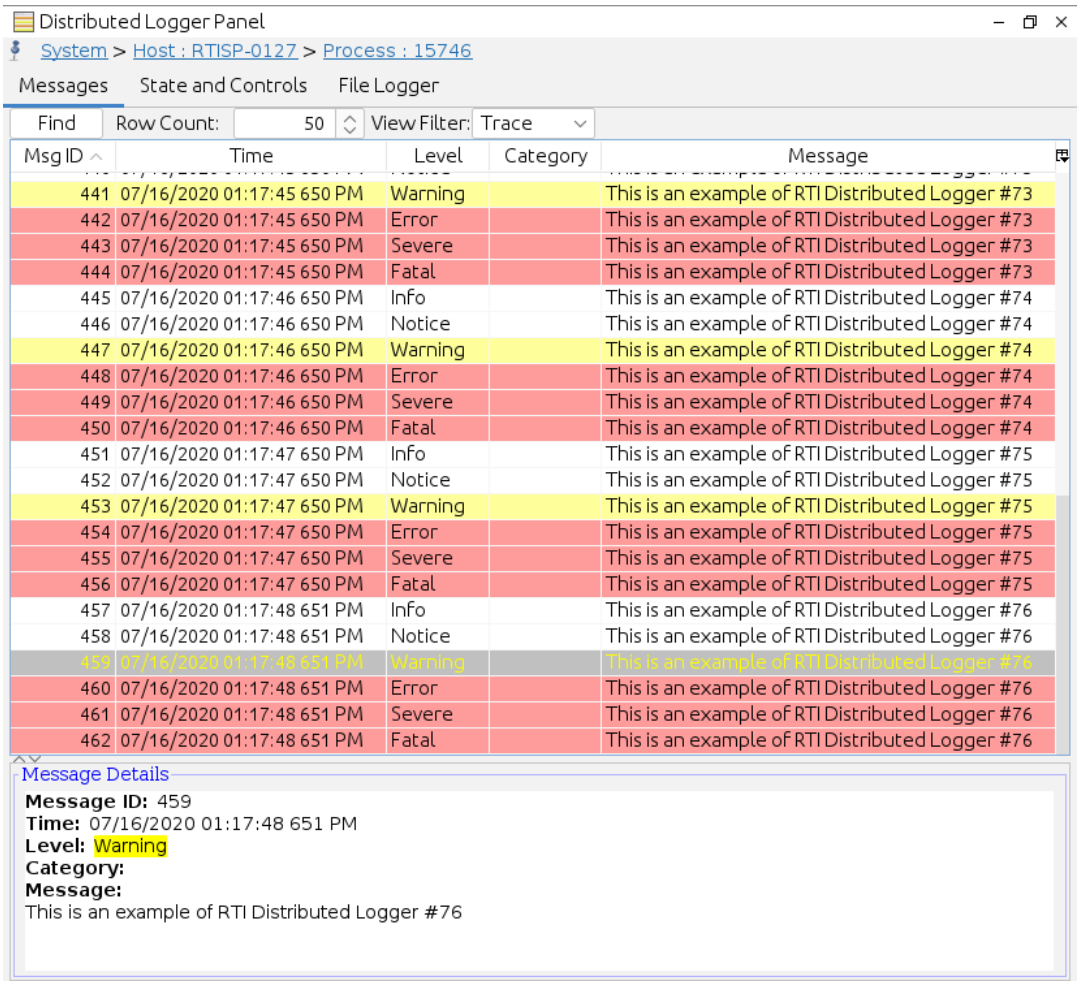
3.3.5.1 Messages Tab

The Messages Tab, shown in Figure 3.9: Distributed Logger Panel above and Figure 3.10: Distributed Logger Panel's Messages Tab on the next page, displays a table containing the log messages from the currently selected Process. The messages can include those logged using the application's logging lib-

rary, RTI Logger, as well as the standard out and standard error of the application, depending on how the application configured *Distributed Logger*.

By default, the messages are presented sorted based on the order in which they were written. By clicking on the column headers, you can re-sort the table to meet your needs. The panel also has a "Find" button to do simple string searches, a control to limit the number of rows which are displayed, and a view filter which shows filters messages that are less severe than the selected level. At the bottom of the panel is a detailed display for the selected messages from the table. This is primarily useful when the log message contains multiple lines (only a single line is displayed in the table).

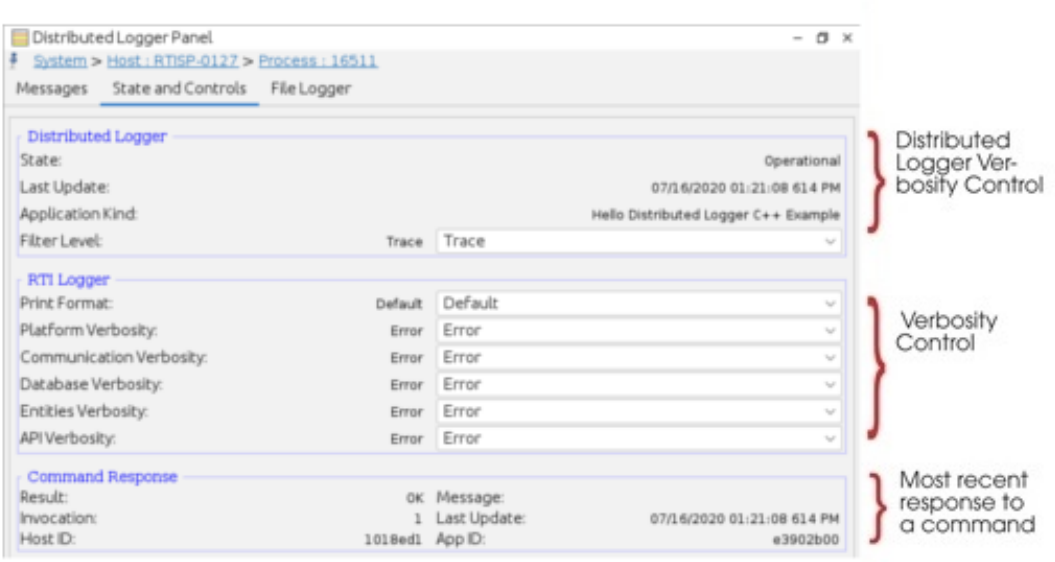
Figure 3.10: Distributed Logger Panel's Messages Tab



3.3.5.2 State and Controls Tab

The State and Controls tab is shown in Figure 3.11: Distributed Logger Panel's State and Controls Tab on the next page. This tab provides a way to control the verbosity of the Distributed Logger instance directly from *Monitor*. It also provides control over the RTI Logger verbosity and shows the most recent (if any) response to commands it has processed.

Figure 3.11: Distributed Logger Panel's State and Controls Tab



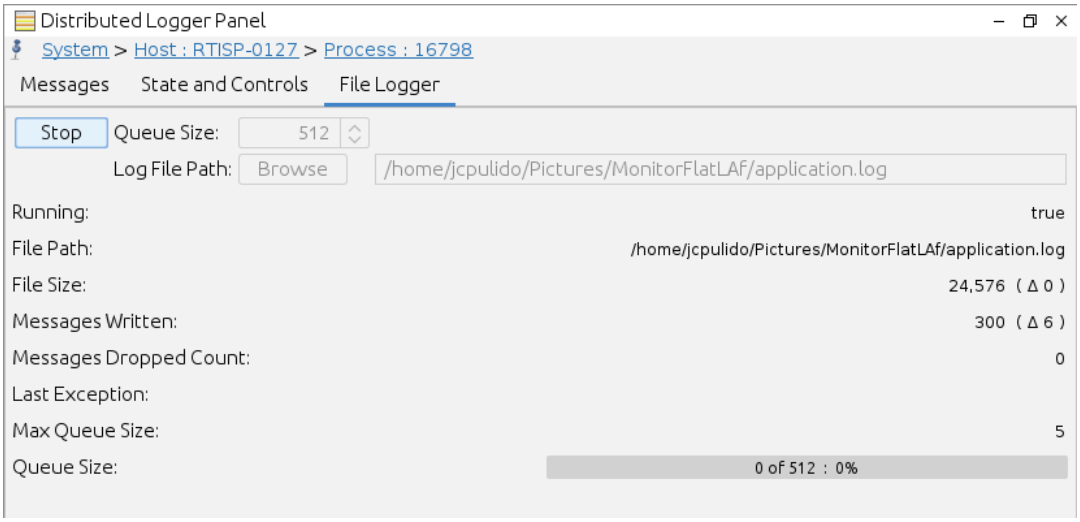
3.3.5.3 File Logger Tab

The File Logger tab, shown in Figure 3.12: Distributed Logger Panel's File Logger Tab on the next page, provides a way to save log messages for a single Process to a file. This is especially useful when issues are noticed for a Process and you want to capture the output to share with others for analysis. Another more general-purpose way to save log messages to a file is provided as an example with *Distributed Logger*, see the RTI Connext Core Libraries User's Manual.

The File Logger tab provides inputs for the file path and the queue size. There are displays that show how many messages have been written as well as dropped (due to queue size). The current and maximum queue sizes are also displayed.

Note: The **File Size** may not update as quickly as the **Messages Written** count. In fact, the **File Size** is usually zero until several messages have been written. This is because buffering is used to increase the throughput performance while writing to the file.

Figure 3.12: Distributed Logger Panel's File Logger Tab



3.4 System-Wide Panels and Tables

System-wide panels and tables show a summary of the states of the whole system.

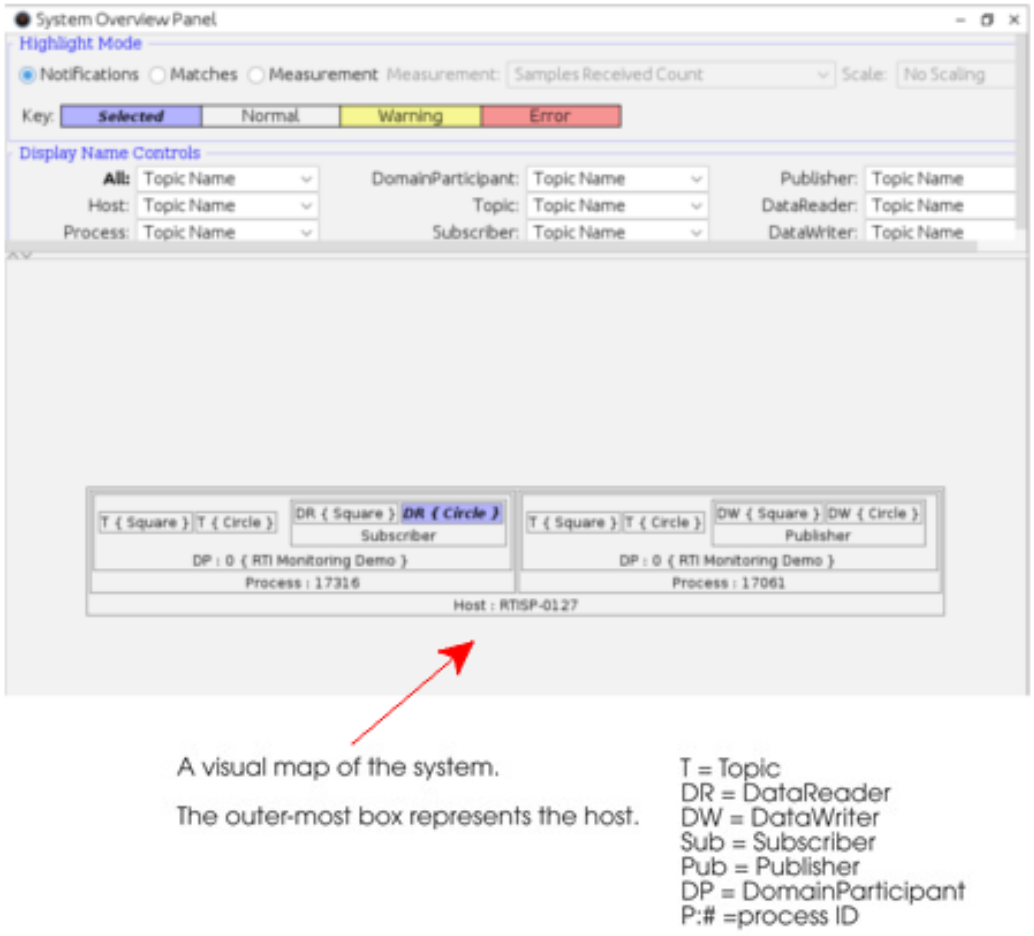
3.4.1 System Overview Panel

The System Overview panel  displays a map of the entities in the system.

You can change the criteria for how items in the map are highlighted by selecting the **Notifications**, **Matches**, or **Measurement** option. A **Key** is provided to indicate the meaning of the different highlights in the map.

If the **Notifications** option is selected, the map will show all the entities in the system and their colors will show if they are in normal, warning or error mode.

Figure 3.13: System Overview Panel (Notifications Option)



If the **Matches** option is selected, the map will show all the entities that are currently matched and all the entities that are currently unmatched due to potential errors.

For an 'ideal match,' opposing endpoints (DataWriters and DataReaders) must have the same domain ID, same topic name, and belong to compatible partitions (if any are specified). This list of ideal matches is compared to the list of actual matches received from *Monitoring Library* to determine which entities are marked as matches or mismatches in the map. There are various reasons for a mismatch, such as incompatible QoS or data types, misconfigured discovery peers, or use of the *Connext* **ignore_* ()** APIs, among other reasons.

The matches are shown at the peer-level only. That is, if you select a Publisher, you will not see matches for the DataWriters that belong to it. Table 3.2 Peer-Level Objects provides more information on what matches are shown for selected entities.

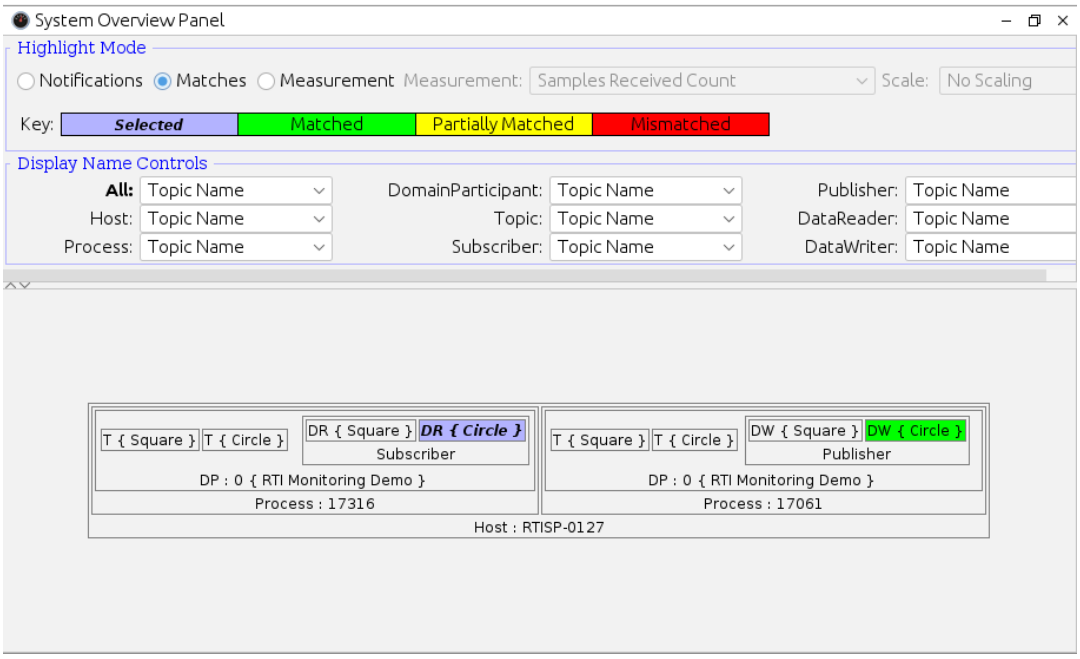
Table 3.2 Peer-Level Objects

If you select this type of object ...	You will see matching information for ...
Host	Hosts
DomainParticipant	DomainParticipants
Publisher	Subscribers
Subscriber	Publishers
DataWriter	DataReaders
DataReader	DataWriters
Topic	Topics

If an entity is highlighted as 'Partially Matched,' *some* (not all) of its child entities are not matched (such as a Publisher that has one matched DataWriter and one or more unmatched DataWriters.)

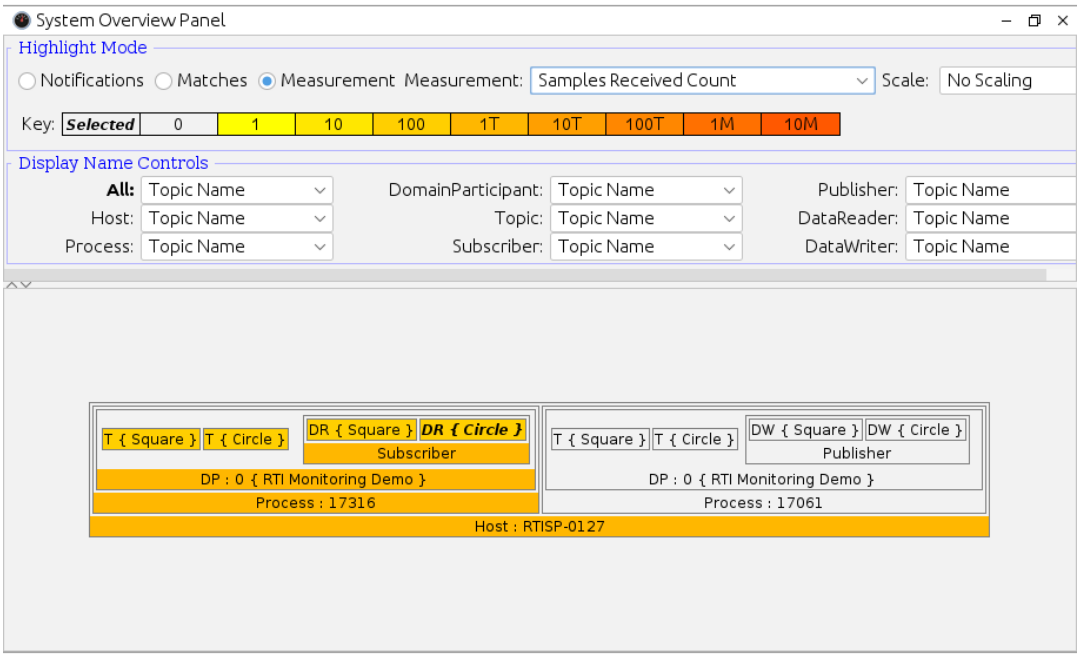
The **-matchRefreshPeriodSeconds** command-line option controls how often the matching information is refreshed (see Table 2.1 Command-line Options for details).

Figure 3.14: System Overview Panel (Matches Option)



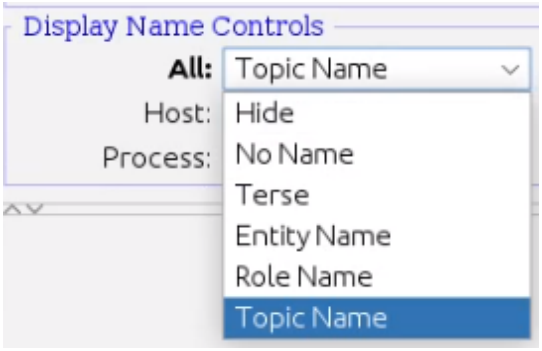
If the **Measurement** option is selected, the **Measurement** and **Scale** drop-down menus are enabled. **Measurement** allows you to select which data value to display in the map. **Scale** allows you to control the scaling factor of the data value.

Figure 3.15: System Overview Panel (Measurement Option)



Move the mouse over an entity in the map will show you the details of that entity for the selected highlight mode.

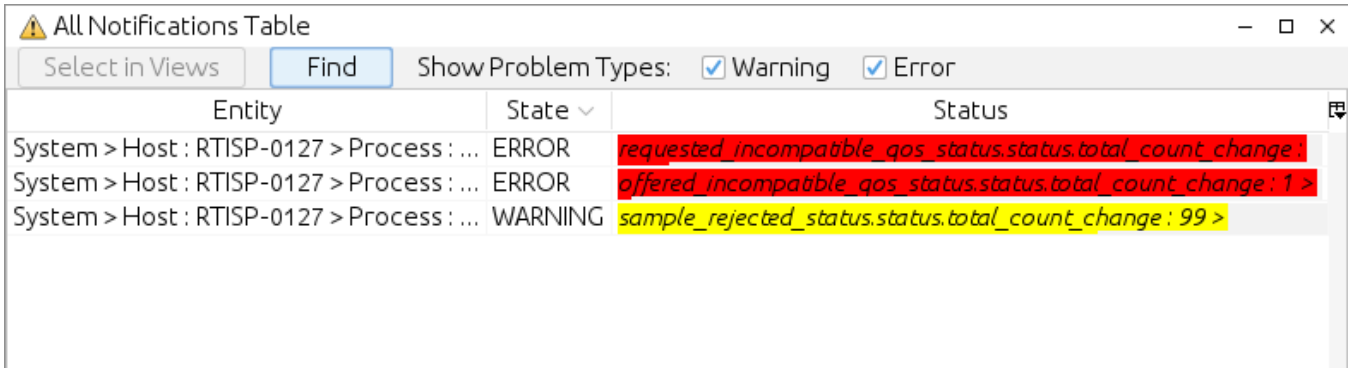
The **Display Name Controls** simply control how the items in the map are labeled (or whether they are hidden). Each entity types can be hidden from the map, appear with no name label, be labeled tersely (with just an abbreviation for the entity type, such as T for a Topic), or include more information, such as Entity Name, Role Name, or Topic Name.



3.4.2 All Notifications Table

The All Notifications Table shows you all the current errors and warnings for the entire system (not just the currently selected entity).

Figure 3.16: All Notifications Table



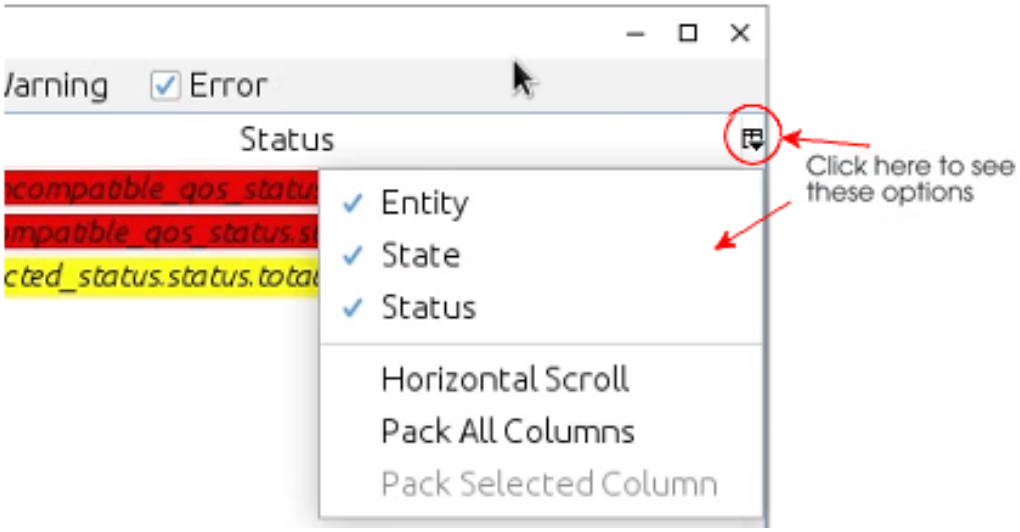
This panel has filters to include or exclude warnings/errors. Errors are shown in red. Warnings are shown in yellow.

If a row is selected in the table, the **Select in Views** button selects the entity in tree views on the left.

The **Find** button is useful for searching through a large table. (This is a simple string search, so you must use the exact same form as displayed in the table.)

Clicking on a column heading will sort the table by the values in that column. Clicking it again will sort in the opposite order.

The  button just above the vertical scroll bar allows you to choose which columns appear in the table. It also has options to pack (resize) columns and enable a horizontal scroll bar. (Note: to enable the 'Pack Selected Column' option, select a cell in the top row.)



You can change the order of the columns by simply dragging them to a new place in the table.

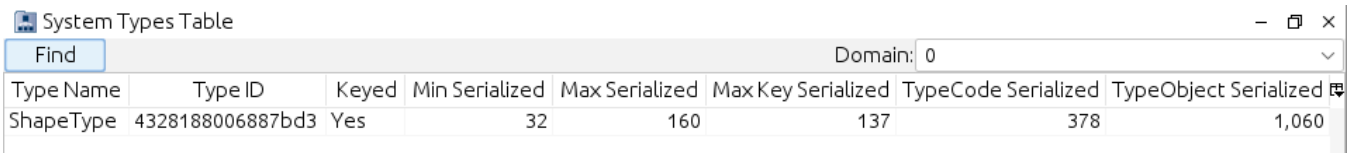
Clearing Notifications

To clear the notifications, select the  button from the toolbar or **Actions, Clear All Notifications** from the menu.

3.4.3 System Types Table

The System Types table displays all the known data types in the selected domain.

Figure 3.17: System Types Table



The **Domain** drop-down menu includes a list of all the joined domain IDs for you to select.

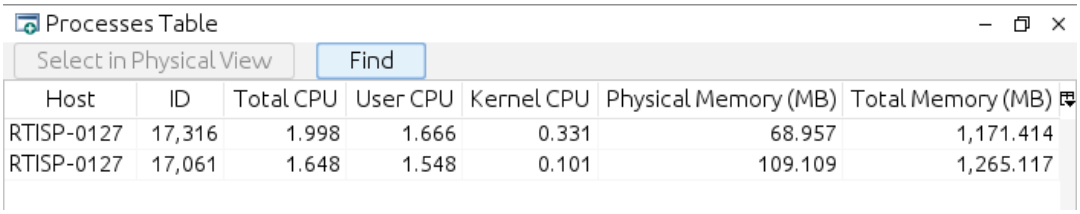
Like the All Notifications table, this table also has a  button (above the vertical scroll bar) to control the columns that appear in the table. You can also sort the table based on any of the columns by clicking the column heading.

The **Find** button is useful for searching through a large table. (This is a simple string search, so you must use the exact same form as displayed in the table.)

3.4.4 Processes Table

The Processes table displays memory and CPU information for all the processes in the system.

Figure 3.18: Processes Table



These values are valid only if the host is a Linux or Windows system.

For multi-core machines, CPU usage can be greater than 1.

Like the All Notifications Table, this table also has a  button (above the vertical scroll bar) to control the columns that appear in the table. You can also sort the table based on any of the columns by clicking the column heading.

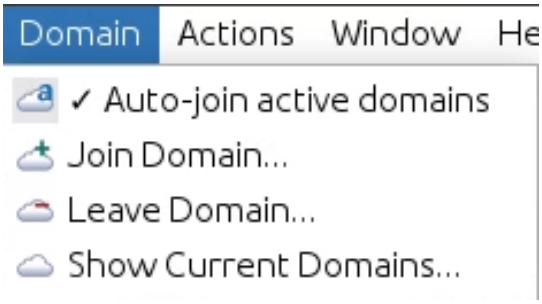
For example, you can quickly sort by Total CPU to see which process is using the most. When the process row is selected in the table, you can use the **Select in Physical View** button to see where this process is within the physical tree.

The **Find** button is useful for searching through a large table. (This is a simple string search, so you must use the exact same form as displayed in the table.)

3.5 Joining and Leaving Domains

By default, *Monitor* will automatically discover and join all active DDS domains.

To enable/disable this feature, use the button or select **Domain, Auto-join active domains** from the menu. The enabled/disabled status for this feature is preserved between launches of *Monitor*.



If the Auto-join feature is enabled: First, *Monitor* will join domain 0. Then all other active domains will be discovered through their announcements in domain 0. Each discovered domain will be joined if it has not already been joined.

To see the currently joined domains, select **Domain, Show Current Domains...** from the menu.

You can manually join and leave specific domains by using the and buttons on the toolbar or the commands in the **Domain** menu. Note: When auto-join is enabled, *Monitor* cannot leave domain 0.

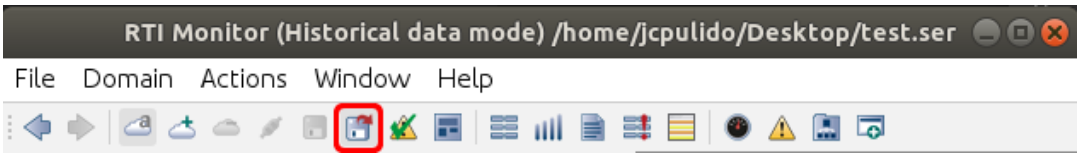
If you leave a domain, it will not be rejoined automatically unless you disable and then re-enable the Auto-join feature.

3.6 Saving and Loading Data

Monitor can work with live data or data that's been saved to a file. To save monitoring data, use the button on the toolbar or select **File, Save Data...** from the menu. The file will be saved in a Serialized Java Objects format (**.ser**).

To load a data file, use the button on the toolbar or select **File, Load Data...** from the menu.

When *Monitor* is working with saved data, you will see **(Historical data mode)** in the title bar, followed by the filename. For example:



Monitor will save notifications, descriptions, and statistics (for DomainParticipants, DataReaders, and DataWriters), up to the history depth or notification history depth for every object being monitored.

Notes:

- While viewing saved data, you will not see built-in topic data or the IDL representation of the data type in the Description panel. This information is not saved in the data file.
- You cannot save data while using a loaded data file.
- After viewing saved data, to return to live data you must reconnect to the domain(s) that you want to monitor by joining the domain (see 3.3.5 Distributed Logger Panel on page 21).

3.7 Connecting and Disconnecting the Display

To stop *Monitor* from updating the display (while still receiving data), select the button on the toolbar or **Actions, Disconnect Display** from the menu.

To resume display updates, select the button on the toolbar or **Actions, Connect Display** from the menu.

Note: Data samples may be lost at the *Connext* level while *Monitor's* display is disconnected because the History QoS is configured to only keep the last few samples.

3.8 Changing Transport Settings in the Configuration File

The QoS profile used by *Monitor* is in **<NDDSHOME>/resource/xml/RTI_MONITOR_QOS_PROFILES.xml**. You can edit this file to adjust the QoS to fit your system's needs. The typical use case is to adjust the transport settings so that they align with the other applications in the system, as these are critical for communication. However, changing any other settings in this file may result in unpredictable behavior and is not supported.

Generally, the configuration file is editable on the system. There are certain circumstances where it cannot be updated, such as on Windows when *Monitor* is installed in the "Program Files" directory. If this is the case, open your text editor with 'administrative' permissions before opening the QoS file.

Appendix A Determining Host Name and Process ID

Monitor uses the familiar concepts of host and process to organize information displayed in the user interface. Processes determined to be running on the same host are grouped together to make it easier to find information. The host name and the process ID are derived from information available to *Monitor* at run time.

Internally, *Monitor* receives Discovery information which is primarily composed of data regarding DomainParticipants, DataReaders, and DataWriters. As part of the DomainParticipant information there are system properties that provide *Monitor* with the host name and process ID.

However, these properties can be disabled through QoS configuration. In this case, because the necessary information isn't in the system properties, *Monitor* will use 127.0.0.1 for the IP address (and lookup the name from that address) and zero for the process ID. This is an attempt to more accurately reflect what information is known. If you see either of these values in the user interface, please reconfigure your QoS settings to allow the system properties to reach *Monitor*.