New Batch Process Technical Document

Table of Contents

- Purpose
- Dependencies (DB tables)
- Logical Data Model
- Physical Data Model
- Service Interface Design (Java)
  - Batch Process Profile
  - Bib Import
  - Order Record Import
  - Invoice Import
- Service Interface Design (REST if applicable)
  - Batch Process Profile
    - Submit Profile (POST)
    - Search Profiles (POST)
    - Edit Profile (POST)
    - Delete Profile (POST)
    - Upload File (POST)
  - Batch Process
    - Upload File (POST)
    - Submit (POST)
    - Create Job (POST)
    - Quick Launch Job (POST)
    - Schedule Job (POST)
    - Destroy Job (GET)
    - Get Reports Files (GET)
    - Get Specific Reports Files (GET)
- User Interface Design
- Data Importing
- Data Exporting (if applicable)
- Workflow
- System Parameters
- Roles and Permissions

Last Published March 14, 2016

Purpose

OLE provides the ability to import and export large chunks of data such as Bibliographic data, Order and Invoice information, Patron records, Location records, etc. through the Batch Process and the Batch Process profile documents. However, there was a growing need for new functionalities and performance in the batch process which necessitated a new batch process. This was addressed through the New Batch Process documents.

The following document would have information on the New Batch Process documents.

Dependencies (DB tables)

Tables part of the New Batch Process in OLE

ole_ng_bat_prcs_job_t	Batch Process Job
ole_ng_bat_prf_t	Batch profile
ole_ng_bat_job_details_t	Batch Job Details

Logical Data Model

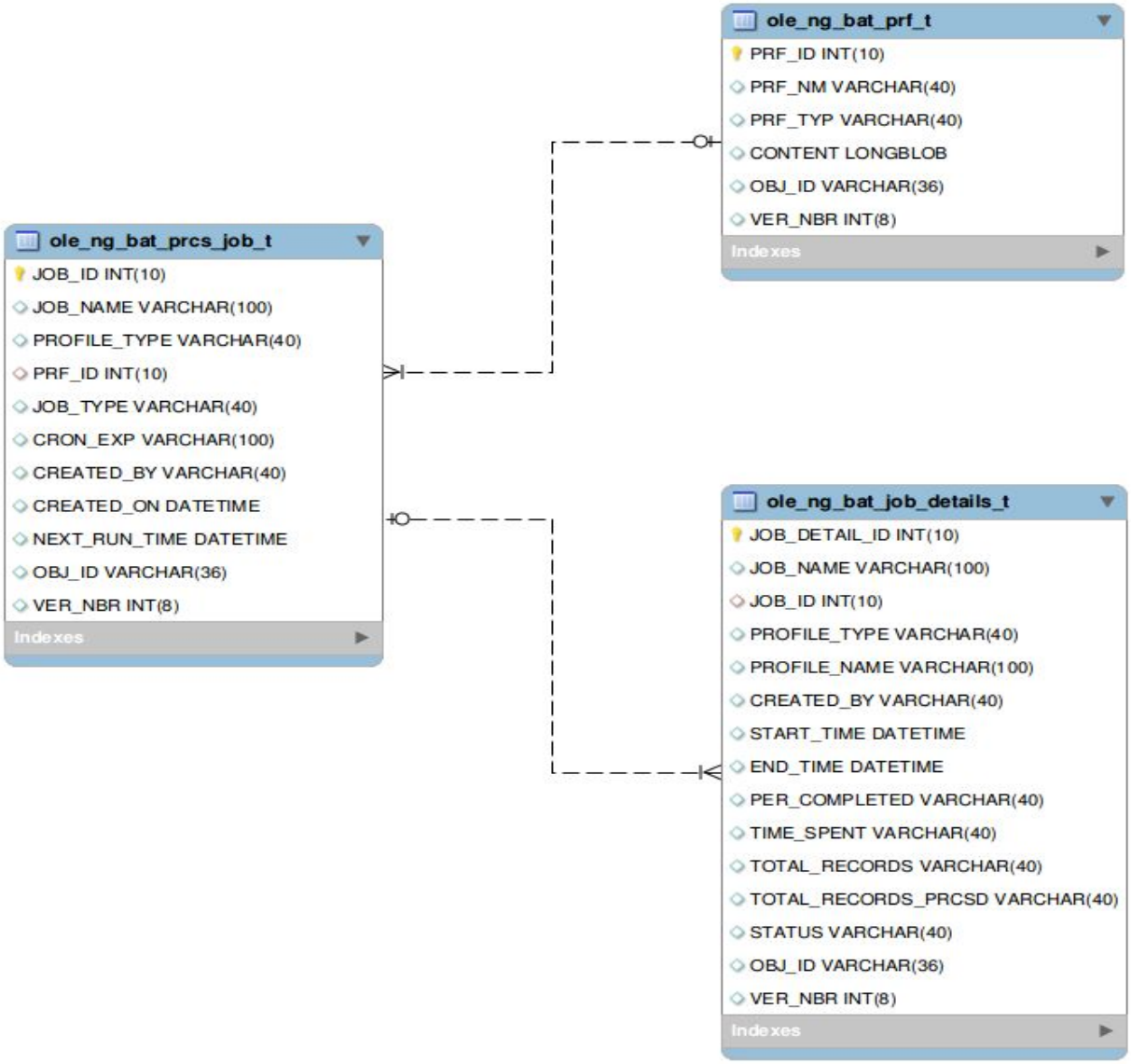
OLE, like the old batch process, makes use of the batch process profile to record match points, data mappings, etc. However, unlike the old process the new batch process profile data doesn't reside in multiple tables. The batch process profile data is saved as a JSON (JavaScript Object Notation) content in the table and is used while processing the batch process data.

Physical Data Model

The new batch process and batch process profile don't use as many tables as used by the old process. This is because the batch process profile which contains all the match points, data mappings and other configurations reside as a JSON content in one of the tables.

The tables involved in this new batch process are *ole_ng_bat_prcs_job_t, ole_ng_bat_prf_t* and *ole_ng_bat_job_details_t*. The ng in the table name stands for Angular since the UI component of the new batch process is designed in AngularJS. The *ole_ng_bat_prcs_job_t* table contains the batch process jobs. The JOB_ID is the primary key and the PRF_ID is the foreign key referenced from the batch profile table. It also contains the CRON_EXP to capture the cron expression based on which the jobs are scheduled. The *ole_ng_bat_prf_t* table contains the batch process profile details. The PRF_ID is the primary key and the CONTENT column contains the various match point, mapping, etc. data in JSON format. The

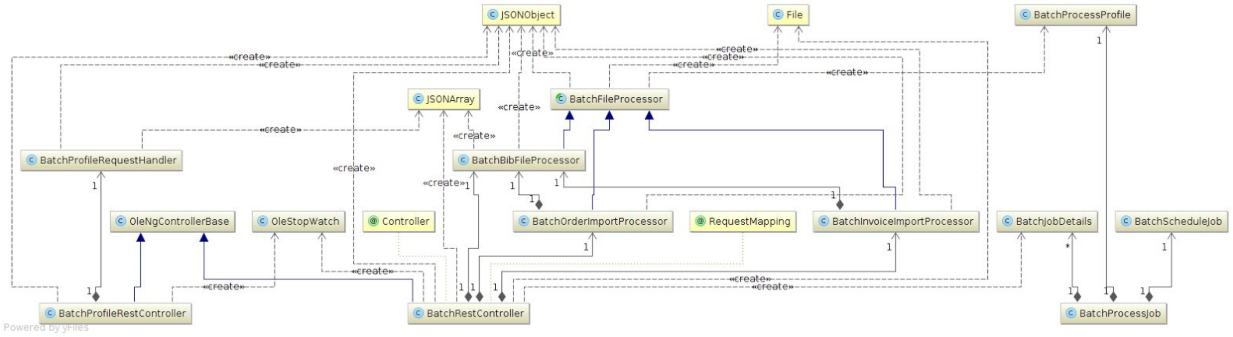
ole_ng_bat_job_details_t table contains the job information. The JOB_DETAIL_ID is the primary key and the JOB_ID is a foreign key referenced from the *ole_ng_bat_prcs_job_t* table. It contains information on when the job was started, completed, time spent, total records, status, etc. These tables work independent of the tables associated with the old batch process.



All tables with persistable data contains two properties by default – Version Number and Object Id. This is in order to take advantage of KRAD features. More information can be found in Rice Documentation here.

Last Published March 14, 2016

Service Interface Design (Java)



Batch Process Profile

The Batch Process Profile is where configurations as to how to handle the incoming files for a Batch process resides. Everything with respect to the batch process profile is handled by the BatchProfileRestController class. Most of the validation resides on the client side handled by AngularJS and the calls made are REST calls. The *submitProfile* method handles the saving of the BatchProcessProfile business object. The business object contains the various attributes of the Batch Process Profile. The *searchProfile, editProfile* and *deleteProfile* methods take care of searching, editing and deleting the batch process profile, respectively. They in turn call relevant methods in the handler class, BatchProfileRequestHandler class which extends the BatchProfileRequestHandlerUtil class. The new batch process allows importing of a batch process profile too. The *UploadFile* (sic) method is called for loading the batch process profile. It converts the raw content to JSONObject from which the profile is extracted and saved.

The new batch process helps in loading Bibliographic data, Purchase orders and Invoices through Bib Import, Order Record Import and Invoice Import, respectively.

The BatchRestController class is where everything with respect to Batch Process happens. The *processBatch* method is central to batch processing. The method calls the *processBatch* method of the BatchFileProcessor class. In here the batch process profile is retrieved and the raw Marc file is converted to a List using the *convertRawMarchToMarc* (sic) method of the MarcXMLConverter class. It then call an abstract method, *processRecords.*

Bib Import

The *processRecords* method of the BatchBibFileProcessor class is the one called for Bib import. A List of records converted from Marc, the BatchProcessProfile business object and the report directory name are passed as parameters. The Marc Record and the match points from the batch process profile are passed to the *prepareSolrQueryMapForMatchPoint* method of the MatchPointProcessor class. This method returns a solr query. The bib id is retrieved by using the solr query. The *prepareRequest* method is called with the parameters, bib id, Marc record

and the batch process profile. In the method, the marc record undergoes various modifications based on the match points and data mapping. The *prepareDataMapping* method and *handleBatchProfileTransformations* method help in the transformations.

Order Record Import

The *processRecords* method of BatchOrderImportProcessor class is the call made for order record import. A list of records converted from Marc through the *convertRawMarchToMarc* method, the BatchProcessProfile business object and the report directory name are the parameters. The bib import profile is retrieved using the *getBibImportProfile* method of the BatchOrderImportProcessor class. Following this the data values are built through the *buildDataValuesForBibInfo* method of the BibUtil class to return a map of bib information. Following this the *processBibImport* method of the BatchOrderImportProcessor class is called. This in turns calls the *processRecords* method of the BatchBibFileProcessor class and the whole bib import part is executed. Following this the *buildUnMatchedRecordsWithBibId* method of the BatchOrderImportProcessor class is called to build unmatched records. The *processOrder* method of the OrderRecordHandler class processes orders and returns PO ids.

Invoice Import

The *processRecords* method of the BatchInvoiceImportProcessor class is the call made for invoice import. The BatchProcessProfile is retrieved using the *getBibImportProfile* method of the BatchInvoiceImportProcessor class. The *processBibImport* method is called which calls the *processRecords* method of the BatchInvoiceImportProcessor class. This results in the whole bib import part being executed. Following this the *prepareInvoiceOrderRercordFromProfile* method prepares the invoice order records from the profile, the *getvalueByPriorityMapForDataMapping* method is used to arrive at the invoice record. The *processMatchpoint* method prepares a list of purchase order items. Finally, the *saveInvoiceDocument* method of the OleNGInvoiceServiceImpl class is called with the Ole invoice document as the parameter and the document is saved.

Service Interface Design (REST if applicable)

The new batch process and batch process profile APIs are exposed as REST APIs.

Batch Process Profile

Submit Profile (POST)

The service helps in submitting a profile.

Parameter	Value	Mandatory
Service	“submit”	Y

Last Published March 14, 2016

Profile content	<search content as JSON>	Y
Sample URL: http://localhost:8080/olefs/rest/describe/profile/submit		

Search Profiles (POST)

The service helps in searching a profile among all batch profiles in the system.

Parameter	**Value**	**Mandatory**
Service	“search”	Y
Profile content	<profile object as JSON>	Y
Sample URL: http://localhost:8080/olefs/rest/describe/profile/search		

Edit Profile (POST)

The service helps in editing a batch profile by retrieving the profile and editing the contents.

Parameter	**Value**	**Mandatory**
service	“edit”	Y
Profile content	<profile object as JSON>	Y
Sample URL: http://localhost:8080/olefs/rest/describe/profile/edit		

Delete Profile (POST)

The service helps in deleting the batch profile identified.

Parameter	**Value**	**Mandatory**
service	“delete”	Y
Profile content	<profile object as JSON>	Y
Sample URL: http://localhost:8080/olefs/rest/describe/profile/delete		

Upload File (POST)

The service helps in uploading a batch process profile as a multipart file.

Parameter	**Value**	**Mandatory**
service	“import”	Y

Batch Process Profile	<profile as a MultipartFile>	Y
Http Servlet Request		Y
Sample URL: http://localhost:8080/olefs/rest/describe/profile/delete		

Batch Process

Upload File (POST)

The service helps in uploading a batch process as a multipart file.

Parameter	**Value**	**Mandatory**
service	“upload”	Y
Batch Process	<job as a MultipartFile>	Y
Profile Name	<String>	Y
Batch Type	<String>	Y
Http Servlet Request		Y
Sample URL: http://localhost:8080/olefs/rest/describe/upload		

Submit (POST)

The service helps in submitting batch details for processing.

Parameter	**Value**	**Mandatory**
service	“/submit/api”	Y
Batch Details	<details as a MultipartFile>	Y
Sample URL: http://localhost:8080/olefs/rest/describe/submit/api		

Create Job (POST)

The service helps in creating a batch for processing.

Parameter	**Value**	**Mandatory**
service	“/job/create”	Y

Last Published March 14, 2016

Batch Details	<details as a MultipartFile>	Y
Sample URL: http://localhost:8080/olefs/rest/describe/job/create		

Quick Launch Job (POST)

The service helps in launching a batch job for processing.

Parameter	**Value**	**Mandatory**
service	“/job/quickLaunch”	Y
Job Id	<String>	Y
Batch Details	<details as a MultipartFile>	Y
Http Servlet Request		Y
Sample URL: http://localhost:8080/olefs/rest/describe/job/quickLaunch		

Schedule Job (POST)

The service helps in scheduling a batch job for processing.

Parameter	**Value**	**Mandatory**
service	“/job/schedule”	Y
Job Id	<String>	Y
Batch Details	<details as a MultipartFile>	Y
Schedule Job	<String>	Y
Http Servlet Request		Y
Sample URL: http://localhost:8080/olefs/rest/describe/job/schedule		

Destroy Job (GET)

The service helps in deleting a batch job that is scheduled for processing.

Parameter	**Value**	**Mandatory**
service	“/job/destroy”	Y
Job Id	<long>	Y

Last Published March 14, 2016

Sample URL: http://localhost:8080/olefs/rest/describe/job/destroy

Get Reports Files (GET)

The service helps in retrieving the reports files.

Parameter	Value	Mandatory
service	“job/getReportsFiles”	Y
Sample URL: http://localhost:8080/olefs/rest/describe/job/getReportsFiles		

Get Specific Reports Files (GET)

The service helps in retrieving specified reports files.

Parameter	Value	Mandatory
service	“job/getSpecificReportsFiles”	Y
Job Details Id	<long>	Y
Sample URL: http://localhost:8080/olefs/rest/describe/job/getSpecificReportsFiles		

User Interface Design

The new batch process screens use AngularJS for the user interface. The javascript based framework takes care of validating the data and handling requests and responses as a controller. The *batchProcess.html* and *batchProcessProfile.html* files provide the html content for the pages. Javascript files such as *oleng.js, batchProcess.js, olengProfileDropDownValues.js, batchViewController.js, batchProfileServiceHandler.js* and *batchProfileValidator.js* files contain methods that handle the various validation and controller tasks.

Data Importing

Both batch processes and batch process profiles can be imported using the REST services.

Data Exporting (if applicable)

OLE uses a RDBMS backend and hence any data can be exported using simple SQL queries.

Last Published March 14, 2016

Workflow

Workflow is not currently available.

System Parameters

Namespace Code	Parameter Name	Description
OLE-SELECT	MAX_NO_OF_THREAD_FOR_ORDER_IMPORT	This parameter is used to set the default number of threads that are to be used during order import.

Roles and Permissions

Roles and Permissions are currently not available.