TIBCO Rendezvous®

Adapter for TIBCO FTL®

Version 8.8.0 | February 2025

Copyright © 1997-2025. Cloud Software Group, Inc. All Rights Reserved.

Contents

Adapter: Communication between Rendezvous and FTL Applications

You can arrange communication between TIBCO Rendezvous applications and TIBCO FTL applications using a daemon with an adapter module.

For brevity, adapter documentation uses the term *FTL application* to refer to any customer program that uses TIBCO FTL API calls. Similarly, the term *Rendezvous application* refers to any customer program that uses TIBCO Rendezvous API calls. Similar terms refer to other items within customer programs or within the adapter module, such as: *FTL message*, *FTL field*, *FTL format*, *Rendezvous subject*, *Rendezvous network*, *FTL message*, *FTL field*, *FTL side*, and *Rendezvous side*.

Note: An understanding of TIBCO FTL messaging technology is helpful when reading this material.

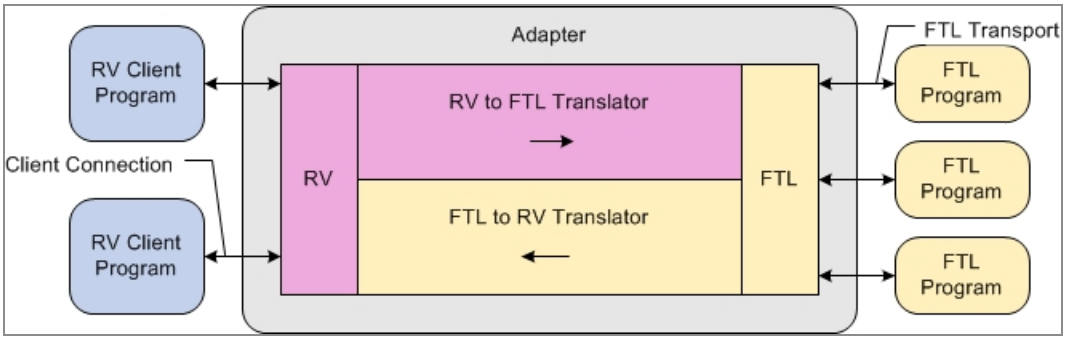
Adapter Overview

The adapter converts messages so TIBCO FTL programs can communicate with existing TIBCO Rendezvous programs. The adapter translates messages in both directions.

To understand the adapter, consider it from three perspectives:

- FTL application perspective
- Rendezvous application perspective
- Adapter perspective

Adapter



FTL Application Perspective

From the perspective of FTL applications (right side of the diagram), the adapter behaves like any other FTL application. It can publish messages and subscribe to messages. It contacts the FTL server to get formats and transports, just as any other application would. It communicates with other FTL applications over FTL transports, as defined in the realm.

Rendezvous Application Perspective

From the perspective of Rendezvous applications (left side of the diagram), the adapter appears like a TIBCO Rendezvous daemon. Indeed, it is a daemon, even though it includes an additional module that provides adapter functionality. This enhanced daemon, `rvda`, replaces an `rvd` instance within the network.

Rendezvous applications connect to the enhanced daemon as they would connect to any daemon. The daemon delivers inbound messages to applications, and accepts outbound messages from applications, as any daemon would.

Adapter Perspsective

Within the adapter module, messages flow in two directions between its Rendezvous side and its FTL side. FTL messages and Rendezvous messages are very different, so the adapter translates these messages according to its configuration.

The central portion of the adapter in the preceding diagram shows the two translators, one for each direction.

Adapter Operation

The adapter has three operating parts: the adapter start phase, the FTL side, and the Rendezvous side.

Adapter Start Phase

When the adapter starts, it does these preparatory steps.

1. The adapter reads the configuration file.
2. The adapter contacts the FTL server to get the realm definition.
3. The adapter validates its configuration.
4. If the start phase succeeds in reaching this point, the adapter starts both its FTL side and its Rendezvous side.

Adapter FTL Side

The adapter's FTL side translates FTL messages into Rendezvous messages and transmits them to Rendezvous clients.

The adapter's FTL side creates one subscriber for each endpoint in the adapter configuration. These subscribers do not include content matchers, so they receive all messages on the endpoint.

Each time a subscriber receives a message with a format specified in one of the configuration’s tags, the adapter attempts to translate it to a Rendezvous message. The tag guides the translation. The adapter first finds the set of FTL fields that translate into the Rendezvous subject or inbox name. Then the adapter translates each remaining field:

- The FTL field name becomes a Rendezvous field name.
- Every FTL field data type either maps unambiguously to a Rendezvous field data type, or raises an error. For details, see Adapter Data Type Mapping: FTL to Rendezvous.
- The FTL field value translates to the Rendezvous field value of the appropriate data type.

After translating all the fields, the adapter transmits the translated Rendezvous message to all its clients that have expressed interest in the translated Rendezvous subject.

Adapter Rendezvous Side

The adapter's Rendezvous side translates Rendezvous messages into FTL messages, and transmits them to FTL clients.

The adapter's Rendezvous side begins by listening for connections from Rendezvous clients. Clients connect to the adapter daemon as they would connect to any TIBCO Rendezvous daemon.

Rendezvous clients send messages. The daemon delivers each message within its Rendezvous network. In addition, the adapter processes each message as follows.

Each time a Rendezvous message matches the subject specification in one of the configuration's tags, the adapter attempts to translate it to an FTL message. The tag guides the translation.

1. The adapter translates and stores the Rendezvous subject.
    - If the Rendezvous subject is a multicast subject name, the adapter stores its elements in FTL fields.
    - If the Rendezvous subject is an inbox name, the adapter maps it to an FTL inbox and stores the FTL inbox in an FTL field.
2. Then the adapter attempts to fill each of the remaining fields of the FTL format with values from the Rendezvous message.

    The adapter attempts to get a Rendezvous field with the same name as the FTL field.
    - If the Rendezvous field type maps to an FTL field data type, then the adapter translates the Rendezvous field value to an FTL field value of the appropriate data type.
    - If the Rendezvous field type does *not* map to an FTL field data type, then the adapter either discards the entire message, or skips the field, depending on the attribute of the tag.

    (See Adapter Data Type Mapping: FTL to Rendezvous.)
3. After translating all the FTL fields, the adapter publishes the translated FTL message on the relevant FTL endpoints.

Rendezvous Unnamed Fields

Rendezvous messages support unnamed fields, accessed only by index or field identifier. In contrast, FTL messages do *not* support unnamed fields.

The adapter does not translate unnamed fields. When translating a Rendezvous message, the adapter ignores unnamed fields.

Configuring the Adapter

The adapter configuration file guides the adapter as it translates and forwards messages between Rendezvous applications and FTL applications.

Note: Earlier releases of TIBCO FTL had an adapter component that was similar *but not identical*. If you have already configured adapter functionality within TIBCO FTL, it is straightforward to convert that configuration for the adapter in TIBCO Rendezvous or TIBCO Rendezvous Network Server. The file format changes from XML to JSON. The names of configuration tags are different, though analogous. Some tags that applied to the adapter in TIBCO FTL do not apply in TIBCO Rendezvous and Network Adapter.

Helpful Reference Material

A sample configuration file and a JSON schema are distributed with the product.

Procedure

1. Create an adapter configuration file in JSON format.

   This file determines the message translation and forwarding behavior of the adapter. For syntax and semantics, see Adapter Configuration Reference.

2. Store that file on the adapter host.

   For fault tolerance, store it on each host of the redundant pair.

3. Supply the file name on the adapter daemon command line.

   The adapter reads its configuration file to guide its operation. If the configuration contains errors, the adapter daemon exits immediately.

Adapter Configuration Reference

The adapter configuration file governs message translation.

General Syntax

The adapter configuration file is a JSON document. For information about JSON syntax and terminology, see http://www.json.org. Use commas to separate elements of arrays, or elements of objects.

Top Level

Name	Description
`realm`	Begin the adapter configuration. Top level object. Exactly one. Required properties: • `applicationName` • `url` • `services` Optional properties: • `username` • `password` • `fromFTL`
`applicationName`	Required. Exactly one. String. The FTL side of the adapter uses this name to get its tailored realm from the FTL server. The realm must define an application with this name. Contained in: `realm`.

Name	Description
`url`	Required. Exactly one. String. A pipe-separated list of FTL server URLs. Contained in: `realm`.

JAAS Authentication

Name	Description
`username`	Optional. Required for JAAS authentication. At most one. String. When present, the adapter authenticates itself to the FTL server using this username. Contained in: `realm`.
`password`	Optional. Required for JAAS authentication. At most one. String. When present, the adapter authenticates itself to the FTL server using this password. To hide the password from casual observers, you can first obfuscate the password using the `tibrealmadmin --mangle` (a command line utility in the TIBCO FTL software product). Contained in: `realm`.

Services

Name	Description
`services`	Required. Exactly one array, with at least one element. Configure Rendezvous services, and configure adapter behavior with respect to each service. Each element denotes a Rendezvous service.

Name	Description
	Required properties of each array element: • `port` • `endpoints` • `fromRV` Contained in: `realm`.
`port`	Required. Exactly one. String. The adapter maps this Rendezvous service (that is, a port number) to one or more FTL endpoints, translating messages in both directions. Contained in: elements of the `services` array.
`endpoints`	Required. Exactly one array, with at least one element. Elements are strings. The strings must be unique. The adapter maps these FTL endpoints to the Rendezvous service (see the enclosing `services` element), translating messages in both directions. When this array contains several endpoint elements, the adapter translates each Rendezvous message from the Rendezvous service only once, and forwards the translation to all of the FTL endpoints. Contained in: elements of the `services` array.

Translating Rendezvous Messages to FTL Messages

Name	Description
`fromRV`	Optional. An array with at least one element. Configure a translation from Rendezvous messages to FTL messages.

Name	Description
	Required properties: • `formatName` • `subjectName` • `parseSubject` Optional properties: • `discardMessages` • `replyFieldName` Contained in: elements of the `services` array.
`subjectName`	Required. Exactly one. String. The adapter selects a subset of Rendezvous messages to translate from the enclosing service element. That is, the adapter translates only messages with subjects that match this specification. The specification may contain Rendezvous wildcard elements. For important details, see Rendezvous Subjects. Contained in: `fromRV` elements.
`parseSubject`	Required. Exactly one array with at least one element. This array instructs the adapter as it parses Rendezvous message subjects into fields in the target FTL format. Each array element has the form *field_name*`:`*n* In each element, *field_name* designates a field in the target FTL format, and *n* represents a subject token position in the Rendezvous subject. The parser stores the *n*th subject token as the value of the field *field_name*. The data type of all of the FTL fields must be `TIB_FIELD_TYPE_STRING`.

Name	Description
	If the Rendezvous subject name has fewer tokens than the parser requires, the adapter discards the message immediately. Contained in: elements of the `fromRV` array.
`formatName`	Required. Exactly one. String. The adapter translates Rendezvous messages into this managed FTL format. Contained in: elements of the `fromRV` array.
`discardMessages`	Optional. At most one. Boolean. This value governs adapter behavior when it cannot translate a Rendezvous field into an FTL field type. When `true`, the adapter discards the entire Rendezvous message and does not produce an FTL translation. When `false`, the adapter skips the offending field and continues its attempt to translate the rest of the message to an FTL message. When absent, the default is `false`. Contained in: elements of the `fromRV` array.
`replyFieldName`	Optional. Required for request/reply interactions. At most one. String. Specifies a field name in the FTL format. If the Rendezvous message includes a reply subject, the adapter maps that Rendezvous inbox name to an FTL inbox, and stores the FTL inbox in this FTL field. Later, when an FTL program sends a reply to that FTL inbox, the adapter translates the reply and forwards the translation to the original Rendezvous inbox subject. Contained in: elements of the `fromRV` array.

Translating FTL Messages to Rendezvous Messages

Name	Description
`fromFTL`	Optional. An array with at least one element. Configure a translation from FTL messages to Rendezvous messages. Required elements: • `formatName` • `assembleSubject` Optional properties: • `discardMessages` • `replyFieldName` • `expectReplyFormatName` Contained in: `realm`.
`formatName`	Required. Exactly one. String. Specifies a managed FTL format. The adapter uses information in this object to translate all FTL messages with this format. Contained in: elements of the `fromFTL` array.
`assembleSubject`	Required. Exactly one array with at least one string element. This array instructs the adapter as it assembles designated fields of an FTL message into a Rendezvous subject name. Each element of the array is a string designating a field name in the FTL format. The adapter assembles a Rendezvous subject by concatenating the values of the FTL fields, separating them with period (`.`) characters. Contained in: elements of the `fromFTL` array.

Name	Description
`discardMessages`	Optional. At most one. Boolean. This value governs adapter behavior when it cannot translate an FTL field into a Rendezvous field type and also when it cannot translate a Rendezvous field in a reply message into an FTL field type. When `true`, the adapter discards the entire message and does not produce a translation. When `false`, the adapter skips the offending field and continues its attempt to translate the rest of the message. When absent, the default is `false`. Contained in: elements of the `fromFTL` array.
`replyFieldName`	Optional. Required for request/reply interactions. At most one. String. Specifies a field name in the FTL format. When present, the adapter maps the inbox value of this FTL field to a Rendezvous inbox reply subject, which it stores on the Rendezvous message. Later, when a Rendezvous program sends a reply to that reply subject inbox, the adapter translates the reply and forwards the translation to the original FTL inbox. Contained in: elements of the `fromFTL` array.
`expectReplyFormatName`	Optional. Required for request/reply interactions. At most one. String. Specifies an FTL format. When present, the adapter remembers that the FTL requestor expects a reply message with this format and uses this format when translating the reply. Contained in: elements of the `fromFTL` array.

Running the Adapter Daemon

Although running the adapter daemon command line is straightforward, correct operation requires correct configuration and client connections.

Procedure

1. Configure the FTL server to support the adapter daemon as a client.

   See *TIBCO FTL Administration*.

2. Configure the adapter.

   See Configuring the Adapter.

3. Ensure that the FTL server is running, and that the adapter daemon can connect to it.

4. Start the adapter daemon, `rvda`.

   Supply the location of the adapter configuration file as a command line parameter.

   Remember that the adapter daemon is also a Rendezvous daemon, and supply appropriate command line parameters. See *TIBCO Rendezvous Administration*.

5. Ensure that Rendezvous clients connect to the adapter daemon.

   To exchange messages with the FTL network, Rendezvous clients must connect specifically to the adapter daemon.

Adapter Daemon Command Line Reference

Administrators use `rvda`, the adapter daemon command line executable, to start an adapter daemon process.

The adapter daemon supports all the command line parameters that an ordinary daemon (`rvd`) supports. In addition, it also accepts parameters specific to the adapter, as described in this topic.

Adapter

Parameter	Arguments	Description
`-config`	`filename`	The adapter reads its configuration from this file (in JSON format). See Adapter Configuration Reference

Adapter Administration: Realm

The realm definition must satisfy *all* of these constraints to support the FTL side of the adapter.

Communication through Transports

- The adapter must be able to communicate with the FTL server.
- Administrators must configure transports to establish a bus to carry messages between FTL programs and the FTL side of the adapter.
- The realm must configure appropriate communications paths for each message stream.

Bidirectionality

During its start phase, the adapter creates a publisher and a subscriber for each of its endpoints. Administrators must ensure that the realm supports these objects for each adapter endpoint. That is, for each adapter endpoint, the connectors in the adapter's application instance definition must cover at least the send ability and the receive ability. This requirement applies even if you intend that messages flow through the adapter in only one direction.

One-to-One Abilities

If the adapter uses an endpoint to forward one-to-one messages, then the connectors that implement that endpoint within the adapter's application instance definition must cover either the send inbox ability, the receive inbox ability, or both, as needed.

In contrast, if the adapter does not forward one-to-one messages, then the connectors need not cover either of these abilities.

Adapter Data Type Mapping: FTL to Rendezvous

The mapping from FTL message fields to Rendezvous message fields is straightforward. Each FTL type maps to a Rendezvous type, according to the following table. The mapping is intuitive and requires no data conversion.

Some FTL types do not map to any Rendezvous representation. These fields raise an error (see Data Type Mapping Errors).

Data Type Mapping: FTL to Rendezvous

FTL Type	Rendezvous Type
TIB_FIELD_TYPE_OPAQUE	TIBRVMSG_OPAQUE
TIB_FIELD_TYPE_LONG	TIBRVMSG_I64
TIB_FIELD_TYPE_LONG_ARRAY	TIBRVMSG_I64ARRAY
TIB_FIELD_TYPE_DOUBLE	TIBRVMSG_F64
TIB_FIELD_TYPE_DOUBLE_ARRAY	TIBRVMSG_F64ARRAY
TIB_FIELD_TYPE_STRING	TIBRVMSG_STRING
TIB_FIELD_TYPE_STRING_ARRAY	TIBRVMSG_STRINGARRAY
TIB_FIELD_TYPE_MESSAGE	*Error*
TIB_FIELD_TYPE_MESSAGE_ARRAY	*Error*
TIB_FIELD_TYPE_INBOX	*Error*
TIB_FIELD_TYPE_DATETIME	TIBRVMSG_DATETIME
TIB_FIELD_TYPE_DATETIME_ARRAY	*Error*

Adapter Data Type Mapping: Rendezvous to FTL

Because TIBCO Rendezvous software supports a very rich set of data types, while TIBCO FTL software supports a deliberately narrow set of data types, the adapter coerces several Rendezvous types into the same FTL type.

Some Rendezvous types do not map to any FTL representation. These fields raise an error (see Data Type Mapping Errors).

Data Type Mapping: Rendezvous to FTL

Rendezvous Type	FTL Type
`TIBRVMSG_OPAQUE`	`TIB_FIELD_TYPE_OPAQUE`
`TIBRVMSG_BOOL`	*Error*
`TIBRVMSG_I8` `TIBRVMSG_I16` `TIBRVMSG_I32` `TIBRVMSG_I64` `TIBRVMSG_U8` `TIBRVMSG_U16` `TIBRVMSG_U32`	`TIB_FIELD_TYPE_LONG`
`TIBRVMSG_U64`	*Error*
`TIBRVMSG_I8ARRAY` `TIBRVMSG_I16ARRAY` `TIBRVMSG_I32ARRAY` `TIBRVMSG_I64ARRAY` `TIBRVMSG_U8ARRAY` `TIBRVMSG_U16ARRAY` `TIBRVMSG_U32ARRAY`	`TIB_FIELD_TYPE_LONG_ARRAY`
`TIBRVMSG_U64ARRAY`	*Error*
`TIBRVMSG_F32` `TIBRVMSG_F64`	`TIB_FIELD_TYPE_DOUBLE`
`TIBRVMSG_F32ARRAY` `TIBRVMSG_F64ARRAY`	`TIB_FIELD_TYPE_DOUBLE_ARRAY`
`TIBRVMSG_STRING`	`TIB_FIELD_TYPE_STRING`

Rendezvous Type	FTL Type
TIBRVMSG_STRINGARRAY	TIB_FIELD_TYPE_STRING_ARRAY
TIBRVMSG_MSG	Error
TIBRVMSG_MSGARRAY	Error
TIBRVMSG_DATETIME	TIB_FIELD_TYPE_DATETIME
All other Rendezvous types	*Error*

Data Type Mapping Errors

When a data type mapping raises an error, the adapter responds according to the attribute.

This attribute can appear in the tag or the tag.

- If the value of is `true`, then the adapter discards the entire message.
- Otherwise, the adapter skips the offending field and continues translating the message.

Request Reply Interactions through the Adapter

Request/reply interactions involve complex configuration constraints. Only three situations are possible.

Primary FTL Request, Secondary Rendezvous Reply to an Inbox

An FTL program sends a request message. The adapter detects that it is a request message because it meets *all* of these criteria.

- The format of the message matches the format of a configuration tag.
- The of that tag specifies the name of an FTL field, called the *reply field*.
- The realm defines the format such that the reply field has type `TIB_FIELD_TYPE_INBOX`.
- That reply field is present in the message.
- The of the tag specifies the name of an FTL format, called the *reply format*. The requestor expects a reply message in this reply format.
- The realm must define the reply format.

To configure this case, ensure that the adapter configuration meets these six criteria.

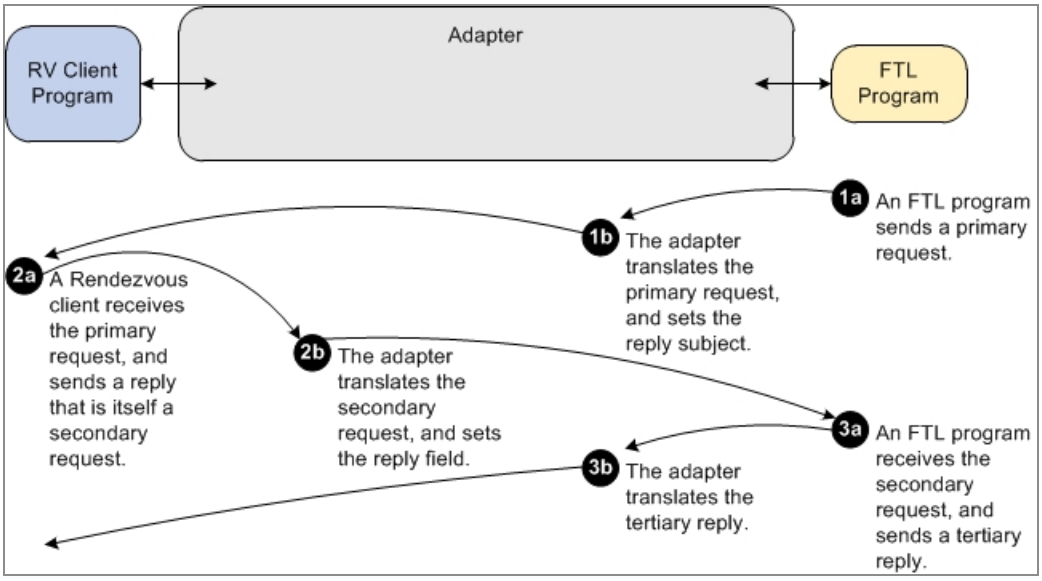
The adapter arranges a mechanism through which a Rendezvous program can reply to the FTL request:

1. The adapter gets the value of the reply field, namely, the FTL inbox where the requestor awaits a reply.
2. The adapter maps that FTL inbox to a Rendezvous inbox within the adapter.
3. In the Rendezvous translation of the request message, the adapter sets the reply subject to that Rendezvous inbox.
4. Later, when a Rendezvous program sends a reply to that inbox, the adapter receives it, translates it to an FTL message using the reply format, and forwards it to the FTL inbox where the requestor awaits the reply.

Secondary Rendezvous Request to an Inbox, Tertiary FTL Reply

A Rendezvous reply to an inbox could itself be a secondary request in a longer chain of point-to-point messages. The adapter detects this situation because the secondary message meets *all* of these criteria.

- The adapter receives the secondary message at an adapter inbox, so the message must be a Rendezvous reply to an FTL request.
- The reply subject of the secondary Rendezvous message contains a Rendezvous inbox name, so the sender must be awaiting a tertiary reply.



The adapter repurposes the reply field from the primary request so an FTL program can send a tertiary reply to the secondary Rendezvous request. The *reply field* is an FTL field name, specified as the value of the of the configuration tag, which is the same tag that governed translation of the primary FTL request message.

1. The adapter gets the reply subject, namely, the Rendezvous inbox name where the requestor awaits a reply.
2. The adapter maps that Rendezvous inbox to an FTL inbox within the adapter, called the *reply inbox*.

3. In the FTL translation of the secondary request message, the adapter stores the reply inbox in the reply field.
4. Later, when an FTL program sends its tertiary reply to the reply inbox, the adapter receives it, translates it to a Rendezvous message, and forwards it to the reply subject. That reply subject is the Rendezvous inbox where the secondary requestor awaits the tertiary reply.

Because you have already configured the adapter to translate the primary request, you do *not* need any additional configuration to *forward* the secondary request, nor to *forward* the tertiary reply. (See Primary FTL Request Secondary Rendezvous Reply to an Inbox.)

Nonetheless, if the format of the tertiary reply differs from the format of the primary request, you must configure a separate tag to *translate* that format. This case has the same form as the configuration for the primary request.

Primary Rendezvous Request to a Subject, Secondary FTL Reply to an Inbox

A Rendezvous program sends a request message to a multicast subject, rather than to an inbox. The adapter detects that it is a multicast request message because it meets *all* of these criteria.

- The subject of the message matches the `subject` attribute of a configuration tag.
- The attribute of that tag specifies the name of an FTL field, called the *reply field*.
- The realm defines the format such that the reply field has type `TIB_FIELD_TYPE_INBOX`.

To configure this case, ensure that the adapter configuration meets these three criteria.

The adapter arranges a mechanism through which an FTL program can reply to the Rendezvous request:

1. The adapter gets the reply subject, namely, the Rendezvous inbox name where the requestor awaits a reply.
2. The adapter maps that Rendezvous inbox to an FTL inbox within the adapter, called the *reply inbox*.
3. In the FTL translation of the request message, the adapter stores the reply inbox in the reply field.
4. Later, when an FTL program sends a reply to that reply inbox, the adapter receives it, translates it to a Rendezvous message, and forwards it to the reply subject. That reply subject is the Rendezvous inbox where the requestor awaits the reply.

Rendezvous Subjects

The Rendezvous subjects you configure for conversion must be distinct.

The adapter validates the `subject` attribute of each tag, using the following rules:

- Within the scope of a `service` tag, the subjects of the tags must *not* collide with one another.

  This restriction does not apply between tags that are in different `service` tags.
- If a literal subject is identical to a previous literal subject, then they collide (*not* permitted).
- If a literal subject matches a previous wildcard subject, then they do not collide (permitted).
- If a wildcard subject matches a previous literal subject, then they do not collide (permitted).
- If a wildcard subject matches a previous wildcard subject, then they collide (*not* permitted).

Examples

Validating Rendezvous Subjects

Subject 1	Subject 2	Conformance
`a.b`	`a.b.c`	Permitted
`a.b`	`a.*`	Permitted
`a.*`	`a.b`	Permitted
`a.b`	`a.>`	Permitted
`a.>`	`a.b`	Permitted

Subject 1	Subject 2	Conformance
a.b	a.b	Collide
a.*	a.>	Collide
a.>	a.*	Collide
a.*	*.b	Collide
*.b	a.*	Collide

Messages that Match Two Rendezvous Subjects

The restriction against colliding subjects helps ensure that the adapter translates each Rendezvous message at most once.

The adapter forbids combinations in which two wildcard subjects collide, preventing ambiguity.

The adapter permits combinations in which a message might match both a literal subject and a wildcard subject. In this situation, the adapter translates the message according to the tag that specified the literal subject.

Adapter Restrictions

Although the adapter is flexible, these restrictions apply.

- The adapter does not support FTL messages with dynamic formats. It can translate an FTL message only if it uses a pre-loaded format. It can translate a Rendezvous message only into a managed FTL format.
- The adapter does not translate FTL messages of types `TIB_BUILTIN_MSG_FMT_OPAQUE` nor `TIB_BUILTIN_MSG_FMT_KEYED_OPAQUE`. Instead, you must define a managed format with one opaque field.

TIBCO Documentation and Support Services

For information about this product, you can read the documentation, contact TIBCO Support, and join TIBCO Community.

How to Access TIBCO Documentation

Documentation for TIBCO products is available on the Product Documentation website, mainly in HTML and PDF formats.

The Product Documentation website is updated frequently and is more current than any other documentation included with the product.

Product-Specific Documentation

The documentation for this product is available on the TIBCO Rendezvous® Product Documentation page.

How to Contact Support for TIBCO Products

You can contact the Support team in the following ways:

- To access the Support Knowledge Base and getting personalized content about products you are interested in, visit our product Support website.
- To create a Support case, you must have a valid maintenance or support contract with a Cloud Software Group entity. You also need a username and password to log in to the product Support website. If you do not have a username, you can request one by clicking **Register** on the website.

How to Join TIBCO Community

TIBCO Community is the official channel for TIBCO customers, partners, and employee subject matter experts to share and access their collective experience. TIBCO Community offers access to Q&A forums, product wikis, and best practices. It also offers access to extensions, adapters, solution accelerators, and tools that extend and enable customers to gain full value from TIBCO products. In addition, users can submit and vote on feature

requests from within the TIBCO Ideas Portal. For a free registration, go to TIBCO Community.

Legal and Third-Party Notices

SOME CLOUD SOFTWARE GROUP, INC. (“CLOUD SG”) SOFTWARE AND CLOUD SERVICES EMBED, BUNDLE, OR OTHERWISE INCLUDE OTHER SOFTWARE, INCLUDING OTHER CLOUD SG SOFTWARE (COLLECTIVELY, “INCLUDED SOFTWARE”). USE OF INCLUDED SOFTWARE IS SOLELY TO ENABLE THE FUNCTIONALITY (OR PROVIDE LIMITED ADD-ON FUNCTIONALITY) OF THE LICENSED CLOUD SG SOFTWARE AND/OR CLOUD SERVICES. THE INCLUDED SOFTWARE IS NOT LICENSED TO BE USED OR ACCESSED BY ANY OTHER CLOUD SG SOFTWARE AND/OR CLOUD SERVICES OR FOR ANY OTHER PURPOSE.

USE OF CLOUD SG SOFTWARE AND CLOUD SERVICES IS SUBJECT TO THE TERMS AND CONDITIONS OF AN AGREEMENT FOUND IN EITHER A SEPARATELY EXECUTED AGREEMENT, OR, IF THERE IS NO SUCH SEPARATE AGREEMENT, THE CLICKWRAP END USER AGREEMENT WHICH IS DISPLAYED WHEN ACCESSING, DOWNLOADING, OR INSTALLING THE SOFTWARE OR CLOUD SERVICES (AND WHICH IS DUPLICATED IN THE LICENSE FILE) OR IF THERE IS NO SUCH LICENSE AGREEMENT OR CLICKWRAP END USER AGREEMENT, THE LICENSE(S) LOCATED IN THE “LICENSE” FILE(S) OF THE SOFTWARE. USE OF THIS DOCUMENT IS SUBJECT TO THOSE SAME TERMS AND CONDITIONS, AND YOUR USE HEREOF SHALL CONSTITUTE ACCEPTANCE OF AND AN AGREEMENT TO BE BOUND BY THE SAME.

This document is subject to U.S. and international copyright laws and treaties. No part of this document may be reproduced in any form without the written authorization of Cloud Software Group, Inc.

TIBCO, the TIBCO logo, the TIBCO O logo, TIB, Information Bus, FTL, eFTL, Rendezvous, and LogLogic are either registered trademarks or trademarks of Cloud Software Group, Inc. in the United States and/or other countries.

Java and all Java based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

All other product and company names and marks mentioned in this document are the property of their respective owners and are mentioned for identification purposes only. You acknowledge that all rights to these third party marks are the exclusive property of their respective owners. Please refer to Cloud SG’s Third Party Trademark Notices (https://www.cloud.com/legal) for more information.

This document includes fonts that are licensed under the SIL Open Font License, Version 1.1, which is available at: https://scripts.sil.org/OFL

Copyright (c) Paul D. Hunt, with Reserved Font Name Source Sans Pro and Source Code Pro.

Cloud SG software may be available on multiple operating systems. However, not all operating system platforms for a specific software version are released at the same time. See the “readme” file

for the availability of a specific version of Cloud SG software on a specific operating system platform.

THIS DOCUMENT IS PROVIDED “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT.

THIS DOCUMENT COULD INCLUDE TECHNICAL INACCURACIES OR TYPOGRAPHICAL ERRORS. CHANGES ARE PERIODICALLY ADDED TO THE INFORMATION HEREIN; THESE CHANGES WILL BE INCORPORATED IN NEW EDITIONS OF THIS DOCUMENT. CLOUD SG MAY MAKE IMPROVEMENTS AND/OR CHANGES IN THE PRODUCT(S), THE PROGRAM(S), AND/OR THE SERVICES DESCRIBED IN THIS DOCUMENT AT ANY TIME WITHOUT NOTICE.

THE CONTENTS OF THIS DOCUMENT MAY BE MODIFIED AND/OR QUALIFIED, DIRECTLY OR INDIRECTLY, BY OTHER DOCUMENTATION WHICH ACCOMPANIES THIS SOFTWARE, INCLUDING BUT NOT LIMITED TO ANY RELEASE NOTES AND "README" FILES.

This and other products of Cloud SG may be covered by registered patents. For details, please refer to the Virtual Patent Marking document located at https://www.cloud.com/legal.

Copyright © 1997-2025. Cloud Software Group, Inc. All Rights Reserved.