Multiple MultiOutput Jobs

Fluorine+

Your Outbound Messages are sent out by a single job *Perspectium MultiOutput Processing* which goes to your Outbound Messages table and sends the messages out per queue. This should cover most cases.

However, if you are doing a high volume of messages to a single queue or spreading your messages across a high volume of queues than you can take advantage of the following feature.

The core concept behind this is the ability to pass in an encoded query to the MultiOutput job to limit the scope of these jobs. In other words you can have multiple jobs responsible for their own unique subset of Outbound Messages.

blocked URL **NOTE:** We ***do not*** recommend for you to simply clone the default *Perspectium MultiOutput Processing* without making the following changes. Doing so can cause you to send the same set of messages out multiple times.

Prerequisites

blocked URL First, you will need to create a ServiceNow dynamic share, create a ServiceNow bulk share, or set up a ServiceBond integration.

blocked URL We also recommend that you take a quick look at the *Perspectium MultiOutput Processing* job to familiarize yourself with it and contact s upport@perspectium.com to validate your work if necessary.

Strategies

There are two main strategies behind this process. The one you use will depend on your use. The details for the implementation for each are covered in the following section.

Bulk Processing on a Queue

> This method is not suggested for dynamic shares as messages will not be in order

This refers to wanting to process a high volume of messages on a specific queue. If you are Bulk Sharing a large amount of messages for a single queue, then this is the path you should lean towards.

This sets up sharing to divide the work for a queue into small distinct chunks - and having multiple jobs each process a chunk. The primary way to do this by querying off of the sys_id of the Outbound Message.

It is important to note that this is querying off the sys_id of the outbound message itself and not the record that the Outbound Message represents. Additionally we share out the records in a way to preserve sequencing on a single queue, **this method does not honor that sequencing.** So we would recommend this strategy if you are Bulk Sharing a large set of data and are not concerned about the order they arrive in.

Segregated Processing for a Group of Queues

This refers to creating multiple jobs, each job to handle a specific queue. If you are sharing data to a large number of queues, then this is the path you should lean towards.

This sets up sharing to divide the work of your Outbound Table into groupings based on the queue they are writing to. Since the queues are processed iteratively, this changes the workflow from 1 job processing all queues to X jobs processing their own subset of queues.

This will retain the sequencing of the data.

Procedure

To create multiple MultiOutput jobs, follow these steps:

1

Make a copy of the Perspectium MultiOutput Processing job

Navigate to **Perspectium** > **Replicator** > **Dynamic Share** or **Perspectium** > **Replicator** > **Scheduled Jobs**. Then, click into the Perspectium MultiOutput job and make a copy of it and rename it appropriately.

Example:

blocked URL

2

Pass in an encodedQuery

Notice in the example above we create a variable named **encodedQuery**. This variable gets passed into psp.processMultiOutput(). You can pass any encoded query you want.

A quick way to get an encoded query is go to your Outbound Table, create a filter, and chose the “Copy Query” option. This will give you the encoded query to use.

blocked URL

Bulk Processing Steps

Create a filter on your current outbound messages based on the sys_id starts with X flag. The sys_ids for these records start with (0, 1, 2, 3, 4, 5, 6, 7, 8, 9, a, b, c, d, e, f) for 16 values. We want to make sure we capture each of these distinctly.

Here is an example where we break these into groups of 4.

blocked URL

blocked URL

Then create the other 3 jobs similarly to the one above. You may also want to limit this to a distinct queue by passing in a target queue into the encoded query.

Here is an example of the script

```
try {
    var encodedQuery="sys_idSTARTSWITH0^ORsys_idSTARTSWITH1^ORsys_idSTARTSWITH2^ORsys_idSTARTSWITH3";

    var psp = new Perspectium();
    psp.processMultiOutput(encodedQuery);
}
catch(e) {
    var logger = new PerspectiumLogger();
    logger.logError("error = " + e, "Perspectium MultiOutput Processing");
}
```

Queue Grouping Steps

Create a filter on your current outbound messages for Target Queue is *queue 1* OR Target Queue is *queue 2* OR Target Queue is *queue 3* and copy the encoded query.

blocked URL

Then pass this encoded query into the job.

blocked URL

It should resemble the query below containing the sys_id of the target queues selected:

```
u_target_queue=XXXX^ORu_target_queue=YYYY^ORu_target_queue=ZZZZ
```

Warnings

This is a advanced capability for the Replicator so we recommend running this through your test environments first.

It is important to know that the purpose of these strategies is to send outbound messages with multiple jobs without any overlap in data transit.

Original Job

The original job *Perspectium MultiOutput Processing* will go through each queue without any encoded query within it. If you do go down this path you should either modify or de-activate this job to make sure your jobs are each processing their own subset of data.

You may also want to place a “X” at the start of the name of this so it is *XPerspectium MultiOutput Processing* to avoid it being auto restarted from the “Start All Jobs”. You will also want to follow Perspectium Update Set Releases so you can maintain these jobs.

Dot Walking

From an optimization standpoint, we do not recommend “dot-walking” with the queries. I.E. do not pass in an encoded query like:

```
var encodedQuery = "u_target_queue.u_nameLIKEdev18450"
var psp = new Perspectium();
psp.processMultiOutput(encodedQuery);
```

This will work, however, with higher volumes it will not be as efficient as directly passing in the sys_id of the target queue.

Overloaded Scheduler

A ServiceNow production instance will generally have 4 nodes which can execute 8 jobs each, for a total of 32 available workers. A Bulk Share is a job, a single MultiOutput processing is a job.

So you can create a job per queue. However, it is important to take into account the total available workers on your instance. I.E. you should not create 16 individual MultiOutput processing jobs on a 4 node instance, because than we may be taking 16 of the 32 available workers.

This allows you to ramp up your processing, just take into account the environment of the instance so we do not hog the processing.

Similar topics

- Add ACL rules to Perspectium UI pages
- Uninstall Replicator for ServiceNow
- Attachments
- Stop/Start All Jobs
- Before/after subscribe scripts

Contact Perspectium Support

US: 1 888 620 8880

UK: 44 208 068 5953

support@perspectium.com