RTI Recording Service

Release Notes

Version 5.2.3

Your systems. Working as one.

Trademarks

Copy and Use Restrictions

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

Release Notes

1 System Requirements

RTI® Recording Service is supported on these platforms:

- ❏ Linux® Platforms:
  - CentOS 5.4, 5.5, 6.0, 6.2 - 6.4, 7.0
  - Raspbian Wheezy 7.0
  - Red Hat® Enterprise Linux 5.0-5.2, 5.4, 5.5, 6.0 - 6.5, 6.7, 7.0
  - SUSE® Linux Enterprise Server 11 SP2, SP3
  - Ubuntu® Server 12.04 LTS, Ubuntu Server 14
- ❏ All Mac® OS X platforms listed in the *RTI Connext DDS Core Libraries Release Notes* with the same version number.
- ❏ All Windows® platforms listed in the *RTI Connext DDS Core Libraries Release Notes* with the same version number.

For more information on these platforms, see the *RTI Connext DDS Core Libraries Release Notes* and *Platform Notes*.

Recording Service is also supported on the QNX 6.6 platform (i86QNX6.6qcc_cpp4.7.3). This is a target platform for which RTI offers custom support. For more information, please contact your local RTI representative or email **sales@rti.com**.

2 Compatibility with Other RTI Products

The *Record* tool supports the standard *Connext DDS* transports (UDPv4, UDPv6, and shared memory), as well as the *RTI Secure WAN Transport* plugins.

Recording Service 5.1.0 and higher is compatible with *RTI Connext DDS* 5.0.0 and higher, as well as *RTI Data Distribution Service* 4.5[b-f], 4.4d, 4.4b, 4.3e and 4.2e[1] except as noted below.

- ❏ *Recording Service* 5.1.0 and higher is not backwards compatible with databases recorded with previous releases of *RTI Recording Service* (5.0 and lower). This applies to all the tools in *Recording Service*: *Record*, *Replay*, *Converter*, and *Recording Console*.

[1] To support compatibility with 4.2e, please see the *RTI Core Libraries and Utilities Release Notes*.

- ❏ *Recording Service* is not compatible with applications built with *RTI Data Distribution Service* 4.5e and earlier releases when communicating over shared memory. For more information, please see the Transport Compatibility section in the *RTI Core Libraries and Utilities Release Notes*.
- ❏ In *Connext DDS* 5.1.0, the default **message_size_max** for the UDPv4, UDPv6, TCP, Secure WAN, and shared-memory transports changed to provide better out-of-the-box performance. *Recording Service* 5.1.0 also uses the new default value for **message_size_max**. Consequently, *Recording Service* 5.1.0 and higher is not out-of-the-box compatible with applications running older versions of *Connext DDS* or *RTI Data Distribution Servic*e. Please see the *RTI Core Libraries and Utiltities Release Notes* for instructions on how to resolve this compatibility issue with older *Connext* and *RTI Data Distribution Service* applications.
- ❏ Some changes were made to the *Record* and *Replay* IDL files starting in *Connext DDS* 5.1.0:
  - In the *Record* IDL file (**resource/idl/rtirecord.idl**):

    The topic names for administration were changed for better alignment with other RTI components:

    - The command request topic name is now rti/recorder/administration/command_request.
    - The command request/status topic name is now rti/recorder/administration/command_response.
    - Some type names and enumeration values in the IDL have been changed so they are more representative.
  - In the *Replay* IDL file (**resource/idl/rtireplay.idl**):

    Topic names did not have string constants in the IDL file that you could use. These names have been added to the IDL: COMMAND_REQUEST_TOPIC_NAME and COMMAND_RESPONSE_TOPIC_NAME.
- ❏ The types of the remote administration and monitoring topics in 5.1.0 are not compatible with 5.0.0. Therefore:
  - The 5.0.0 *Record* and *Replay* shells, *Admin Console* 5.0.0 and *Connext* 5.0.0 user applications performing monitoring/administration are not compatible with *Recording Service* 5.1.0 and higher.
  - The *Record* and *Replay* shells, *Admin Console,* and *Connext* 5.1.0 and higher user applications performing monitoring/administration are not compatible with *Recording Service* 5.0.0.

2.1 Command-Line Options Compatibility

Starting with 5.1.0, the *Replay* tool's command-line parameter, **-forceXmlTypes**, is deprecated. The XML type configuration will always be used if it is available.

For details on how the *Replay* tool selects a type definition for a Topic, see Section 7.10 in the *Recording Service User's Manual*.

3 What's New in 5.2.3

3.1 New Platforms

This release supports these new platforms:

- ❑ CentOS 7.0
- ❑ OS X 10.11
- ❑ Red Hat Enterprise Linux 6.7
- ❑ Windows 10, Windows Server 2012 R2

For details on these platforms, see the *RTI Core Libraries and Utiltities Platform Notes.*

3.2 Recorder now Stores Type-Object as well as Type-Code

Previous versions of *Recording Service* worked only with type-codes in serialized format when using types via discovery. Now *Recording Service* will first look for a type-object, then look for a type-code. The type-object will be serialized and stored in the DCPSPublication table.

This allows Recorder, Replay and Converter to work with mutable types and types containing optional members without the need to provide type information via XML, like in previous releases.

3.3 Ability to Replay Multiple Files

The *Record* tool can split recording in several files, whereas the *Replay* tool previously could only replay from a single file. Starting with this release, the *Replay* tool now provides a <fileset> tag inside the <replay_database> tag. You can use this tag to specify a file prefix, a set number, and start and end segment numbers (just like for the *Record* tool).

This allows you to replay a subset of the recorded files from the same recording session. The *Replay* tool will seamlessly switch from one recorded file to the next.

4 What's Fixed in 5.2.3

4.1 Replay Failed when Replaying Built-in Type DDS::Octets Recorded in Deserialized Format

When the DDS::Octets built-in type was recorded in deserialized format, the *Replay* tool failed to replay it and reported this error:

```
DDS_DynamicData_set_octet_seq:type mismatch for field <no name> (id=1)
PLAYBACKTopic_copyFromFields:!topic:[VideoJANUSMulti1] error serializing
field 'value'- DynamicData set failure
```

This problem has been resolved.

[RTI Issue ID RECORD-247]

4.2 Wrong Type-Code and Type-Code Length Recorded for Topics

When recording a topic that shared type-information via discovery, if Admin Console was started, the administration topics' types were recorded in the DCPSPublication table with the wrong type-code and type-code length, that of the first topic being recorded. This problem has been resolved.

[RTI Issue ID RECORD-653]

4.3 Replay Published Samples Marked as Invalid

The **SampleInfo_valid_data** value indicates if a recorded sample is valid. The *Replay* tool did not look at this field and therefore may have published some invalid samples. This problem has been resolved.

[RTI Issue ID RECORD-743]

4.4 Converter Failed To Export Data When Several Types Present in Database

Converter may have failed to export data when the database file contained topics with different type definitions. In this case, you may have seen the following (or similar) errors:

```
Converting to XML
DDS_DynamicData_set_string:type mismatch for field a (id=1)
exception:[DRT_DynamicData_set_data_value@3797]:Internal process to set
data value to DDS_DynamicData instance.
exception:[DRT_DynamicData_set_field_value@3939]:Failed to set a value in a
field.
exception:[RTIConverterModel_copyFromFields@814]:DynamicData set failure
exception:[RTIConverterModel_userDataTableCallback@1300]:Failed to serial-
ize user-data fields
exception:[RTIConverterModel_convert@3584]:Failed to convert table: Exam-
ple one
exception:[main@568]:failed to convert rti_recorder_default.dat_3_0 to XML
RTI Recording Service - Convert: Finished
```

This problem has been resolved.

[RTI Issue ID RECORD-745]

4.5 Replay always Published Serialized Samples with all Empty Fields for Optional Members

When *Recording Service* stored data in a serialized format containing optional members, the *Replay* tool did not properly reconstruct the samples when publishing them and the optional members always appeared as empty even if they weren't. This was due to the serialization format *Recording Service* used to store the sample. This has problem has been resolved. Compatibility of *Replay* with old 5.2 recordings has been preserved.

[RTI Issue ID RECORD-754]

5 Previous Releases

This section includes:

- What’s New in 5.2.0 (Section 5.1)
- What’s Fixed in 5.2.0 (Section 5.2)

5.1 What's New in 5.2.0

5.1.1 New Platforms

This release added support for these platforms:

❏ Mac OS X 10.8, 10.10

❏ Raspbian Wheezy 7.0

❏ Red Hat Enterprise Linux 6.5 and 7.0

❏ SUSE 11 SP3

❏ Ubuntu 14

5.1.2 Removed Platforms

These platforms are no longer supported:

❏ Fedora 12

❏ Windows platforms using Visual Studio 2005

5.1.3 Record and Replay Now Compatible with RTI Limited Bandwidth Endpoint Discovery Plug-in

The *Record* and *Replay* tools have been updated to link dynamically against the *RTI Connext DDS Core Libraries*. This allows these tools to load and use the *RTI Limited Bandwidth Endpoint Discovery (LBED) Plug-in*.

In order for your application to discover the *Record* or *Replay* tools that are using the LBED plug-in, these tools' configurations must set the *DomainParticipant's* **participant_name**. There is no restriction on what string is used for the **participant_name**, but if you intend to use the *Record* and *Replay* tools with *RTI Administration Console*, the **participant_name** for the *Record* tool must start with the text "**RTI Recorder:** " and the participant name for the *Replay* tool must start with "**RTI Replay:** " (notice there is a space after the colon.) For example:

```
<participant_qos>
    ...
    <participant_name>
        <name>RTI Recorder: MyParticipant</name>
    </participant_name>
</participant_qos>
```

5.1.4 Extended Precision in Converter when Exporting Doubles and Floats

The minimum precision when exporting floating-point data (double and float fields) has been extended from 6 digits to 10 digits.

5.1.5 Record Tool Displays Recording File Name

The *Record* tool now displays the name of the file in which it is recording. For example:

```
RTI Recorder started
Recording to file rti_recorder_default.dat_1_0
...
```

5.1.6 Increased Error Level for 'Incompatible QoS' Error Messages

If an incompatible-QoS error prevented Recorder from recording data, an error message was shown with 'INFO' level. Now that message will be shown with a higher error level (WARNING).

5.2 What’s Fixed in 5.2.0

5.2.1 Recording Console Failed to Run from RTI Launcher on 64-bit Red Hat 6 Platform

When using *RTI Launcher* to start *Recording Console* on a 64-bit Red Hat 6 platform, the following error would occur:

```
<path to rti install>/RTI_Recording_Service_5.0.0/scripts/rtirecordingcon-
sole: line 16: echo: write error: Bad file descriptor
```

This problem has been resolved; *Recording Console* now starts correctly from *RTI Launcher*.

[RTI Issue ID RECORD-615]

5.2.2 Recorder and Replay could not Use Built-in QoS Profiles

Record and *Replay* couldn't use the built-in QoS Profiles in previous versions. Both tools are now integrated with these profiles and can refer to them via any of the QoS loading methods.

[RTI Issue ID RECORD-637]

5.2.3 Converter Allocated Unnecessary Memory for Types in DCPSPublication Table

Converter allocated unnecessary memory by duplicating type information already extracted from the DCPSPublication table. This may have lead to high memory consumption and memory failures. This problem has been resolved.

[RTI Issue ID RECORD-648]

5.2.4 Replay Checked Maximum Number of Columns for Serialized-Format Recordings

When attempting to replay a file that was recorded in serialized mode, the maximum number of columns should not affect the replay. However, *Replay* was performing the deserialized precalculations even for tables that were recorded in serialized format. You would see the following errors and the program would exit:

```
RTI Replay 5.1.0 initializing ...
PLAYBACKTopic_getMembersFromTypeCodeCallback:!envelope[4999]column limit
for topic exceeded
PLAYBACKTopic_initialize:!parse typeCodeExample data$RecordAll$domain0
PLAYBACKSession_initialize:!create topics, no data for any topic
PLAYBACKSession_new:!init playback session
PLAYBACKDatabase_createSession:!create playback session
PLAYBACKDatabase_initialize:!create playback session
PLAYBACKService_initialize:!init PLAYBACKDatabase child objects
PLAYBACKService_new:!init replay_service
main:!create replay service
Terminating service...
```

This problem has been resolved. *Replay* will no longer perform deserialized-format precalculations on serialized tables.

[RTI Issue ID RECORD-652]

6 Known Issues

6.1 Issues Related to Replay Tool

❑ The *Replay* tool currently does not support the following XML configuration modes:

- <replay_service> <auto_exit> (has no effect)
- <replay_topic> <output> <keyed> (has no effect)
- <time_control> <start_mode> MATCHED or LOOP modes
- <time_control> <rate> AS_FAST_AS_POSSIBLE (except for session level)
- <topic_time_control> <start_mode> MATCHED mode

❑ Limitations with the *Replay* tool’s shell commands:

- The **step** command is functional for session and topic entities only (not service or database)
- The **rate** command is functional for topic entities only

❑ Performance and indexing with the *Replay* tool:

The *Replay* tool replays stored samples in the same order in which they were received, using SQLite indexes to retrieve the samples in sorted order. SQLite automatically builds indexes when opening an SQLite table for sorted access; for large tables the process of building the index may take some time. To improve *initialization* performance, the *Replay* tool attempts to create and store indexes, rather than depend upon automatic indexing, for the tables which it will be replaying, saving initialization time on subsequent replays.

The *Replay* tool's ability to store indices is controlled by the <readonly> parameter of the <replay_database>. Setting <readonly> to true prevents *Replay* from storing indices for a table; in this mode. the *Replay* tool will display a message during initialization for each table opened stating that it was unable to store the table index. Setting <readonly> to false (the default) will allow the *Replay* tool to write the table indices to the database.

The *Replay* tool’s performance is not affected by this option; it will use the fastest means of retrieving samples in either case. But setting the <readonly> option to false may help improve the tool’s *initialization* performance.

❑ When loading a large file for playback, please be aware that this operation may take some time.

❑ If you load the configuration file, **examples/replay_simple_config.xml**, and select the **fast_replay** configuration profile while using your own recorded data file (instead of the example recording from RTI), the *Replay* service will exit and log a message regarding ‘no match in the recording for A_Topic.’

❑ The *Record* and *Replay* Shells are not completely compatible with standard input piping of commands.

❑ For *RTI Admin Console* to work properly with the *Replay* tool, do not use the XML <name> tag under <administration>. *Admin Console* will not recognize the replay service and will not be able to administer it. This will be addressed in a future release. [RTI Issue ID BIGPINE-429]

6.2 Issues Related to Recording Console

❑ In *Recording Console*, when changing playback speed, or skipping to another playback location, occasionally playback will appear stuck (it is actually paused). The workaround is to click the Pause button twice.

❑ *Recording Console* may fail to shut down gracefully after stepping through to the end of a recording. If a recording is paused and then stepped through to the end, the *Replay* service may not shut down properly. In this case, *Recording Console* displays an error that the service stopped unexpectedly. [RTI Issue ID RECORD-135]

- ❏ Interaction between *Recording Console* and *Admin Console*

  This issue only applies if you are using *Recording Console* and *RTI Admin Console* at the same time, and you have configured *Admin Console* to join domain ID 99. In this scenario, do not use *Admin Console* to pause or disable any *Recording Console* services (their names begin with "RTI-Recorder-" or "RTI-Replay-"). Doing so may cause an error in *Recording Console*. [RTI Issue ID BIGPINE-795]

- ❏ *Recording Console* will not reflect stopped status if recording is stopped by another tool.

  When recording data with *Recording Console*, *RTI Admin Console* can send a command to stop the recording. In this case, recording will stop but *Recording Console* won't reflect the stopped status in any way; it will appear that recording is still in progress, although the file won't grow in size.

  Pause commands work fine and are reflected by both sides, *Recording Console* and *Admin Console*.

  [RTI Issue ID RECORD-253]

- ❏ Welcome screen may appear blank on some platforms

  The welcome screen may appear blank if the operating system does not have a web browser that is compatible with Eclipse. [RTI Issue ID DIABLO-538]

6.3 Issues Related to Converter

- ❏ When using *Converter* on a recording created with *Recording Console*, you may see a warning related to internal topics used by the *Console*:

  ```
  exception:[RTIConverterModelPublisherCallback@2293]:Failed to
  create type com_rti_tools_remotectx
  ```

  You can safely ignore the warning—the conversion results *are* valid.

- ❏ *Converter* (**rtirecconv**) cannot convert tables with only a subset of the data. Most SQLite database viewer tools include functionality to export the database contents to other formats such as XML or CSV. In cases where the database was recorded with filtered fields, it's possible to use one of these tools to export the data.

- ❏ In files recorded on Windows systems, the recorded timestamp is the number of microseconds since the device was booted, not since January 1, 1970. Therefore the **-time gmt** option to *Converter* (**rtirecconv**) will not show the correct time.

6.4 Other Issues

- ❏ When you record a database using the PRAGMA feature (**<sqlite_pragmas>** in the **<recorder_database>** settings), the resulting databases may be incompatible with *Recording Console*. This is due to a third-party incompatibility. The following exception will be thrown:

  ```
  java.sql.SQLException: file is encrypted or is not a database
  ```

  To replay the database, use the *Replay* tool.

  [RTI Issue ID RECORD-574]

- ❏ Recording and/or replaying mutable types requires the type definition to be provided via XML configuration using the <type_config> tag. If the type definition is not provided via XML, the *Record* tool will display the following error messages:

- When recording in deserialized mode:

  ```
  Failed to get valid typecode information for Publisher.
  Recorder cannot confirm that the entity publishes a
  supported type.
  ```

- When recording in serialized mode:

  ```
  DDS_DynamicData_from_stream:ERROR:Bad
  parameter:encapsulation_kind of stream
  ```

❑ To record a data type that has more than 5,050 primitive types, you must set the **deserialize_mode** property to RTIDDS_DESERIALIZEMODE_NEVER. Otherwise, you will see the following error message and recording will fail:

```
"exception:[RTIDRTUserDataTable_update@610]:too many SQL variables"
```

[RTI Issue ID RECORD-38]

❑ The DynamicData API does not support out-of-order assignment of members with a length greater than 65,535 bytes. In this situation, the following error is reported:

```
sparsely stored member exceeds 65535 bytes
```

For example:

```
struct MyStruct {
    string<131072> m1;
    string<131072> m2;
};
```

With the above type, the following sequence of operations will fail because **m2** is assigned before **m1** and has a length greater than 65535 characters.

```
str = DDS_String_alloc(131072);
memset(str, 'x', 131072);
str[131071]= 0;
DDS_DynamicData_set_string(
    data,"m2", DDS_DYNAMIC_DATA_MEMBER_ID_UNSPECIFIED, str);
DDS_DynamicData_set_string(
    data,"m1", DDS_DYNAMIC_DATA_MEMBER_ID_UNSPECIFIED, str);
```

If the member **m1** is assigned before **m2,** the sequence of operations will succeed.

[RTI Issue ID CORE-3791]

❑ RTI does not recommend using files that are mounted over NFS to store recorded data. *Recording Service* uses file-locking, which has known issues working over NFS. If file-locking is not working, *Recording Service* will hang. In particular, this problem may appear on Yellow Dog Linux systems.

❑ Leading and trailing spaces in a Topic Name are ignored. However, spaces within the string are allowed. For example, " My Topic " will be treated as "My Topic".

❑ Fully qualified field names in struct's cannot be longer than 1,024 characters.

❑ Sequence and array indices cannot be used in Topic or Field expressions.

❑ *Recording Service* cannot communicate with DataReaders or DataWriters of Topics with a data type that includes bit fields. You may see the following message, but *Recording Service* will continue to work normally otherwise:

```
DDS_DynamicDataTypeSupport_initialize:type not supported
       (bitfield member)
```

[RTI Issue ID CORE-3949]

❑ *Recording Service* and *Converter* cannot deserialize bit fields. If this type is used, the deserialize mode must be RTIDDS_DESERIALIZEMODE_NEVER.

❑ If the *Connext DDS* application being recorded has a keyed data-type and **DataWriter-ProtocolQosPolicy.disable_inline_keyhash** is set to TRUE (not the default), *Recording Service* may misinterpret samples as being from the wrong instance.

❑ If you start an instance of the *Record* tool using command-line options (not a configuration file), then sending a new configuration to that instance of the *Record* tool using the remote shell will not work.

❑ When <time_mode> is set to TOPIC_RELATIVE, the first sample in a recording is not sent right away when replay starts. [RTI Issue ID RECORD-133].

❑ There is a known limitation when recording data in serialized format in environments where multiple versions of a type are published. If the writers do not publish their type-code information, the *Record* tool may store samples from unwanted versions. [RTI Issue ID RECORD-346]

❑ On Windows 8 systems, be aware of a limitation in the OS regarding the write permissions in some folders. Even if you are using an administrator account, some folders (such as C: or "Program Files") cannot be used to store user data. If you try to create a recording database there, Windows 8 will automatically create it in a virtual storage unit (usually found under **C:\Users\<*user_name*>\AppData\Local\VisualStore**). This folder might be hidden. [RTI Issue ID RECORD-525]