Aras Innovator 35

RabbitMQ Configuration Guide

Document #: *D-009072*

Last Modified: *5/13/2025*

Copyright Information

Aras Corporation
100 Brickstone Square
Suite 100
Andover, MA 01810
Phone: 978-806-9400

Notice of Rights

Notice of Liability

Table of Contents

Send Us Your Comments

Aras Corporation welcomes your comments and suggestions on the quality and usefulness of this document. Your input is important in determining the information used for future revisions.

- Did you find any errors?
- Is the information clearly presented?
- Do you need more information? If so, where and what level of detail?
- Are the examples correct? Do you need more examples?
- What features did you like most?

If you find any errors or have any other suggestions for improvement, indicate the document title, and the chapter, section, and page number (if available).

You can send comments to us in the following ways:

Email:
TechDocs@aras.com
Subject: Aras Product Documentation

Or,

Postal service:
Aras Corporation
100 Brickstone Square
Suite 100
Andover, MA 01810
Attention: Aras Technical Documentation

If you would like a reply, provide your name, email address, address, and telephone number.

If you have usage issues with the software, visit https://www.aras.com/support/

1 Introduction

1.1 Purpose

This configuration guide provides information required to configure RabbitMQ for CAE with Aras Innovator and defines the Aras support level for this solution.

1.2 Scope

This document provides an overview of RabbitMQ, instructions on installing and configuring RabbitMQ for CAE, and an alternative solution for the cache invalidation services with Aras Innovator.

1.3 Target Audience

This document is intended for Aras administrators responsible for configuring RabbitMQ to work with Aras Innovator.

2 RabbitMQ Usage Overview

Specific settings and configurations for RabbitMQ are recommended for use with Aras Innovator and are outlined in this document. These settings are based on the current documentation provided by RabbitMQ and may be subject to change at any time by RabbitMQ. Aras is not responsible for RabbitMQ, so any further questions on its use or issues arising from using RabbitMQ should be directed to RabbitMQ.

Aras Innovator is an enterprise application that customers can host in a clustered environment. Deploying Aras Innovator on multiple servers allows the implementation to scale to support more concurrent users and processes. Load balancers uniformly distribute the workload associated with user requests across the configured nodes, allowing for a better end-user experience. However, deploying Aras Innovator across multiple nodes introduces the need to synchronize cached data across the different nodes.

User operations can impact the state of metadata across the system. This information is cached to provide quicker access across requests and improve overall system performance. Changes to cached data on one node mean that the cached data will become stale and out of sync on the other nodes. For the data to be properly updated and consistent, the cached data needs to be invalidated across all nodes – not just the node that handled a specific user request. Failure to invalidate and update the cached data across all nodes will result in inconsistencies across the different nodes, resulting in inconsistent results generated across all nodes.

Metadata syncing across nodes in Aras Innovator is accomplished through server-side cache invalidation. When a cached key-value pair is invalidated on one server, the corresponding cache segment is invalidated on the other server(s).

By default, Aras Innovator utilizes a SQL Server database-based invalidation broker to facilitate cache synchronization. However, this implementation does not work in environments where SQL Server is unavailable. In such scenarios, Aras Innovator can be reconfigured to support an alternative cache invalidation broker implementation to ensure that cached metadata is properly invalidated across all nodes when a change occurs on any of the nodes. Aras Innovator can be configured to work with RabbitMQ to provide these services.

RabbitMQ is a message brokering service that allows applications to implement event-driven processing. Aras Innovator supports a configurable cache invalidation framework that allows a RabbitMQ-based implementation (and potentially other implementations) to replace the default database-based implementation to synchronize the invalidation of caches across each node. The implementation allows certain Aras Innovator caching invalidating events to broadcast when triggered. Each server node would be configured with a listener to receive these broadcast events. The local nodes receiving the event would then apply the same process effectively as the node that originally processed the event. All stale cached metadata would be invalidated across all nodes.

RabbitMQ must be installed separately from Aras Innovator and configured to work with the message broker to provide event-driven invalidation services. The remainder of this document provides the necessary details for setting up RabbitMQ as the invalidation broker for Aras Innovator.

2.1 Cache Invalidation Process Overview

RabbitMQ acts as a messaging service to provide messages between nodes for cache invalidation.

The following steps outline the process of cache invalidation:

1. A cache invalidating action is performed on Server “A” (e.g., an ItemType item received a modification to its definition and was saved).
2. Server-side cache is invalidated for Server “A.”
3. A message containing the Key to be invalidated is added to Rabbit’s Queue.
4. RabbitMQ disperses messages to all listening servers.
5. Server “B” invalidates the corresponding cache entry in its own server cache.

2.2 Support Level

RabbitMQ is a third-party stand-alone application that can be used with Aras Innovator to provide an alternate solution for cache invalidation services. It has not been implemented nor supported by Aras, and it is not required to work successfully on Aras Innovator. This document will describe the basic instructions for installing RabbitMQ and configuring it to work with Aras Innovator. It presumes that the standard installation procedure provided by RabbitMQ is sufficient to get RabbitMQ installed and running in a manner that Aras Innovator can readily utilize.

While Aras will attempt to help a customer resolve an issue related to the configuration of RabbitMQ for use with Aras Innovator, Aras Support will not be responsible for general RabbitMQ installation and setup issues that are unrelated to Aras Innovator. This includes but is not limited to environmental issues such as OS-related issues or issues that may result from conflicts with other applications that may be installed on the same system. Customers are advised to use the specific version(s) of software identified in this document and to use server configurations that match as much as possible to those identified in this document. This will help customers minimize potential conflicting issues.

As RabbitMQ is an open-source product, Aras Support can only address issues related to configuring RabbitMQ as described in this document. If your company requires general RabbitMQ support services, please access RabbitMQ Support for additional information.

Aras support for RabbitMQ configuration is restricted to Windows-based deployment of Aras Innovator.

3 RabbitMQ/Erlang OTP Installation

3.1 Installation Overview

RabbitMQ is an open-source message broker that provides extensible message queuing support. It is implemented using the Erlang OTP language.

The following steps outline the process of setting up RabbitMQ to use with Aras Innovator:

1. Set up the RabbitMQ server.
2. Install Erlang OTP.
3. Install RabbitMQ.
4. Configure Aras Innovator server nodes.

RabbitMQ should only be used in Aras Innovator clustered environments, as synchronizing cache across nodes is necessary only then.

3.2 RemoteMQ Server Setup

A Windows Server 2019 server must be allocated to run the RabbitMQ services. The resources needed for this service will vary from customer to customer depending on the workload required to support the Aras Innovator cluster. The following section provides more guidance for determining the resource needs for a given RabbitMQ server: RabbitMQ Cluster Sizing and Other Considerations.

Alternatively, a RabbitMQ consultant can be consulted to provide guidance.

A dedicated RabbitMQ server is recommended. RabbitMQ must be installed on a non-HA node like an independent service machine. Do not install it on an Aras Innovator node.

3.3 Erlang/OTP Installation

Erlang/OTP is required to run RabbitMQ.

The following steps outline the process of installing the Erlang/OTP:

1. Download Erlang / OTP 26.0.2 Windows 64-bit installer from the following site: {https://www.erlang.org/downloads}.

   Note: Various versions of Erlang OTP may be available, and the one referenced above may not be the most recent version. However, use the one referenced above to reduce potential issues and restrict the configuration to one explicitly tested and/or used successfully by Aras.

2. Install Erlang/OTP 23 by following the instructions provided on the site.

3.4 RabbitMQ Installation and Setup

The following steps outline the process of installing and setting up the RabbitMQ:

1. Install and configure RabbitMQ using the RabbitMQ documentation{ https://www.rabbitmq.com/docs/install-windows}.
2. Create an administrative user which will be used later in configuration.

   Go to http://localhost:15672 on the server where RabbitMQ is installed and log in with the Default credentials:

   Username: guest

   Password: guest
3. Create another administrative user account as follows:
   - Go to the Admin tab
   - Add a user
   - Enter any specific name
   - Enter any password & confirm password
   - Set Admin from Tags
   - Click on Add user
   - Click on the Name of the added user
   - Click on Set Permissions
   - Click on the Admin tab & verify that the administrative user successfully added
   - Record username and password for a later step

   RabbitMQ should now be installed and verified.

3.5 Aras Innovator Configuration

The following steps outline the process of configuring Aras Innovator to use RabbitMQ in place of the default SQL Server invalidation broker:

1. Open the InnovatorServerConfig.xml on the current Aras Innovator server node.
2. Make the following changes:

   Turn off the current invalidation broker by commenting out its configuration entry:
   - **Original entry:** `<Cache invalidation_broker_type="DatabaseDependency"/>`
   - **Deactivated (“commented out”) entry:** `<!-- Cache invalidation_broker_type="DatabaseDependency"/-->`

   Create a cache invalidation broker entry for the newly configured RabbitMQ server by adding the following entry (at the same XML level as the original entry):

```
<Cache
            RabbitServiceHost="%ARAS_RABBIT_HOST%"
            RabbitUser="%ARAS_RABBIT_USER%"
            RabbitPassword="%ARAS_RABBIT_PASSWORD%"
            RabbitExchange="%ARAS_RABBIT_EXCHANGE%"
            RabbitHostPort="%ARAS_RABBIT_HOST_PORT%"
            invalidation_broker_type="%ARAS_CACHE_DEPENDENCY%" />
```

   Substitute in the appropriate values:
   - %ARAS_RABBIT_HOST% - Host server that is hosting RabbitMQ
   - %ARAS_RABBIT_USER% - Administrative RabbitMQ user (recently added administrative user)
   - %ARAS_RABBIT_PASSWORD% - Administrative RabbitMQ password
   - %ARAS_RABBIT_EXCHANGE% - Rabbit MQ Exchange on which messages will be sent

- %ARAS_RABBIT_HOST_PORT% - Port for messaging on RabbitMQ (Default is 5672)
- %ARAS_CACHE_DEPENDENCY% - Should be set to 'MessageQueueDependency'

For example:

```
<Cache RabbitServiceHost="192.168.201.71" RabbitUser="admin_44"
RabbitPassword="admin_44" RabbitExchange="arasExchange_test"
RabbitHostPort="5672"
invalidation_broker_type="MessageQueueDependency" />
```

3. Repeat steps 1 and 2 for all Aras Innovator server nodes.
4. Once all nodes have been updated, restart IIS on all nodes. This will cause each node to be restarted with the updated configuration settings.
5. Validate connections within RabbitMQ.

   Login to RabbitMQ server from one of the Aras Innovator nodes and go to http://{ip_of_server_where_RabbitMQ_is_installed}:15672

   Login with the following credentials for the new administrative user:

   User Name: %ARAS_RABBIT_USER%

   Password: %ARAS_RABBIT_PASSWORD%

   - new Exchange ('%ARAS_RABBIT_EXCHANGE%') should be created.
   - Queues, Channels, and Connections should be created on the RabbitMQ page.

3.6 Connecting SSL Support to Rabbit MQ

The following steps outline the process of connecting SSL Support to RabbitMQ:

1. Edit the `<Cache.../>` string in **InnovatorServerConfig.xml** on Innovator Server. Add the following attributes:

   `RabbitSslEnabled="true"` - parameter turns TLS support on or off. It is off by default.

   `RabbitSslCertPath="/path/to/client_key.p12"` - path to the client's certificate in PKCS#12 format.

   `RabbitSslCertPassphrase="yourPassword"` - If your certificate has a password, specify it here. If your certificate doesn't have a password, leave it blank.

   `RabbitSslServerName="SAN_RABBIT_HOST"` - expects this to match the Subject Alternative Name (SAN) or Common Name (CN) on the certificate that the server sends over.

Example blow:

```
<Cache RabbitServiceHost="10.17.144.7" RabbitUser="innovator1"
RabbitPassword="innovator" RabbitExchange="arasExchange_test"
RabbitHostPort="5671"
invalidation_broker_type="MessageQueueDependency"
RabbitSslEnabled="true" RabbitSslCertPath="c:\Certificates\tls-
gen\basic\result\client_SUP-2016-PERF.p12"
RabbitSslCertPassphrase="innovator" RabbitSslServerName="SUP-2016-
PERF"/>
```

2. Restart RabbitMQ.
3. Restart IIS on Innovator server.

3.6.1 Checking the Connection Between Aras Innovator and RabbitMQ

1. Launch RabbitMQ UI client.
2. Log into the system.
3. Confirm that the new exchange is created with SSL / TLS similar to the image below.