**EAS Push to DRS**

|  |  |
| --- | --- |
| **Resources** | |
| JSR 352: Batch Application for the Java Platform | https://jcp.org/en/jsr/detail?id=352 |
| Spring Batch | http://docs.spring.io/spring-batch/ |

**Benefits of using Spring Batch**

EAS uses Spring Batch to manage the jobs that push packets and items to DRS2.

Can use chunking so that even if one chunk fails the previous chunks are processed. Can then restart the failed job once the issue that caused the job to fail has been fixed and the job will continue from where it failed.

A job can only be started and run to completion once with a given parameter – if it fails it **can** be restarted.

**Process**

**User Interface Push to DRS**

The user selects items to push to DRS and initiates the push via the EAS user interface. This groups the selected items into archive batch chunks for pushing to DRS and removes them from the EAS user interface displays.

**Prearchiver**

This is run every 5 minutes via crontab.

The prearchiver launcher selects the next archive batch to be run and then launches the Spring Batch prearchiver job for that batch.

The prearchiver job is made up of several steps:

*readWritepacketStep*

This step finds all the related packets for the items in the archive batch and creates them in DRS 2 if they do not already exist there. It uses the DRS2 Object Ingest Service for this purpose.

*readWriteItemsStep*

For each archive batch chunk in the archive batch this step creates a DRS2 batch of items in an EAS archiveQueue directory.

*copyToDRSDropboxStep*

For each archive batch chunk in this archive batch, this step copies the batches from the archiveQueue directory to the DRS2 EAS secure dropbox. Once all the files for a batch have been copied this writes a LOADING file directly in the DRS2 EAS secure dropbox.

Each EAS archive batch chunk results in one DRS2 batch.

The naming convention for the batch directory is:

EASItem\_<archive\_batch\_id>\_<archive\_batch\_chunk\_id>

*updateArchiveBatchArchivingStep*

This step updates the archive batch state to “archiving” if all it’s associated archive batch chunks have been successfully copied to the DRS2 EAS secure dropbox.

**Postarchiver**

This is run every 5 minutes via crontab.

The postarchiver launcher launches the Spring Batch postarchiver job using the current date/time as an identifying job parameter – this effectively means that the job may be run multiple times.

The postarchiver cleans up the items for any archive batch chunks which have been successfully loaded into DRS2.

*Overview*

DRS writes a LOADREPORT file for each successful load – this is read by the postarchiver.

EAS keeps track in the database of which individual items have not yet been successfully processed.

EAS keeps track in the database of which individual chunks have not yet been successfully processed.

EAS keeps track in the database of which archive batches have not yet been successfully processed.

The postarchiver also makes use of a tracking table – archive\_item\_cleanup\_state.

This table is used to track the cleanup state of each component of an item independently – whether it has been removed from the solr index, the database and the file system.

The archive\_item\_cleanup\_state primary key will be the item id however it will not be a foreign key to the item table. This allows the flexibility to delete the item from the database while retaining the row for the item in the archive\_item\_cleanup\_state so that the item may be removed from the file system after it has been removed from the database.

The postarchiver job is made up of several steps:

*prepareLoadReportsStep*

For each archive batch chunk that is being archived, if a successful load report has been created in it’s DRS2 dropbox, read the report into the archive batch chunk table and update the status to indicate it has been archived. It also copies the successful load report to the archiveQueue directory.

*deleteFromDropboxStep*

For each archive batch chunk that has been archived, delete the corresponding batch directory and it’s contents from the DRS2 dropbox.

*processLoadReportsStep*

For each archive batch chunk that has been archived, parse the item ids from the report and create a row for each item in the archive\_item\_cleanup\_state tracking table. Update the state for the archive batch chunk to indicate it can be cleaned up.

*deleteFromIndexStep*

For each archive batch chunk which can be cleaned up, delete all it’s items from the solr index and update the archive\_item\_cleanup\_state tracking table rows for the items to indicate that those items have been removed from the solr index.

*deleteFromDatabaseStep*

For each archive batch chunk which can be cleaned up, delete all it’s items from the database and update the archive\_item\_cleanup\_state tracking table rows for the items to indicate that those items have been removed from the database.

*deleteFromFileSytemStep*

For each archive batch chunk which can be cleaned up, delete all it’s items from the EAS file system (the converted directory) and update the archive\_item\_cleanup\_state tracking table rows for the items to indicate that those items have been removed from the file system.

*completeArchiveBatchChunkStep*

For each archive batch chunk which was being cleaned up, for any items for that chunk have been deleted from the solr index, database and file system (as recorded in the archive\_item\_cleanup\_state tracking table) then delete those entries from the tracking table.

If all items have been deleted for the archive batch chunk then set it’s state to complete, if not set it’s state to ‘incomplete’.

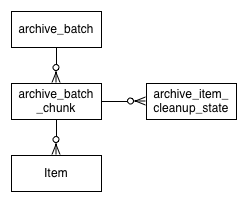
*updateArchiveBatchArchivedStep*

For each archive batch whose state is ‘archiving’, if all it’s associated archive batch chunks have a state of ‘complete’, then set the archive batch state to ‘archived’.

*archiveQueueCleanupStep*

For each archive batch whose state is ‘archived’ , delete all it’s associated batches from the archiveQueue directory. If all the directories are successfully deleted, update the archive batch state to ‘complete’.

**EAS Archive Batch Data Model:**

****

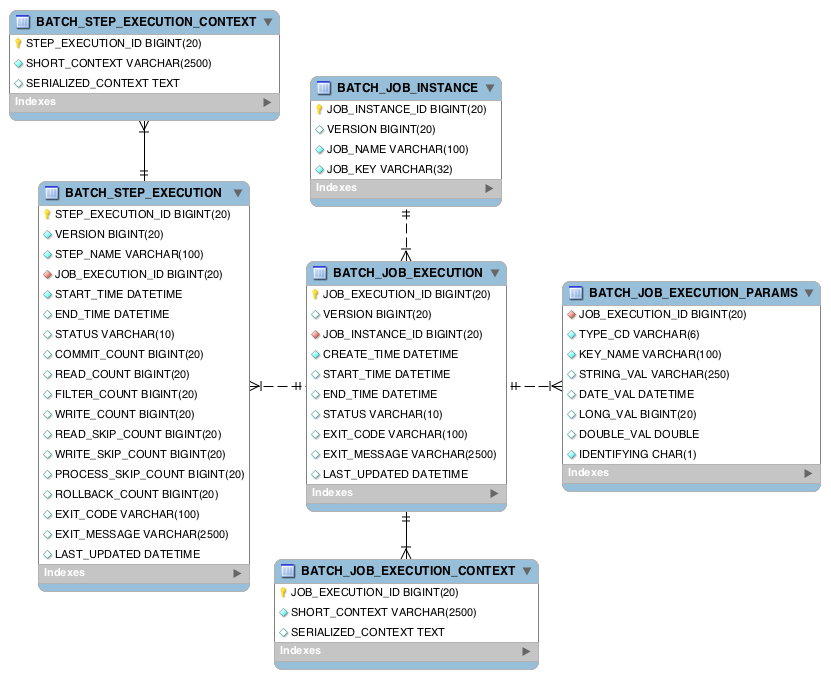
|  |  |
| --- | --- |
| **archiveBatch** | |
| This represents a batch of items selected by a user in the EASi user interface for deposit to the preservation system, DRS2. | |
| Id | Primary key |
| hib\_version | Used by hibernate for optimistic locking |
| archive\_state | Used for tracking the state of this archive batch |
| depositor | Identifier, currently the huid, of the user who “pushed” the batch to DRS2 |
| archive\_failure\_email | Email address where failure reports are to be sent |
| archive\_success\_email | Email address where success reports are to be sent |
| state\_last\_mod\_time | Time the state was last updated |
| create\_date | Time the archive batch was created |

|  |  |
| --- | --- |
| **archiveBatchChunk** | |
| To make a batch more manageable for ingest into the destination preservation system, the batch is subdivided into chunks, ensuring that any associated emails and attachments are in the same batch chunk. | |
| id | Primary key |
| hib\_version | Used by Hibernate for optimistic locking |
| archive\_state | Used for tracking the state of this archive batch chunk |
| report | Success report from DRS2 |
| state\_last\_mod\_time | Time the state was last updated |
| archive\_batch\_id | Foreign key to archive\_batch |

|  |  |
| --- | --- |
| **archiveItemCleanupState** | |
| Used for tracking the cleanup state for an item which has been successfully ingested into DRS2 | |
| id | Primary key – is assigned the id from item which is being tracked |
| hib\_version | Used by Hibernate for optimistic locking |
| locator | The relative location of the item’s file on disk |
| db\_state | Deletion state of the database entry for the item:  1: ready to delete  2: deleted |
| index\_state | Deletion state of the Solr index entry for the item:  1: ready to delete  2: deleted |
| fs\_state | Deletion state of the file system file for the item:  1: ready to delete  2: deleted  9: failed |
| archive\_batch\_chunk\_id | Foreign key to archive\_batch\_chunk |
| eas\_packet\_id | Foreign key to eas\_packet |

**Spring Batch Data Model**:

(<http://docs.spring.io/spring-batch/reference/html/images/meta-data-erd.png> )



**EAS Entity Lifecycles:**

|  |  |
| --- | --- |
| **archiveBatch** | |
| ready\_to\_archive | Archive batch has been created with items associated with it and it is now ready to be archived |
| in\_archive\_queue | The Pre archiver service has picked up this batch and has started processing it |
| archive\_queue\_failed | The Pre archive service failed to process this batch – this could be due to communication problems with Wordshack or DRS2 etc or due to unexpected bad data |
| archiving | The Pre archiver service has finished processing all the chunks for this batch, deposited it to the DRS2 dropbox and it is now ready for the DRS2 loader to process |
| archived | All the chunks for this archive batch have been successfully loaded into DRS2 and the item entries have been removed from the EAS database, solr index and the processing file system. |
| complete | The corresponding files have been removed from the EAS archive queue file system |

|  |  |
| --- | --- |
| **archiveBatchChunk** | |
| ready\_to\_archive | Items associated with this archive chunk are ready to be archived |
| In\_archive\_queue | The Pre archiver service has processed this archive chunk, has built the batch locally and is ready to copy the archive chunk to the DRS2 dropbox |
| archiving | The Pre archiver service has finished processing this archive chunk, deposited it to the DRS2 dropbox and it is now ready for the DRS2 loader to process |
| archived | The items associated with this archive chunk have been archived by the DRS2 loader |
| archive\_failed | The items associated with this archive chunk failed to be archived by the DRS2 loader |
| cleanup | The items associated with this archive chunk may now be removed from the EAS database, index and file system |
| complete | All items in this archive chunk have been successfully loaded into DRS2 |
| incomplete | Some items in this archive chunk have not been successfully loaded into DRS2 – this is an anomaly and should not occur. |

|  |  |
| --- | --- |
| **Item** | |
| ready\_to\_import |  |
| imported |  |
| import\_failed |  |
| ready\_to\_archive |  |
| in\_archive\_queue |  |
|  |  |
|  |  |
|  |  |

**Testing**

To Test entire life cycle of Push to DRS functionality (for developers)

Setup:

Ensure that the build-local.properties file has the following properties set:

archive.chunk.size=3

This will permit testing that a small batch is divided up correctly into chunks.

Ensure that the application has been configured using these values by running:

ant config

Start up EASi:

./bounce\_tomcat.sh start

Verify that started up correctly by examining the following logs:

catalina.out easi\_log4j\_log easi\_log

1. Push to DRS

1. 1. Go to EASi brief display and select HUL.TEST from the dropdown

1.2. Select 10 items from one page – ideally a mixture of email messages and attachments.

1.3. Record the EAS IDs for the items to be pushed.

1.4. Click on the “Push to DRS” link and then on the “Push to DRS” button in the dialog that is displayed. Click “OK” on the dialog which is displayed and which states: Move to DRS successful”.

1.5. The items are no longer displayed in the brief display. To verify that they are no longer searchable go to the brief display and search for the item EAS ID for one of the items which was pushed to DRS. No item is found. Clear the search and click on the EAS ID for any email message that was displayed. In the browser location change the item parameter to one of the email messages that was pushed. A page will be displayed which states that the email message no longer exists.

1.6 Check the database to make sure the archive batch entries were correctly created:

select i.id as item\_id,i.state as item\_state,

b.id as archive\_batch\_id, b.archive\_state as batch\_state,

c.id as archive\_batch\_chunk\_id, c.archive\_state as chunk\_state

from item i, archive\_batch\_chunk c, archive\_batch b

where i.archive\_batch\_chunk\_id = c.id

and c.archive\_batch\_id = b.id

and b.id = <archive batch id>

order by archive\_batch\_id, archive\_batch\_chunk\_id;

Where <archive batch id> is the id for the generated archive batch.

Verify that the returned item\_state, batch\_state and chunk\_state all have a value of “ready\_to\_archive” and that all 10 items are returned.

2. PreArchiver

Setup: Verify that DEV DRS2 web services are up and running.

Verify that QA Wordshack is up and running

Verify that the drs2 file system is mounted

Troubleshooting:

If there is a problem in the prearchiver, check the system/log/prearchive.log file.

If the DRS2 loader fails to load the items, check the DRS loader email messages. Based on that examine the descriptor files for the failed batch.

* 1. Run the edu.harvard.hul.ois.eas.batch.jobs.PreArchiverJobTest test using the following java vm arguments:

-Dlog4j.debug=true -Dlog4j.configuration=file:<workspace>/eas/conf/prearchiver\_log4j.properties -Dbatch=<archive batch id>

Where <archive batch id> is the archive batch id that was returned in step 1.6.

Or

Run bin/drs\_postarchiver.sh

This will pick up the next batch to be run

* 1. Verify that three batches are created (one for each chunk) in the DEV DRS2 eas secure dropbox incoming directory. Note that the batch directory naming convention is:

EASItem\_<archive\_batch\_id>\_<archive\_batch\_chunk\_id>

Wait approximately 5 minutes for the DRS loader to process the batches, then check the DRS dropbox batch directories for successful load reports.

Wait another 5 minutes for the DRS2 processes to index the newly created objects. Then query for the objects using the DRS2 web admin and verify that they look ok. Download and verify that the descriptors look ok.

* 1. Verify the database archive batch states by running the query from step 1.6. This time the item state shoud be “in\_archive\_queue” for all Items, the batch state and the chunk states should be “archiving”.

3. DRS Loader

3.1 Verify that the items have been loaded.

Use DRS2 Web Admin to search for batches with

batch load date > <today>

Verify that the batch names for all the archive batch chunks are listed.

4.1. PostArchiver

Troubleshooting:

If there is a problem in the postarchiver, check the system/log/postarchive.log file.

Step 3.1

Record the location for each item which was loaded (item.locator) so you can ensure the item was deleted.

select 'accounts/'||a.account\_dir ||'/import/'|| i.eas\_packet\_id ||'/'|| i.locator

from item i, account a

where i.account\_id = a.id

and i.id in (<list of EAS item IDs>);

Step 3.2

Run the postarchiver

Step 3.3

Verify that the database states for the batch and chunks are set to “complete”.

select b.id as archive\_batch\_id, b.archive\_state as batch\_state,

c.id as archive\_batch\_chunk\_id, c.archive\_state as chunk\_state

from archive\_batch\_chunk c, archive\_batch b

where c.archive\_batch\_id = b.id

and b.id = <batch id for this batch>

order by archive\_batch\_id, archive\_batch\_chunk\_id;

Verify that the batch directories have been deleted from the secure dropbox.

ls <drs2 eas secure dropbox>/incoming/EASItem\_<batchId>\_\*

no files should be found

Verify that the items have been deleted from the EAS secure file system

Use the results from step 3.2 to verify that those files have been deleted.

Verify that the items have been deleted from the database

select count(\*) from item

where id in (<list of EAS item IDs>);

count should be 0.

Verify that the items have been deleted from the index

[https://localhost.harvard.edu:10035/solr/eas-collection1/select?q=\*%3A\*&fq=item\_id%3A%28<itemId\_1>+OR+<itemId\_2](https://localhost.harvard.edu:10035/solr/eas-collection1/select?q=*%3A*&fq=item_id%3A%28%3citemId_1%3e+OR+%3citemId_2)> . . .<itemId\_n>%29&wt=json&indent=true

numFound should be 0.

Verify that the EAS batch has been deleted from the EAS archive queue directory.

ls <archiveQueue>/drs2/EASItem\_<batchId>\_\*