Aras Innovator 33

Configuring Vault Replication

Document #: *D-008098*

Last Modified: *8/22/2024*

Copyright Information

Aras Corporation
100 Brickstone Square
Suite 100
Andover, MA 01810
Phone: 978-691-8900

E-mail: support@aras.com

Website: https://www.aras.com/

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 an important part of 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 Overview

Aras Innovator has the capability to replicate vaulted files across multiple vault servers, allowing users to retrieve files from their local vault, or from another vault in an order of preference, if a copy exists there, rather than having to get the file from the vault it was originally saved. This capability was designed to work transparently to the user, with straightforward configuration by the administrator.

Vault Replication is available on servers with feature licenses only. This document outlines these features, how to configure them and how to use them.

- How to create and configure vaults for replication.
- How to create replication rules.
- Types of vault replication.
- Configuration settings for common use cases
- Programming notes

In Aras Innovator, there is no 'master vault'. Files are uploaded to a specific vault based on a specific User's selected Default Vault property. With the right permissions, a user can get a file from any vault.

Replication of a file can be due to when a user requests a file (a 'get'), when the file is uploaded to the vault, or when other events occur and explicitly request a check for the need to replicate (programmatically through a call to a method) a file.

Since a file can now exist in multiple vaults, and a replication may not have occurred for a file which has been changed (either due to no rule triggering a replication transaction, the transaction being scheduled but not yet being run, or the transaction failing due to some reason), Aras Innovator keeps track behind the scenes of all vaults housing a file. It automatically locates a copy when requested from the users' local vault, if it exists there, or finds it and delivers it from another vault.

Replication works in a multi-step asynchronous process.

First, an event triggers the Aras Innovator server to check if a file should be replicated, by checking the replication rules for a match. For each rule that matches, a replication transaction record is created, specifying the file ID, source vault, target vault, requested execution time, a 'Filter Method' and other parameters.

Next, the Aras Innovator server checks (at regular intervals determined by a configuration file through an internal thread, or through a process scheduled externally, or manually through a special call) to find queued replications due to run.

These transactions must meet a variety of conditions:

- The transaction is due for execution.
- The file exists and is not claimed.
- No other transaction is already executing for that file (which is a 'Pending' status in the replication transaction queue).
- File is absent in the target vault.
- Passes the test given by the filter method, if specified, by the initiating replication rule.

If any of the first three conditions are negative, the transaction is skipped but remains in the queue. If any of the last two are negative, the transaction is removed and logged as discarded.

For transactions that meet the conditions, the Aras Innovator server sends execution requests to the target vault servers.

These are acknowledged, processed by the vault server, and when complete a message is sent back to the Aras Innovator server to be logged. The Replication Logs should be reviewed regularly to determine and diagnose any problems that exist with the configuration.

During the process of replication, the replication transaction status is set to pending, and the file is temporarily claimed until the replication transaction completes or it fails.

When a user saves a file, it is always saved into the Default Vault defined for the user. When a user gets a file from Aras Innovator, the selection of which vault can be defined by a new User Relationship named ReadPriority. For any file retrieved, the vault with the lowest numeric value for ReadPriority with a copy of the file is selected to return the file to the user. If no vaults with a ReadPriority have the file (or none were defined), it is returned by the user's Default Vault, or finally by any other vault having a copy of the file.

1.1 Installation of Aras Agent Service

In order to configure Vault Replication, the Aras Agent Service must also be installed. For instructions on installing the Aras Agent Service, refer to Section 5.12 of the *Installation Guide*.

2 Creating and Configuring Replicated Vaults

To aid the administrators in deployment, each of the following topics is laid out in the order they should be considered to set-up Aras Innovator vault replication.

2.1 Obtain a Feature License

Vault replication requires a **Feature License** in order to run. Vault replication is enabled with the Activation key provided as a Subscription benefit.

If you do not have a current feature license, refer to *Feature License in Aras Innovator 12 – Installation Guide* for instruction on obtaining one.

2.2 Install Aras Agent Service

Vault Replication requires the Aras Agent Service in order to run. For instructions on installing the Aras Agent Service, see Section 5.12 of the *Installation Guide*.

2.3 Configure Alternate Vault Installations

The first step in setting up vault replication is to create the alternate vaults, if they do not already exist. Refer to **Configuring Alternate Vaults Installations** *section* in the *Aras Innovator - Installation Guide* for instructions to set up each vault.

2.4 Replication Process Configuration

Aras Innovator must periodically check for replication transactions in the queue that are due to run. There are two ways to configure the server running Aras Innovator to execute replication processing – either (a) within the Aras Innovator Server as a thread or (b) outside of Aras Innovator as an Aras Innovator service/scheduler or a program configured with standard Windows scheduler.

Each option has pros and cons. The advantage of the option (a) is that there is no need to configure an additional external service in order to initiate the processing of the replication queue. The disadvantage of the option (a) is where the Aras Innovator server is installed in a cluster environment the queue processing thread on each server of the cluster might compete with each other which might cause certain problems.

Besides scheduled replication transactions, some transactions may be created with no scheduled execution time, and are ‘Manual’ replication transactions. These are run through a special method call described later in this section. Also, see *Programming Notes – Manual Replication Transactions* for more information on programming.

2.4.1 Within Aras Innovator Server – Replication Configuration File

If you choose to have the Aras Innovator server run a special thread that handles the replication queue, the thread depends on a set of parameters (e.g. ‘queue processing interval, etc.). When the Aras Innovator server starts it reads a replication configuration file (called replication.config) that defines replication parameters.

The replication.config file is located in, and must reside in the Aras Agent Service. It is initialized to be disabled, and must be enabled, and populated with appropriate configuration settings to start replication with this option.

Because one Aras Innovator server can support multiple databases the configuration file might contain several nodes (one per database) with replication parameters. If a database supported by the Aras Innovator server does not have a corresponding node in the replication.config file, then files referenced from the database are never replicated. The replication parameters include:

- Replication interval is a time interval in seconds between consecutive invocations of the replication processing thread that goes through replication transactions and executes transactions that must be executed. The parameter is optional with the default value 60 sec.
- User name is the user, who is used by the processing thread for making requests to vault servers. The user is required purely for authentication purposes as any request to a vault server must have a valid user credentials set in the request header. All claiming/update/etc. operations are done by the processing thread on behalf of the user but with granting the user administrative privileges using GrantIdentity(...). Based on the above, it's recommended to create a special vault replication user in Aras Innovator with minimal privileges (i.e. who belongs only to World identity) and specify this user in the replication configuration file. This is a parameter with the default value: user=vadmin. If you use the default, make sure you have a user with a vadmin username, otherwise, the routine does not run.
- Max (Transactions per) Batch and Maximum (Transactions) Pending are the maximum amount of replication transactions that could be executed in a single ProcessReplicationQueue request (max_batch) and maximum amount of pending transactions allowed at any time (max_pending). If not specified, the server uses the default values for those parameters which are max_batch=25 and max_pending=50.

  Note: The default values were picked based on experiments performed using Windows 2012 Server on a machine with four processors and 4 GB of memory. If the machine running the Aras Innovator server has more processors and more memory and none of vault servers are run by the same IIS worker process as the Aras Innovator server these numbers could be increased.

  Based on testing, it is recommended to pick these numbers roughly as: max_batch = number of processors on the server * 10
  max_pending = max_batch*2

  If these numbers are too high, the IIS may hang up.

The replication.config file is considered to be active only if it contains the attribute status that has value enabled. In all other cases the configuration file is considered as inactive and the replication queue processing thread is not started.

The format of the replication.config file is the following (optional parameters shown in []), and where {db name} is the name of the Aras Innovator database:

```
<replication status="enabled">
  <rnode db="{db name}" [user="..."] [interval=".."] [max_batch=".."]
[max_pending=".."] />
  ...
</replication>
Given that format, and example replication.config file could look like this:
<replication>
  <rnode db="Innovator" user="vadmin" interval="90" max_batch="30"
max_pending="60" />
</replication>
```

Note: After making adjustments to the replication.config file, the Aras Innovator Agent should be restarted via **Control Panel --> Administrative Tools --> Services**.

2.4.2 Processing Queue Controlled Outside Aras Innovator Server

The queue processing can be initiated by sending to the Aras Innovator server a request with SOAP action ProcessReplicationQueue. The following AML may be passed to the SOAP action: <Item type='ReplicationTxn' [max_batch='...'] [max_pending='...'] /> where max_batch is the maximum amount of replication transactions that could be executed in a single ProcessReplicationQueue request and max_pending is the maximum amount of pending transactions allowed at any time. If not specified, the server uses the default values for those parameters which are max_batch=25 and max_pending=50.

The returned result XML has the following format: `<Result><Item type="ReplicationTxn" [processed="{n}" failed="{n}" need_processing="{n}" locked_by_others="{n}"/>]</Result>` (inside the standard `<SOAP-Envelope><SOAP-Body>` tags),

where attributes have the following meaning:

- `processed` - integer number of transactions successfully processed.
- `failed` - tells how many of transactions among processed transactions failed and were assigned status Failed. A typical situation why a replication transaction processing can fail at this stage is, for instance, when the vault server to which the replication transaction was sent is not accessible.
- `need_processing` - integer that tells how many transaction in the queue are ready to be processed but were not processed in this processing cycle (e.g. because number of currently pending transactions were equal to max_pending).
- `locked_by_others` - integer that tells how many of transactions that need processing (see need_processing above) cannot be executed because the file is claimed not by the user on which behalf the replication queue processing is performed but by someone else. If claimed_by_others equals need_processing it usually means that the processing loop should be stopped because no more transactions could be executed unless user(s) who claimed those file unclaim them.

Warning Do not call this server method repeatedly on the server only within Aras Innovator, since Aras Innovator server calls get combined into one transaction which is committed only upon exit.

Note: In Aras Innovator, the nash utility now has an additional SOAP action 'ProcessReplicationQueue' which can be executed for development testing purposes.

2.4.3 Processing Manual Replication Transactions Programmatically

Replication rules can be configured to make some replication transactions not have a scheduled execution time. These transactions have a Replication Mode set to Manual. If any manual transactions exist, they are held until executed programmatically through a call to a special new server method in the form - Aras.Server.Replication.Queue.Process(XmlDocument inDom,XmlDocument outDom).

inDom has the XML format: <Item type="ReplicationTxn" [max_batch="..."] [max_pending="..."] /> , as shown in section 2.4.2.

The returned outDom XML has the following format: <Result><Item type="ReplicationTxn" [processed="{n}" failed="{n}" need_processing="{n}" claimed_by_others="{n}"/>]</Result> (inside the standard <SOAP-Envelope><SOAP-Body> tags), Definitions of each are shown in section 2.3.

Warning	Do not call this server method repeatedly on the server only within Aras Innovator, since the Aras Innovator server calls get combined into one transaction which is committed only upon exit.

See *Programming Notes – Manual Replication Transactions* for more information on programming.

3 Configuring a User's Preferred Vault

A user's Default Vault is the vault server where all files saved by the user are saved. For systems configured for Vault Replication, it is also the preferred vault from which files are retrieved for a given user, unless specific Read Priority vaults are set. Vaults can be specified and prioritized for each user by setting one or more vaults, including the User's Default Vault, on the 'Read Vaults' relationships tab for a User.

When a user gets a file, Aras Innovator determines all the vaults with copies of the file, and selects the vault to send the file based on:

1. if it is the vault with the highest Read Priority (lowest numeric value),
2. it is the user's default vault (if that vault is not in the Read Priority list),
3. it is another vault.

Setting the Read Vaults should be done for users to improve performance accessing files, based on latency, bandwidth, server load and other considerations.

A User's Default Vault and Read Vault priority is also significant in triggering Replication. The user's Default Vault sets, where a file is first saved or changed, so Replication Rules must be set on that vault to generate onChange Replication Transactions. The Read Priority, and the vault location(s) of the the file, determines which vault Aras Innovator chooses to send the file, and Replication Rules must exist on the chosen vault to generate onDemand Replication Transactions. Replication Rules are described in the next chapter.

3.1 Setting the Read Vaults Priority for a User

Administrators may set the priority order for Read Vaults by editing a User. If the relationship tabs are not visible, click the Hide Tabs button on the toolbar. Then select the Read Vaults tab. Add/edit/delete the vault relationships, setting the Priority property from lowest to highest for the most to the least preferred vaults. If no Read Vaults are specified, the server chooses the Default Vault set for the user as most preferred, and if a copy of the file is not there, it chooses some other vault to retrieve a file. If Read Vaults are specified, the one with the lowest Priority numerically with a copy of the file is chosen to retrieve a file, followed by the Default Vault (if it is not prioritized in the Read Vaults) list, followed by any other vault.

Generally, if you set any Read Vault priorities for a user, the highest priority (lowest numeric value Priority) Vault should also be their Default Vault. One exception would be if your organization wants users to always save to a 'Master Vault', so it is set as the Users' Default Vault, but other vault(s) are to be used when possible to get a file, and thus are set as the Users' Read Priority vault(s).

4 Creating Replication Rules

This section outlines how replication rules are created. Replication rules define which files get replicated, from where, to where, and when. These rules are checked when a file is downloaded from a vault server (this happens when users get or view a file), when a file is uploaded to a vault server (this happens when users add or modify a file), and when requested programmatically. Replication rules are created for each vault server, and can be written to move files to another vault server or copy files to multiple vault servers. User Default Vault and Read Priority settings, as mentioned in the prior chapter, interact with Replication rules, so care must be taken in defining both to work together to give the desired results.

Replication rules are checked by the Aras Innovator Server, and any match causes the creation of replication transaction record(s) for the specified file and source vault to each target vault specified in the rule. These transaction records are regularly checked by the Aras Innovator Server based on the settings determined in how the Replication Transaction thread was configured (in Section 2), and sent to and executed by the vault servers per the date/time the transaction is due to execute.

4.1 Adding a Replication Rule

To add a replication rule:

1. Click 

to edit the source Vault for the files (**Administration>File Handling>Vaults**).

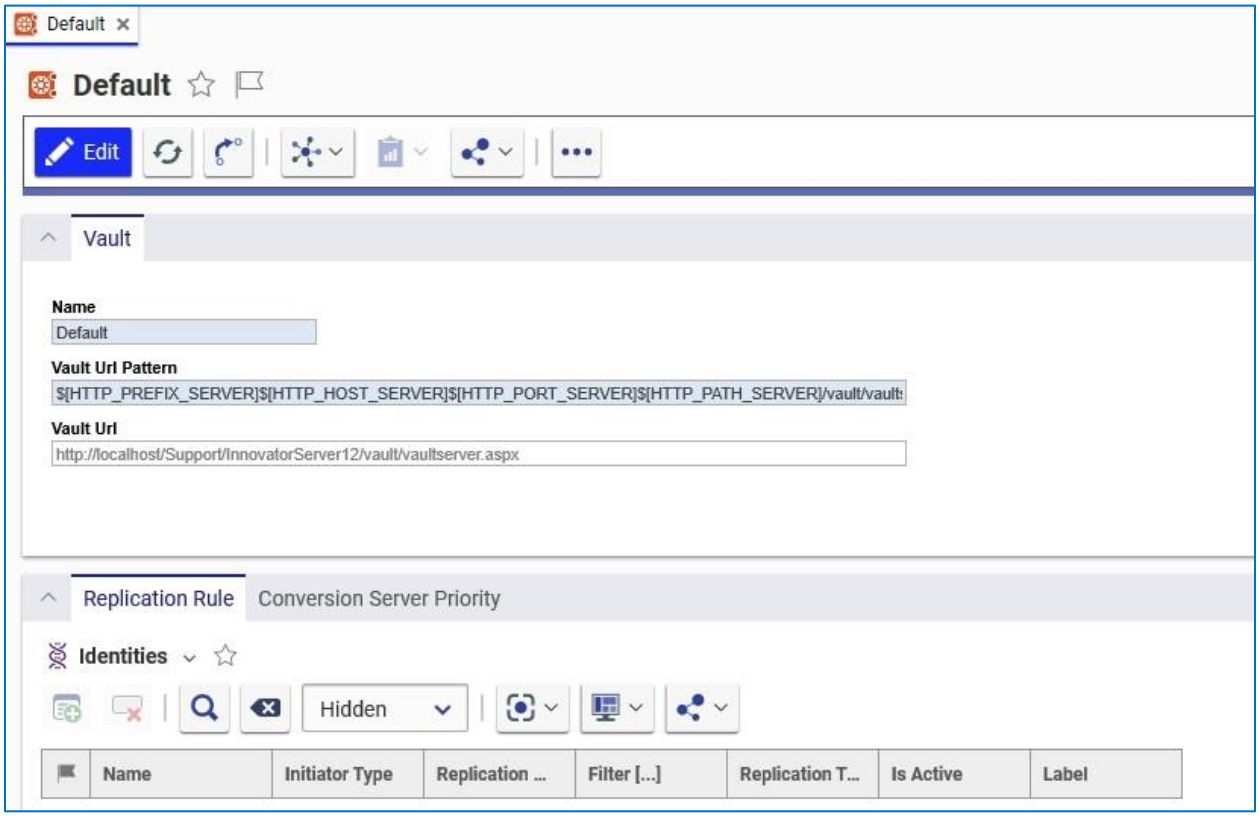

Figure 1.

2. On the new **Replication Rule** tab, click the Add Identities icon  . The Identity search dialog box appears.

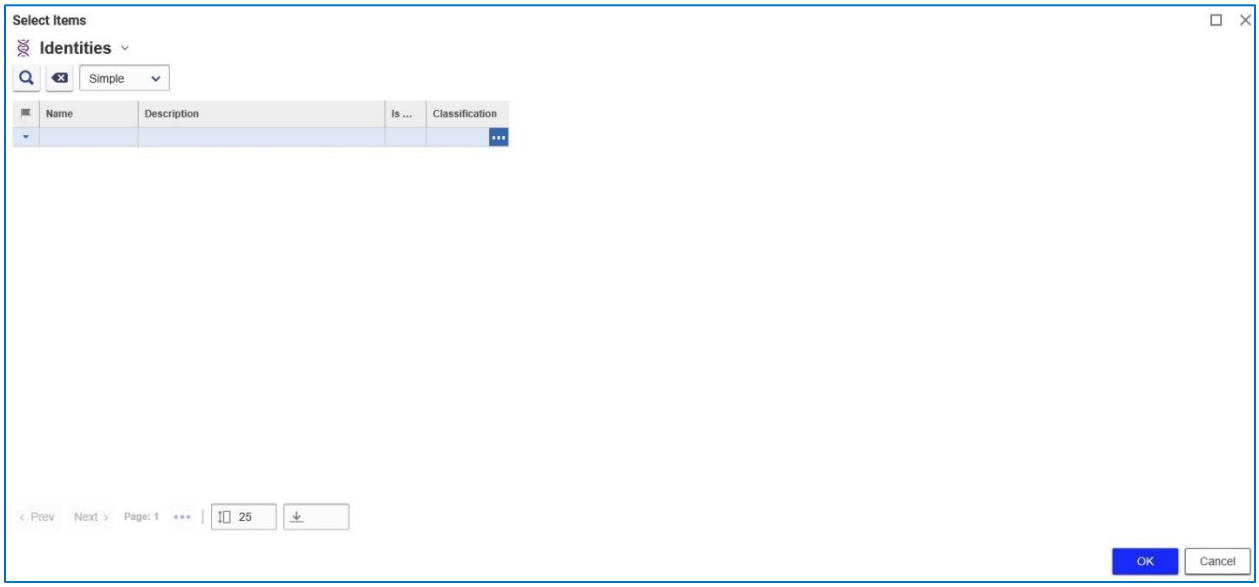

Figure 2.

3. Click the Search icon  and select the identity that executes the initiating event. Usually, this is set to World. If you would like to replicate files that are read or updated in the vault by only a particular group of users then specify that identity on the ‘Replication Rule’; otherwise set the identity to ‘World’.

4. Right click on the new entry, and click **Replication Rule>Open**. The **Replication Rule** form appears.

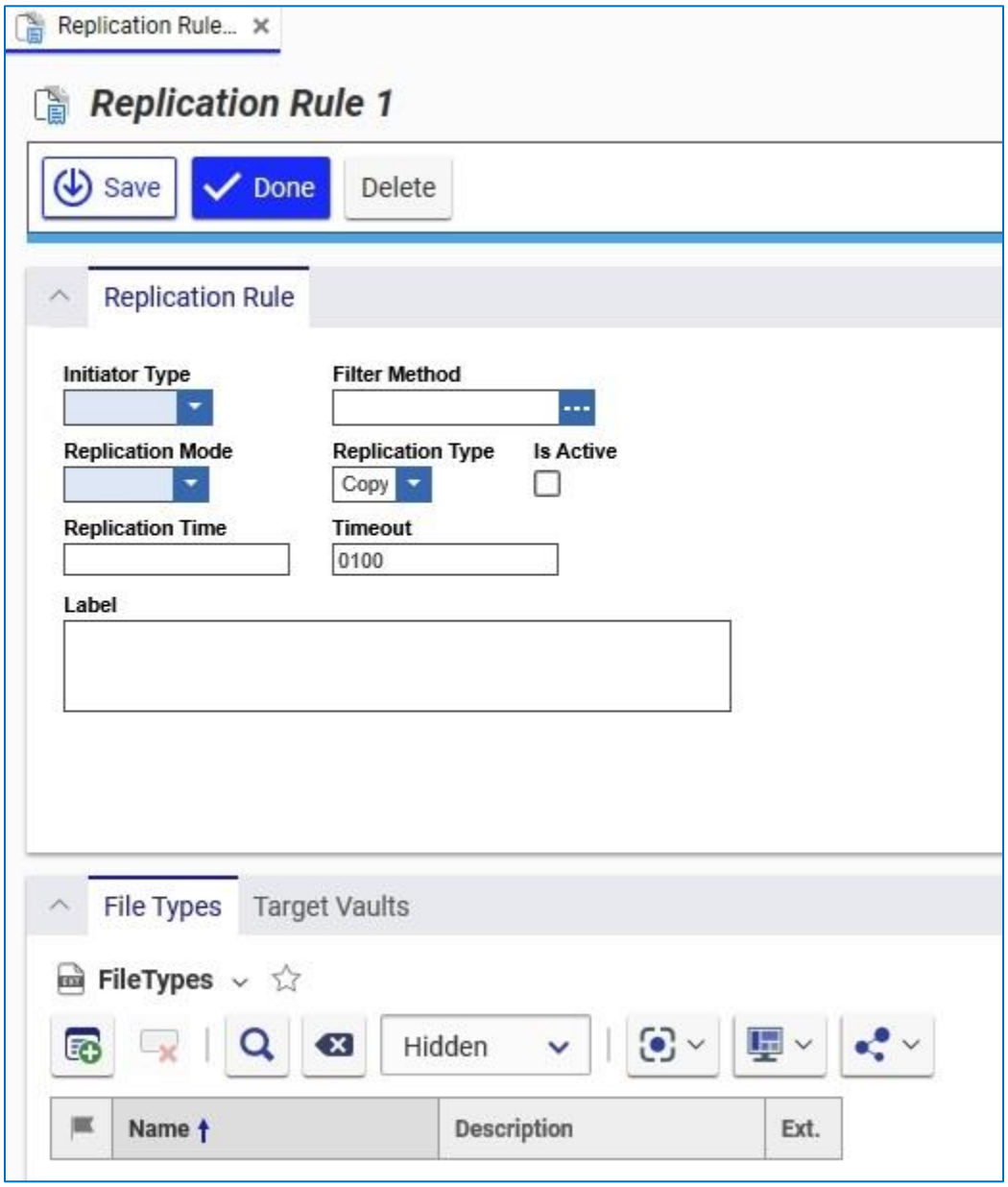

Figure 3.

5. You may set the following properties:
   - Initiator Type defines the initiating event to run this rule. The options are:
     - **onDemand**, for cases where a user belonging in the rule's related identity tries to get the physical file. In this case, target vault is the highest priority "read' vault of the user initiating this request, and related Target Vaults (see below) are ignored. If a file already exists in more than one vault, keep in mind that the rule may or may not create a replication transaction, depending on which vault the file is retrieved from. Therefore, you should usually have similar onDemand rules for each vault to get consistent behavior.

- **onChange**, for cases where the physical file is added or changed by a user belonging in the rule's related identity.
- **onEvent**, for cases where a method explicitly requests replication checks be performed. The user must belong in the rule's related identity, or must be granted special temporary credentials in the method through a GrantIdentity() call. onEvent replication rules are checked when AML of the form "<Item type='File' action='replicate' id='…' />" is sent to the Aras Innovator server for a specific file ID. Optionally, preferred vault(s) to replicate from may be specified by adding the Located relationship(s) in preference order to the AML. However, if none of the vaults specified have a copy, or no vaults are specified, Aras Innovator checks for the first vault with a copy.

OnDemand and **onChange** events are automatically checked when a get or upload, respectively, occurs for a specific file on a vault. To replicate files for any situation other than these, use a method to pass File Item ID(s) with a replicate action, and create a matching **onEvent** rule.

- **Filter Method** identifies a server method which is used to determine if a file having a replication transaction record should be replicated at the time the replication transaction is about to execute. It references a method with signature Item Method (Item file) that gets an IOM.Item of type File with the file ID set, and returns an IOM.Item with AML <Result>1</Result> in case the file must be replicated, or any other AML if the replication should not occur. The method is called immediately before executing a replication transaction created based on the rule and is not when the replication rule is checked, nor when the replication transaction is created. The reason for this is the necessity to check file properties (or properties of other items to which the file could be related) immediately prior to executing the transaction, as the properties could change since the replication transaction is created. If the filter method determines that the replication should not occur, it is recommended that the result be set to a text value indicating why, so this reason appears in the replication transaction log.

Note: In case the method returns "<Result>Do NOT replicate the file</Result>" the transaction is discarded (i.e. its status is set to Discarded) and the content of the returning <Result> tag is set on the ReplicationTxn.error_msg property.

- **Replication Mode** specifies the timing for executing a replication transaction.
  - **Immediate** sets the execution time to the moment the transaction was created.
  - **Delayed** sets the execution time to the moment the transaction was created plus the defined Replication Time.
  - **Scheduled** sets the execution time to the UTC time specified in the Replication Time field defined below for today, or tomorrow if that UTC time has already passed for today.
  - **Manual** does not set an execution time, and the transactions are held until executed programmatically through a call to a special new method in the form - Aras.Server.Replication.Queue.Process(XmlDocument inDom, XmlDocument outDom). Note, a single call of Aras.Server.Replication.Queue.Process(...) only runs one iteration of the replication queue processing, so if there are more 'Manual' replication transactions than could be executed in a single replication queue processing iteration, some are not be completed, so you must take that into account in your code. See the *Programming Notes* section for more detail on Manual Replication.
- **Replication Type** determines if the file is left in the source vault or not
  - **Copy** leaves the file in the source.

        - **Move** removes the file from the source vault after copying it to the target vault. Note that you should only have one target vault (or none chosen for onDemand Initiator types) when this option is chosen.
    - **Is Active** allows specific replication rules to be either in use or ignored for debugging and other administrative purposes. Those set to active are checked, and those not set are ignored.
    - **Replication Time** is relevant for **Delayed** and **Scheduled Replication Modes** only. For **Delayed** modes, it is the quantity of time in hours and minutes from when the rule creates a transaction record to the earliest that transaction should execute. For **Scheduled** modes, it is the hour and minute of the day in UTC Time (today if later than the present time, tomorrow if that time has already passed for today) representing the earliest the transaction should execute. Format for both modes is hhmm with 0000 representing immediately for **Delayed**, and midnight for **Scheduled** modes. The maximum allowed value for **Scheduled** modes is 2359, and any value not recognized is treated as 0000.
    - **Timeout** represents the maximum amount of time allowed from when a transaction execution request is sent to a vault server and accepted (and the transaction is set as Pending) until it completes either successfully (Completed) or not (Failed). Transactions not completed in the allotted time are automatically failed (Failed). A default value of 0100 (one hour) is set and assumed. Note that the vault server may be processing multiple requests. Format is hhmm.
    - **Label** is an arbitrary text field that is output on the replication transaction log. Labeling the rules may be helpful in searching and reading the logs.

Two relationships are made in the Replication Rule:

- **File Types** specifies and restricts which types of files are replicated using the rule. If no file types are specified, all file types are included.
- **Target Vaults** specifies the vault or vaults to copy the file to. This relationship is not appropriate, and is ignored, for rules where the Initiator Type is onDemand since the target is set to the initiating user's highest 'read' priority vault. Only one vault should be specified where the Replication Type is Move. Having more than one target results in transaction errors since the file no longer exist in the source vault after the first completed 'Move' transaction.

4.2 Removing a Replication Rule

You can set a replication rule as Inactive (refer to the description of the Is Active property) and you can delete a replication rule completely. However, if there are any outstanding Replication Transactions, you cannot delete a rule until they are completed and put into the Transaction Log. If you wish to delete a rule, you can first set it as Inactive, to inhibit the creation of any new replication transactions, wait for any existing transactions using that rule to complete or manually delete them, then delete the replication rule.

5 Resubmitting Failed Replication Transactions

This section describes the configuration and operational details of how failed replication transactions can be resubmitted. This feature allows recovery from the situation where for some reason the replication of files from one of the source Vaults to one or more of the destination Vaults fails. Such failures are expected occasionally in any distributed application, such as issues with Network Connectivity. Automatic, Manual, and Batch resubmissions are employed to recover from the failures. The total number of attempts actually performed is recorded in the Property labeled 'Execution Attempt' of the 'Replication Transaction' and the 'Replication Transaction Log' item types. The following are the details of the three recovery mechanisms that can be employed based on the nature and extent of the issue:

5.1 Automatic Resubmission

To work around failures in replication transactions caused by temporary conditions such as network errors, Aras Innovator can be configured to automatically resubmit failed Replication Transactions up to 5 times. To configure this setting, you must change the value of the MaxAutomaticReplicationAttempts Variable Item:

MaxAutomaticReplicationAttempts: integer value between 1 and 5 (The default value is 1)

Note: Aras Innovator stops automatic resubmission after 5 attempts, even if the value of the variable is set to be greater than 5.

5.2 Manual Resubmission

Administrators may review the Replication Transaction Log Items for failures that could not be automatically resolved. After diagnosing and resolving the cause of Failures, the Administrator can manually resubmit any 'Failed' Replication Transaction Log Item as a new Replication Transaction using 'Resubmit' from the Action menu for the Replication Transaction Log Item. Replication Transactions can be manually resubmitted any number of times until the transaction succeeds, and the number of attempts is not restricted by the configured value of the MaxAutomaticReplicationAttempts Variable.

By default, resubmitting the Replication Transaction also deletes the existing Replication Transaction Log item. To retain this failure log, you must edit the RemoveReplicationLogOnRerun Variable Item and set the value to false.

RemoveReplicationLogOnRerun: possible values are true/false (default value true)

If batch resubmission is used, it is recommended that the Variable value remain 'true'.

5.3 Batch Resubmission

To allow the system to continuously attempt to recover failed Replication Transactions, the method '`GroupResubmitReplicationTxn`' has been made available for use with the Aras Innovator Scheduler Service. By default, this method builds a list of failed Replication Transaction Log Items from the last 24 hours and automatically run the 'Resubmit' action against each of these transactions.

Due to the variable nature of how many replication transactions this method creates, it is recommend that you run this method during hours of low-usage of the system. The results of this method should be reviewed periodically to ensure it is not creating unwanted network traffic in peak usage hours.

Refer to the *Aras Innovator - Scheduler Service* documentation for information regarding the installation and configuration of the Aras Innovator Scheduler Service.

It is possible that the `GroupResubmitReplicationTxn` Method may not help in all situations, and may not be used in all installations. The `GroupResubmitReplicationTxn` Method could be customized to use different filter criteria for building the list for resubmission, but doing so requires that you include the testing of this Method as part of your acceptance tests after any changes to the system, like an upgrade or deployment of a new solution.

6 Typical Replication Configurations

This section provides examples of typical types of replication configurations. These can be used as is, or may be modified for your special circumstances. A combination of these configurations may be used.

These replication rule configurations interact with the Read Vaults Priority and Default Vaults you assign to each user. In most cases, it is recommended to set the user's Read Priority vaults to give them the best performance at their usual location for reading the files.

There are two options for setting the User's Default Vault. The first is to assign it to the Vault which gives them the best performance, often the same one as the highest Read Priority. The second is to assign it to a 'Master' vault. There are tradeoffs for each approach. The first approach allows you to use the 'onDemand' replication rules to place files onto their local vault when they first view them, the second approach puts files immediately onto a master vault.

6.1 Replicate All Files to All Vaults

This configuration allows users to almost always access files from their Read Priority vaults, unless a file has not yet been replicated (because it is claimed, or because the scheduled replication has not yet executed). Usually, you would want to set this type of replication to run immediately or to be scheduled to occur during off hours, to capitalize on periods with low server use and lower network traffic.

For this configuration, one replication rule must be set for each vault -

- **Initiator Type** – onChange
- **Filter Method** – none (blank)
- **Replication Mode** – Immediate or Scheduled
- **Replication Type** – Copy
- **Is Active** – checked
- **Replication Time** – If Scheduled, set to an off hour. You should consider staggering the replication times for the different rules, so excessive replication transactions are not set to execute all at the same time.
- **Timeout** – depends on the installation, but a default value of 0100 (one hour) is a good starting point.
- **Label** – a unique name for each rule.

And for the two relationships made in the Replication Rule -

- **File Types** – should be blank.
- **Target Vaults** – each other vault.

6.2 Replicate All Files to a Master Vault

This configuration is good for an organization where there is one main location with many users, and smaller satellite locations, or where you want to maintain a backup repository, and you don't want to assign the master vault as every user's Default Vault. Often this would be combined with onDemand rules to copy files to user's highest priority Read Vault (or Default Vault if there are none) from the master vault when they first open a file.

Usually, you would want to set this type of replication to run immediately or to be scheduled to occur during off hours, to capitalize on periods with low server use and lower network traffic.

For this configuration, one replication rule must be set for each vault except the master vault.

- **Initiator Type** – onChange
- **Filter Method** – none (blank)
- **Replication Mode** – Immediate or Scheduled
- **Replication Type** – Copy
- **Is Active** – checked
- **Replication Time** – If Scheduled, set to an off hour. You should consider staggering the replication times for the different rules, so excessive replication transactions are not set to execute all at the same time.
- **Timeout** – depends on the installation, but a default value of 0100 (one hour) is a good starting point.
- **Label** – a unique name for each rule.

And for the two relationships made in the Replication Rule -

- **File Types** – should be blank.
- **Target Vaults** – only the master vault.

6.3 Move All Files of Specific Types to a Specific Vault

This configuration is especially useful for cases where files of certain types are either exceptionally large and that vault is specially configured to handle them, or are regularly accessed by another server-based application at that vault site, or for special backup or security reasons.

For this configuration, one replication rule should be set for each vault which is configured as the Default Vault for any user who might save that type of file, except the specific vault to which the files are to be moved.

- **Initiator Type** – onChange
- **Filter Method** – none, or configured based on the business need. Remember, the filter method is checked before the replication transaction for a file is executed.
- **Replication Mode** – Scheduled, Delayed, or Immediate, depending on the file type and business need
- **Replication Type** – Move
- **Is Active** – checked
- **Replication Time** – If Scheduled, set to an off hour. You should consider staggering the replication times for the different rules, so excessive replication transactions are not set to execute all at the same time.
- **Timeout** – depends on the installation, but a default value of 0100 (one hour) is a good starting point. Large files may require longer Timeout values.
- **Label** – a unique name for each rule.

And for the two relationships made in the Replication Rule -

- **File Types** – select the type(s) of files to move.
- **Target Vaults** – the selected vault (only one should be selected for a Move replication type).

6.4 Copy Files to Vault(s) When a Document is Released

This configuration provides Released Document Files to users at remote sites. It requires writing a method which is specified to run on a Lifecycle Transition to the Released state. See *Programming Notes – Document Release Transition onEvent Method* for an example of that code.

For this configuration, a master vault would typically be used as the main repository, and one replication rule must be set for the master vault, or otherwise for all vaults which might have the only copy of a 'released' document file.

- **Initiator Type** - onEvent
- **Filter Method** – none, or configured based on the business need. Remember, the filter method is checked before the replication transaction for a file is executed.
- **Replication Mode** – Scheduled, Delayed, or Immediate, depending on the file type and business need
- **Replication Type** - Copy
- **Is Active** - checked
- **Replication Time** – If Scheduled, set to an off hour. You should consider staggering the replication times for the different rules, so excessive replication transactions are not set to execute all at the same time.
- **Timeout** – depends on the installation, but a default value of 0100 (one hour) is a good starting point. Large files may require longer Timeout values.
- **Label** – a unique name for each rule.

And for the two relationships made in the Replication Rule -

- **File Types** – select the type(s) of files to move.
- **Target Vaults** – the selected vault(s).

6.5 Copy Files On Demand to a User's Highest Read Priority Vault Upon First Access

This configuration allows copies of files to be local to a user so that subsequent access (for anyone else at their site) is faster. It is useful for files that need repeated access, but may not be needed by users at all sites. It minimizes access times, and reduces site-to-site data traffic and storage requirements at the sites.

For this configuration, one replication rule should be set for each vault:

- **Initiator Type** - onDemand
- **Filter Method** – none, or configured based on the business need. Remember, the filter method is checked before the replication transaction for a file is executed.

- **Replication Mode** – Scheduled, Delayed, or Immediate, depending on the file type and business need.
- **Replication Type** - Copy
- **Is Active** - checked
- **Replication Time** – If Scheduled, set to an off hour. You should consider staggering the replication times for the different rules, so excessive replication transactions are not set to execute all at the same time.
- **Timeout** – depends on the installation, but a default value of 0100 (one hour) is a good starting point. Large files may require longer Timeout values.
- **Label** – a unique name for each rule.

And for the two relationships made in the Replication Rule:

- **File Types** – None, or select the type(s) of files to copy.
- **Target Vaults** – None (blank). The user's highest Read Priority vault is automatically used.

7 Throttling File Replication Network Traffic

It should be noted that throttling of a website helps limit the network traffic rate originating from this website, i.e., if the primary vault is on this website and is replicating to another vault on another website.

7.1 Procedure to setup Bandwidth Throttling for a Website

It is assumed that necessary setup has been performed to replicate files from the one vault to another and that each vault is running on a separate IIS website.

For IIS7, perform the following for Bandwidth Throttling on the website with Primary vault:

1. Start IIS Manager.
2. Select the website and click **Advanced Settings**.
3. Under section **Connection Limits,** specify the Maximum Bandwidth (in Bytes/sec)
4. Restart the website.

For IIS6, perform the following for Bandwidth Throttling on the website with Primary vault:

1. Start IIS Manager.
2. Right click on the website and select Properties.
3. On the Performance TAB, in the Bandwidth Throttling Section click the Checkbox and select the Maximum Bandwidth (in kilobytes/sec)
4. Restart the website.

Once configured, these settings restrict the download or any files from the website. This limits the amount of bandwidth taken by the replication, but if your website also controls access to Aras Innovator this also restricts the bandwidth used by Aras Innovator. It is recommended you have a thorough execution plan before putting these setting in place, as they can have an impact on end-user experience.

7.2 Verifying the results of Bandwidth Throttling for a Website by Network traffic Monitoring and Logging

To verify that Throttling is playing a role during replication of data the following is one of the methods:

Depending on the specific version of Windows perform either one of the options below:

- **Start--> Settings--> Control Panel --> Administrative Tools --> Performance**

  It brings up a MMC (Microsoft Management Console) pre-populated with System Monitor and Performance Logs and Alerts Snap-ins.

- **Start--> Settings--> Control Panel --> Administrative Tools**

  In the Administrative Tools selections available, double-click **Reliability and Performance Monitor**.
  It brings up a MMC (Microsoft Management Console) with Monitoring Tools folder, under which there is ‘Performance Monitor’.

8 Programming Notes

For details on any subjects in this section, refer to the Aras Innovator - Programmer's Guide. This section is only meant to be a reference to areas of interest for programmers.

8.1 External Program to Run Scheduled Replication

An example of an external program to execute scheduled replication is provided on the Aras Community Projects site under *Vault Replication Examples* (https://github.com/ArasLabs/vault-replication-examples).

The example automatically creates files and submits them to Aras Innovator, which generates replication transactions if a replication rule is properly set. Next, a sample process runs on the queue until it is empty. See the `ProcessReplicationQueueSample.cs` file in the download for more information.

8.2 Manual Replication Transactions

An example of an Aras Innovator action to execute manual replication transactions is provided on the Aras Community Projects site under *Vault Replication Examples* (https://github.com/ArasLabs/vault-replication-examples).

See the readme file in the download for instructions on installing this example. An 'Execute Manual Transactions' Action is added, which calls a client method, which in turn calls the server method repeatedly to process all outstanding Manual replication transactions in the queue.

8.3 File onEvent Method

Here is an example of VB code to request replication on a specific file. It triggers the server to check through the Replication Rules for a matching onEvent replication rule for that requesting user's Identity and Vault. This example would be run through an Item Action on a file item to call this method. You would run this as an administrator by navigating to Administration>Files, locating the file through a search, and running the action on a file.

```
' FileId refers to the file to be replicated, in this case, the File selected
when the action is run
Dim FileId As String = Me.getID()
Dim inn As Innovator = Me.getInnovator()
Dim innrep, innresult As Item
innrep = inn.newItem("File","replicate")
innrep.setId(FileId)
innresult = innrep.apply()
```

8.4 Document Release Transition onEvent Method

Here is an example of a server method to request replication for file(s) attached to a document which is transitioning to Released state. It would be specified as the Post Server Method on the transition(s) to Released state on the document's Lifecycle Map.

```
Dim inn As Innovator = Me.getInnovator()
Dim thisDoc As String = Me.getID()
Dim innrep, innresult As Item
```

```
Dim myQry As Item = Me.newItem("Document File","get")
myQry.setProperty("source_id", thisDoc)
myQry.setAttribute("select","related_id")
Dim results As Item = myQry.apply()
Dim docfiles As Item = results.getItemsByXPath("//Item[@type='Document
File']")
Dim w As Integer = 0
For w=0 To DocFiles.getItemCount()-1
   Dim DocFile As Item = DocFiles.getItemByIndex(w)
   Dim FileId As String = DocFile.getProperty("related_id")
   innrep = inn.newItem("File","replicate")
   innrep.setId(FileId)
   innresult = innrep.apply()
Next w
Return Me
```

8.5 External Program to Submit Files for Replication

An example of an external program to execute scheduled replication is provided on the Aras Community Projects site under Vault Replication Examples https://github.com/ArasLabs/vault-replication-examples).

The example gets a list of files with the filename starting with 'file' in the Default vault on the Aras Innovator server, and requests onEvent replication for each. If an appropriate onEvent replication rule exists on the Default vault, replication transactions should be created. These can then be processed a number of ways, including those shown in 6.1 or 6.2, depending on the Replication Mode set in the replication rule. See the CreateReplicationTxnsSample.cs file in the download for more information.

8.6 Replication Filter Method

Here is a sample of VB server method code to determine if a file should be replicated. Based on the ID of the file, other properties and relationships can be determined and used to make the decision.

```
' FileId refers to the file to be replicated
Dim FileId As String = Me.getID()
Dim FileSize as Integer = Me.getProperty("file_size")
Dim inn As Innovator = Me.getInnovator()
Dim innresult As Item
' put your code here to determine if the file should be replicated or if
' replication transaction should be discarded
If (FileSize > 1000) Then ' put your test condition here
' return a 1 if the file should be replicated
   innresult = inn.newResult("1")
Else
' return an error message for the log if file should not be replicated
   innresult = inn.newResult("Error Message for Replication Log")
End If
Return innresult
```

9 Troubleshooting

Depending on the configuration, vault replication can be somewhat complex to troubleshoot. The following are some guidelines to assist you in handling typical issues.

It is important to isolate what part of the replication process is not running properly. Reviewing the Replication Transactions and Replication Transaction Log is your first step to diagnosis.

9.1 Replication Transactions Not Being Created

- If replication transactions are not being created for a file, either an event to trigger checking of the replication rules has not occurred, or the file does not match the criteria set for any active replication rule. Another rare potential cause is that one or more of the vault servers have not been upgraded to the same Aras Innovator version as the main server.
- Confirm that the installed version of Aras Innovator on each vault server is the same Aras Innovator version as the main server.
- Determine the rule which should have triggered a replication transaction and confirm it is Active.
- Confirm the file exists in the Vault where that rule is defined, and matches a file type in the File Type Filter for that rule, if any are specified.
- Confirm that a copy of the file does not exist on the vault(s) to which the file should replicate.
- For an onChange rule –

  The most common reason for an onChange rule not to execute is that file was saved to a vault which does not have a matching onChange rule. Based on the User who saved the file, confirm a matching onChange rule exists on the Vault for that User's Default Vault.

- For an onDemand rule –

  The most common reason for an onDemand rule not to execute is that the file was retrieved for the user from a different vault than one with the rule. Aras Innovator gets the file based on where the file exists, and selects the vault based on the Read Priority, then Default Vault of the user, or from any other vault. So if the file is retrieved from a vault which does not have an applicable onDemand rule, instead of another vault with such rule, no replication transaction is created.

- For an onEvent rule –

  Confirm that the method written to call the replication request actually executes. If so, confirm that a copy of the file with the file ID specified by the method exists on the vault with the onEvent rule. This may be done by viewing the file as a user configured with that vault as its highest Read Priority, then noting the vault which actually returned the file through the URL of the viewer, or by an AML nash or other query of the Located relationship(s) for the file.

9.2 All Immediate/Scheduled/Delayed Replication Transactions Not Starting

If all immediate, scheduled and delayed replication transactions are not executing, even though they are due to run, the replication transaction thread is not running. Getting it to run depends on how it is configured to run.

In all cases, confirm that the parameters for running it are correct. Common errors are the database is a different name, or the user ID does not match one in Aras Innovator.

If the internal process is used, either the replication.config file is not correct, or not in the right location, or the global.asax was not modified as directed to run the replication process thread.

9.3 Some Immediate/Scheduled/Delayed Replication Transactions Not Starting

If some replications do not start, and they are due to run based on the Not Before value, either the file is most likely claimed, or the replication thread has not yet run following passing the Not Before date/time, or another transaction is in the ‘pending’ (executing) state for that file during this replication iteration.

Also, keep in mind the values chosen for max_batch and max_pending. No more than the max_batch count of transaction records are initiated in one thread cycle, and no more than max_pending transactions run at once, so several iterations over a period of time may be necessary to clear the transactions in queue.

If the replication transaction(s) not starting have a filter method set, check that the filter method is returning a result. If it is not returning a result, or errors out or returns an error, the replication transactions using that filter method remains in the queue. You may use the ‘Check syntax’ button when editing the method, add statements to activate debugging, or temporarily remove the filter method from the replication rule to determine if the problem is due to the method.

9.4 All Manual Replication Transactions Not Starting

If all manual replication transactions do not start, the way you are calling the method must have an error. Create a simple temporary method like shown in the Programming Notes to confirm you have the correct replication settings, and once that is confirmed, debug the rest of your code.

9.5 Some Manual Replication Transactions Not Starting

Reasons for some Manual Replication Transactions remaining not starting are similar to that for Immediate/Scheduled/Delayed transactions. The most common would be the file being claimed. Refer above.

9.6 Some Replication Transactions Not Completing

If replication transactions fail to complete, the replication transaction records or replication transaction log should indicate if the specific transaction was discarded or failed, due to a copy already existing on the target vault, or the filter rule not returning a ‘1’, or the file no longer existing on the source vault. The Status and Message properties for the transaction indicate why the transaction did not complete.

Regularly review the Replication Transaction Log and look for problems and similarities in the transactions that do not complete, to determine if they are acceptable, or if something must be modified in your vault replication configuration.