RTI Routing Service

for

RTI Data Distribution Service

Release Notes

Version 2.0.1

Printed in U.S.A. First printing.
June 2010.

Trademarks

Copy and Use Restrictions

Technical Support

Real-Time Innovations, Inc.
385 Moffett Park Drive
Sunnyvale, CA 94089
Phone: (408) 990-7444
Email: support@rti.com
Website: http://www.rti.com/support

Contents

Release Notes

1 Supported Platforms

RTI® Routing Service is currently supported on these platforms:

Table 1.0 **Supported Platforms**

Platform	Operating System	Architecture
Linux®	Red Hat® Enterprise Linux 4.0 (2.6 kernel)	i86Linux2.6gcc3.4.3 x64Linux2.6gcc3.4.5
	Red Hat Enterprise Linux 5.0 (2.6 kernel)	i86Linux2.6gcc4.1.1 x64Linux2.6gcc4.1.1
	Red Hat Enterprise Linux 5.1, 5.2 (2.6 kernel)	i86Linux2.6gcc4.1.2 x64Linux2.6gcc4.1.2
Windows®	Windows 7 Windows Server® 2008 R2 Windows 2003 Windows Vista® Windows XP Professional	i86Win32VS2005 i86Win32VS2008 i86Win32VS2010 x64Win64VS2005 x64Win64VS2008 x64Win64VS2010

RTI Routing Service is built on top of, and intended for use with *RTI Data Distribution Service* 4.5c.

2 Compatibility

2.1 RTI Data Distribution Service Compatibility

With the built-in DDS adapter, this *RTI Routing Service* release can be used to forward and transform data between DDS applications built with *RTI Data Distribution Service* 4.5c, 4.5b, 4.4d, 4.3e, and 4.2e.

2.1.1 RTI Data Distribution Service 4.2e Compatibility

If the applications' data types contain 8-byte or larger primitive types (double, long long, unsigned long long or long double), *RTI Routing Service* will have to be run with the command line option **-use42eAlignment** in order to be compatible with *RTI Data Distribution Service* 4.2e.

If the applications use large data, *RTI Routing Service* must be configured with the following properties set to 1 in order to be compatible with *RTI Data Distribution Service* 4.2e:

- **dds.data_writer.protocol.use_43_large_data_format**
- **dds.data_reader.protocol.use_43_large_data_format**

2.1.2 RTI Data Distribution Service 4.3e Compatibility

If the applications use large data, *RTI Routing Service* must be configured with the following properties set to 1 in order to be compatible with *RTI Data Distribution Service* 4.3e.

- **dds.data_writer.protocol.use_43_large_data_format**
- **dds.data_reader.protocol.use_43_large_data_format**

2.1.3 Mixing Different RTI Data Distribution Service Versions

This *RTI Routing Service* version can be used simultaneously with applications built using *RTI Data Distribution Service* 4.5c, 4.5b, 4.4d, 4.3e and 4.2e, unless special configuration is required as described above.

2.2 Command-Line Options Compatibility

Starting with *RTI Routing Service* 1.1.0, the command-line parameter **-srvName** has been replaced with **-cfgName** (to select a configuration) and **-appName** (to name the service execution). In previous *RTI Routing Service* versions, the **-srvName** parameter was not required. However, in this version the equivalent parameter **-cfgName** is required.

The allowed values for the **-verbosity** command-line option have changed. The new **-verbosity** option coalesces the old **-verbosity** and **-ddsVerbosity** options into a single parameter.

For additional information about the new command-line options, see Chapter 3 in the *RTI Routing Service Getting Started Guide*.

2.3 XML Compatibility

- Starting with *RTI Routing Service* 1.1.0, the attribute "name" in the **<routing_service>** tag is now required. Old XML files without the name attribute will not be parsed by *RTI Routing Service* 1.1.0 and higher.
- Starting with *RTI Routing Service* 2.0.0, the way to register and configure transformations in the configuration file has changed:
  - The tag <transformation_class_library> has been replaced with <transformation_library>
  - The tag <transformation_class> has been replaced with <transformation_plugin>.
  - The content of these tags has also changed. With the new configuration there is only a single entry point to the library. For example:

```
<transformation_library name="MyTransfLib">
   <transformation_plugin name="MyTransfPlugin">
        <dll>mytransformation</dll>
        <create_function> <!-- Entry point -->
            MyTransfPlugin_create
        </create_function>
   </transformation_plugin>
</transformation_library>
```

  - The configuration of a transformation within <route> is done using properties instead of the <expression> and <parameter> tags. For example:

```
<transformation plugin_name="TransformationLib::Assignment">
   <property>
        <value>
            <element>
                <name>X</name>
                <value>Y</value>
            </element>
```

```
            <element>
                <name>Y</name>
                <value>X</value>
            </element>
        </value>
    </property>
</transformation>
```

- The configuration of the assignment transformation distributed with *RTI Routing Service* is now done with properties. For example:

  Before this release:

```
<transformation className="TransformationLib::Assignment">
    <expression></expression>
    <parameter>position.x=position.y</parameter>
    <parameter>x=10</parameter>
</transformation>
```

  In this release:

```
<transformation plugin_name="TransformationLib::Assignment">
    <property>
        <value>
            <element>
                <name>position.x</name>
                <value>position.y</name>
            </element>
            <element>
                <name>x</name>
                <value>10</name>
            </element>
        </value>
    </property>
</transformation>
```

2.4 Transformation API

The transformation API of this *RTI Routing Service* release is not compatible with the API of previous releases.

The new API follows the same model as the adapter API (see Section 5.1.1) where we introduced the concept of Plugin as a C structure that contains all the function pointers that implement the interface.

The registration of a transformation plugin with the new model requires a single entry-point to the shared library; the entry-point is a function that creates the Plugin structure which contains the implementation. For example:

```
<transformation_library name="MyTransfLib">
    <transformation_plugin name="MyTransfPlugin">
        <dll>mytransformation</dll>
        <create_function> <!-- Entry point -->
            MyTransfPlugin_create
        </create_function>
    </transformation_plugin>
</transformation_library>
```

The following table shows how deprecated functions map to the new API.

Previous API (rti_routingservice.h)	2.0.1 API (routingservice_ transformation.h)	Comments
RTITransformationClass_ loadFnc()	**RTI_RoutingServiceTransformation Plugin_CreateFcn()**	This is the entry-point function.
RTITransformationClass_ unloadFnc()	Function declaration: **RTI_RoutingServiceTransformation Plugin_DeleteFcn()** Member in Plugin struct: **transformation_plugin_delete**	
RTITransformationClass_ createFnc()	Function declaration: **RTI_RoutingServiceTransformation Plugin_CreateTransformationFcn()** Member in Plugin struct: **transformation_plugin_create_ transformation**	
RTITransformationClass_ deleteFnc()	Function declaration: **RTI_RoutingServiceTransformation Plugin_DeleteTransformationFcn()** Member in Plugin struct: **transformation_plugin_delete_ transformation**	

Previous API (rti_routingservice.h)	2.0.1 API (routingservice_ transformation.h)	Comments
RTITransformationClass_ modifyFnc()	Function declaration: **RTI_RoutingServiceTransformation_ UpdateFcn()** Member in Plugin struct: **transformation_update**	
RTITransformationClass_ transformFnc()	Function declaration: **RTI_RoutingServiceTransformation_ TransformFcn()** Member in Plugin struct: **transformation_transform**	In the new API, the transform function accept multiple samples. In addition, the output samples must be created by the transformation instead of being passed in by *RTI Routing Service*.
(none)	Function declaration: **RTI_RoutingServiceTransformation_ ReturnLoanFcn()** Member in Plugin structure: **transformation_return_loan**	This function is used to return the loan on the samples returned by the transform function.

3 What's New in Release 2.0.1

This section describes what's new since the previous release, 2.0.0.

3.1 New Example Adapters in C and Java

To support integration with non-DDS systems, *RTI Routing Service* provides an API to develop custom adapters. There is an API for C adapters and an API for Java adapters.

An *adapter* is a pluggable component that allows *RTI Routing Service* to consume and produce data for different data domains. By default, *RTI Routing Service* is distributed with a built-in DDS adapter.

To serve as examples and to provide a base for building your own custom adapters, the source code for three other adapters is provided with this release:

- A file adapter (implemented in C)

- ❑ A TCP socket adapter (implemented in C)
- ❑ A JMS adapter (implemented in Java)

For more details about adapters, see Chapter 7 in the *RTI Routing Service User's Manual*.

To start compiling and using the distributed adapters, see the *RTI Routing Service Getting Started Guide.*

3.2 Ability to Execute RTI Routing Service Shell Commands from a File

In addition to reading commands from the prompt, the remote shell can read and execute commands from a file. To specify such a file, use the new command-line option, **-cmdName**.

A command file may contain any number of commands, one per line, exactly as they would be entered in the shell prompt.

For convenience, a new command has been introduced to sleep between the execution of other commands: **-sleep <seconds>**.

4 What's Fixed in Release 2.0.1

This section describes bugs that have been fixed since the previous release, 2.0.0.

4.1 Topic Routes May Fail to Start when TypeCode was not propagated

In the previous release, if the matching publications or subscriptions of a topic route did not propagate their typecode and that topic route was configured with the ON_DOMAIN_MATCH creation mode, it would never start, even if the type was defined in the XML file.

This problem has been resolved; now the propagation of the typecode is not strictly required when the type is defined in XML.

[RTI Bug # 13177]

4.2 Field Assignment Transformation did not Support Typedefs

The transformation that assigns fields from one sample to another sample did not support fields of a typedef'd type. This problem has been resolved.

[RTI Bug # 13286]

4.3 Parsing Failure in USER_ROUTING_SERVICE.xml file May Have Lead to Segmentation Fault

If a parsing error was found in USER_ROUTING_SERVICE.xml, *RTI Routing Service* would report warning about the parsing error and continue. This lead to errors during execution that sometimes resulted in a segmentation fault. In this release, parsing errors in USER_ROUTING_SERVICE.xml are considered a failure condition instead of a warning and execution will not continue.

[RTI Bug # 13407]

4.4 Using Multiple <participant_1/2> Tags not Reported as an Error

In the previous release, *RTI Routing Service's* XML parser did not report the use of multiple <participant_1>/<connection_1> or <participant_2>/<connection_2> tags in the same <domain_route> as a configuration error. These errors will now be caught by the parser.

[RTI Bug # 13463]

---

5 Previous Releases

5.1 What's New in Release 2.0.0

5.1.1 Adapter API

To support integration with non-DDS systems, *RTI Routing Service* provides an API to develop custom adapters.

An *adapter* is a pluggable component that allows *RTI Routing Service* to consume and produce data for different data domains. By default, *RTI Routing Service* is distributed with a built-in DDS adapter.

For more details about adapters, see Chapter 7 in the *RTI Routing Service User's Manual*.

5.1.2 Publication with Original Timestamp

There is a new tag called <publish_with_original_timestamp> under <route>, <topic_route>, <auto_route> and <auto_topic_route>. This tag allows *RTI Routing Service* to publish samples with their original timestamp.

For more details, see Chapter 2 in the *RTI Routing Service User's Manual*.

5.2 What's Fixed in Release 2.0.0

5.2.1 RTI Routing Service Crashed if <registered_type_name> Missing in <topic_route>

If the tag <registered_type_name> is missing within <topic_route>, *RTI Routing Service* should report an error. However, in the previous release, it resulted in a crash. This problem has been resolved. [RTI Bug #13245].

5.3 What's New in Release 1.1.0

5.3.1 Command-Line Options

RTI Routing Service 1.1.0 introduces the following command-line options:

- **-cfgName:** Selects a routing service configuration. This option is required except when **-remoteAdministrationDomainId** and **-noAutoStart** are used.
- **-appName:** Assigns a name to the execution of the *RTI Routing Service*.

The new command line options **-cfgName** and **-srvName** replace the now deprecated **-srvName** option of previous releases.

The deprecated option is still functional to preserve backwards compatibility. However, it should not be used as it may not be available in future releases.

The allowed values for the **-verbosity** command-line option have changed. The new **-verbosity** option coalesces the old **-verbosity** and **-ddsVerbosity** options into a single parameter.

For additional information about command-line options see Chapter 3 in the *RTI Routing Service Getting Started Guide*.

5.3.2 New Ways to Load Configuration Files

This *RTI Routing Service* release can load its configuration from one additional location:

- ***<working directory>*/USER_ROUTING_SERVICE.xml**

  This file is loaded automatically if it exists.

For additional information about configuration loading see Section 2.2 in the *RTI Routing Service User's Manual*.

5.4 What's Fixed in Release 1.1.0

5.4.1 RTI Routing Service May Not Discover DDS Entities if Domain Route Disabled, then Enabled

In the previous release, if a Domain Route was disabled and then re-enabled, *RTI Routing Service* did not always discover all the DDS entities. In this situation, topic routes may not have been created. This problem has been resolved. [RTI Bug # 13187]

6 Known Issues

6.1 Short-term Limitations

This issue will be fixed in the next release:

- In the Adapter API, **Connection::get_attributes()** and update operations are not supported.

6.2 Sequences of Transformations in a Route are Not Supported

The tag <transformation_sequence> within a <topic_route> is not supported. In this version only one transformation per route is supported.

6.3 Assignment Data Transformation only Supports Assignment of Primitive Fields Not Part of Arrays or Sequences

The data transformation library distributed with *RTI Routing Service* only supports the assignment of primitive fields (including strings) that are not part of arrays or sequences.

For example:

```
<transformation className="TestTransformationLib::FieldMapping">
  <expression></expression>
  <parameter>position.x=position.y</parameter> <!- This is supported ->
  <parameter>x=y</parameter>                   <!- This is supported ->
  <parameter>x[0]=y[0]</parameters>            <!- This is not supported ->
  <parameter>position=position</parameter>     <!- This is not supported ->
</transformation>
```

For additional details about data transformation, see Chapter 3 in the *RTI Routing Service User's Manual*.

6.4 Asymmetric Communication Scenarios Not Supported with TCP Transport

This version of *RTI Routing Service* does not support the TCP transport modes NDDS_TRANSPORT_TCPV4_TRANSPORT_MODE_CLIENT and NDDS_TRANSPORT_TCPV4_TRANSPORT_MODE_SERVER.

For additional details about asymmetric communication scenarios, see Chapter 6 in the *RTI Routing Service User's Manual*.

7 Available Documentation

RTI Routing Service documentation includes:

- **Release Notes** (RTI_Routing_Service_ReleaseNotes.pdf)—Describes system requirements and compatibility, as well as any version-specific changes and known issues.
- **Getting Started Guide** (RTI_Routing_Service_GettingStarted.pdf)—Highlights the benefits of *RTI Routing Service*. It provides installation and startup instructions, and walks you through several examples so you can quickly see the benefits of using *RTI Routing Service*.
- **User's Manual** (RTI_Routing_Service_UsersManual.pdf)—Describes how to configure *RTI Routing Service* and use it remotely.