The integration between Repox and the Sip Manager

The Repox and the Sip manager will coordinate their activity by excahnging data via a shared data store.

This requires the creation of a common data base schema for data and the definition of a synchronization protocol for operations.

DB Schema

In this paragraph the database schema is presented. The figure 1 shows a diagram of the schema built using the UML static diagram symbols: class symbols represent tables, class attributes represent fields, and associations represent relationships and their cardinality. Stereotypes above table names indicate the component owning the “write” permissions on the table, for instance the table Aggregator can be modified by Repox component while the table Uris can be modified by Sip component. Some tables are shared and can be modified by both.

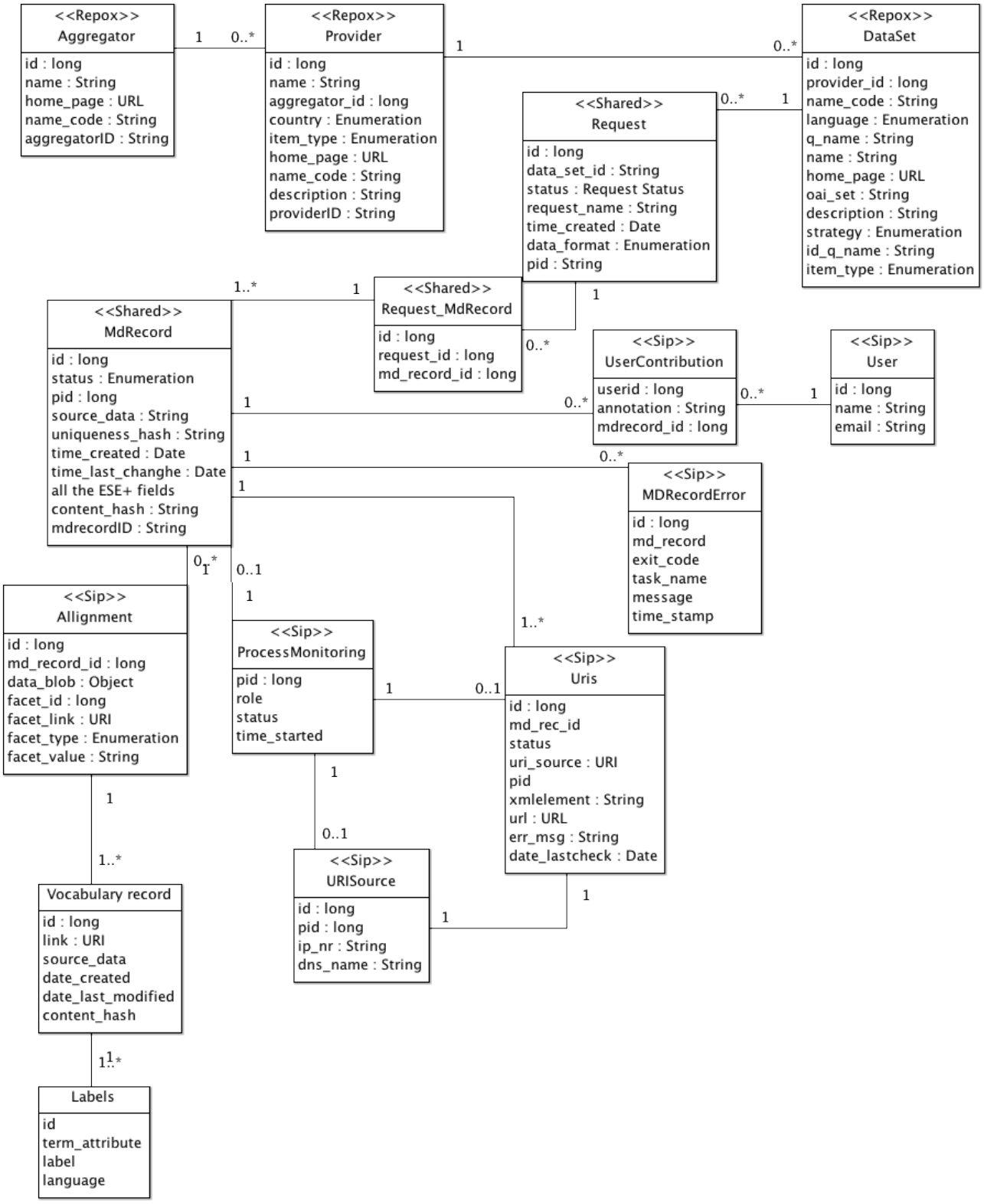


Figure DB Schema

Aggregator table
The field *name_code* is set by europeana currently something like "97". The field *name i*s a human readable name of the aggregator.
The field *aggregatorID* contains the id generated by the Repox.
Provider table

Country is two letter ISO code 3166-1-alpha2 (http://en.wikipedia.org/wiki/ISO_3166-1_alpha-2).
The field *name_code* is set by europeana currently something like "004"

The "Type" values (for the moment) can be:
Museum
Archive
Library
Audio Visual Archive
Aggregator

The field *description* contains a short description of the provider.
The field *providerID* contains the id generated by the Repox.

DataSet table

The value of the "Language" field is the java Locale code.
Only UTF-8 encoding for harvested files.
The value of the field "q_name" is the qualified name used by the provider to identify the root element of the metadata record.
The field *id_q_name* is the xpath expression identifying the record id. If The value is empty this means that the Repox will automatically generate the identifier.
The *name* is the name of the data set sent by provider (may be empty), the name_code is the name created by europeana (e.g. 03901_Ag_FR_MCC_joconde). The field *file_name* contains the original request as single file for traceability
The *description* is a Repox mandatory field and describes the data set; this field is used for OAI server.

The *strategy* field indicates the ingestion strategy adopted by the harvester for a specific data set. Possible values:

DataSourceDirectoryImporter
DataSourceOai
DataSourceZ3950

About *item_type*: currently ESE is the only accepted metadata format but in the future we almost certain extend the harvesting to other formats such as LIDO.

Request table
This table identifies a specific harvesting for a given data set. It also indicates if this request can be sent to production.
When Repox is parsing the file the request should be in the state "under construction", and Repox may abort the request and set the status accordingly, when this process is done the "status" value must be changed in "import completed" and after this Repox cannot change the data.

The field "status" can assume the following values:
under construction - repox is creating a new request
import completed - repox ready, sip can take control when ready
aborted - something went wrong

sip processing - sip has found the request repox may not any more delete the request
pending validation sign off - all records for this request completed
pending AIP sign off
creating AIP
AIP completed

RequestMDRecord table
This table links all the metadata records belonging to a given request.
Repox can only insert records in this table whose request status is "under construction", if the request is aborted don't remove links from this table, this task will be done manually.
This table is needed because we want to maintain history of requests for the same data set.

MDRecord table
This table contains the original record and all its refinements. The value of the field "ContentHash" will be provided by the Harvester and is used to identify the ESE record.
The Repox may only insert new MDRecord and cannot change or delete existing items.
The two fields that must be actually filled by Repox are "Source_data" with the delivered content dump and "content_hash", all other fields should have the default values.
The algorithm for generating content_hash is: sha256 hash with all the linefeeds stripped.
The field "status" can assume the following values:

created – (default) the record is created but not yet processed in any way
idle - nobody is touching it, waiting for more checks
processing - a checker is working on this record
problematic - something went wrong, human intervention might save the record
broken - record is invalid, some check decided this ese is not acceptable
verified - all checks succeeded, could be sent to production

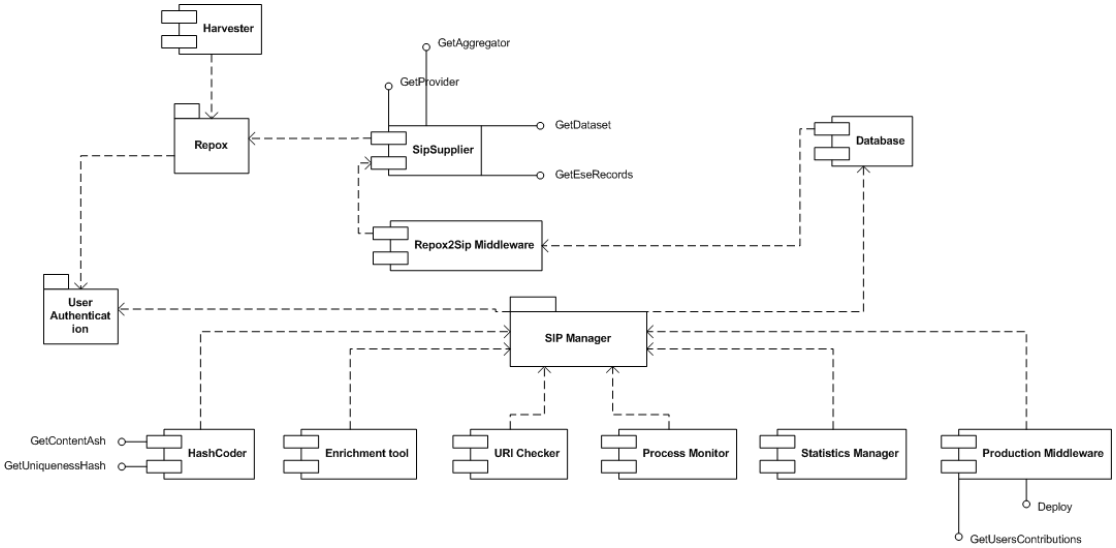
The *mdrecord_id* is an identifier used by Repox: when no id xpath is provided it is automatically generated by Repox, otherwise it is extracted using the Xpath defined in the *id_q_name* field of the Data_set table.

URIS table

A Uri record is created by a thread and end as completed or failed

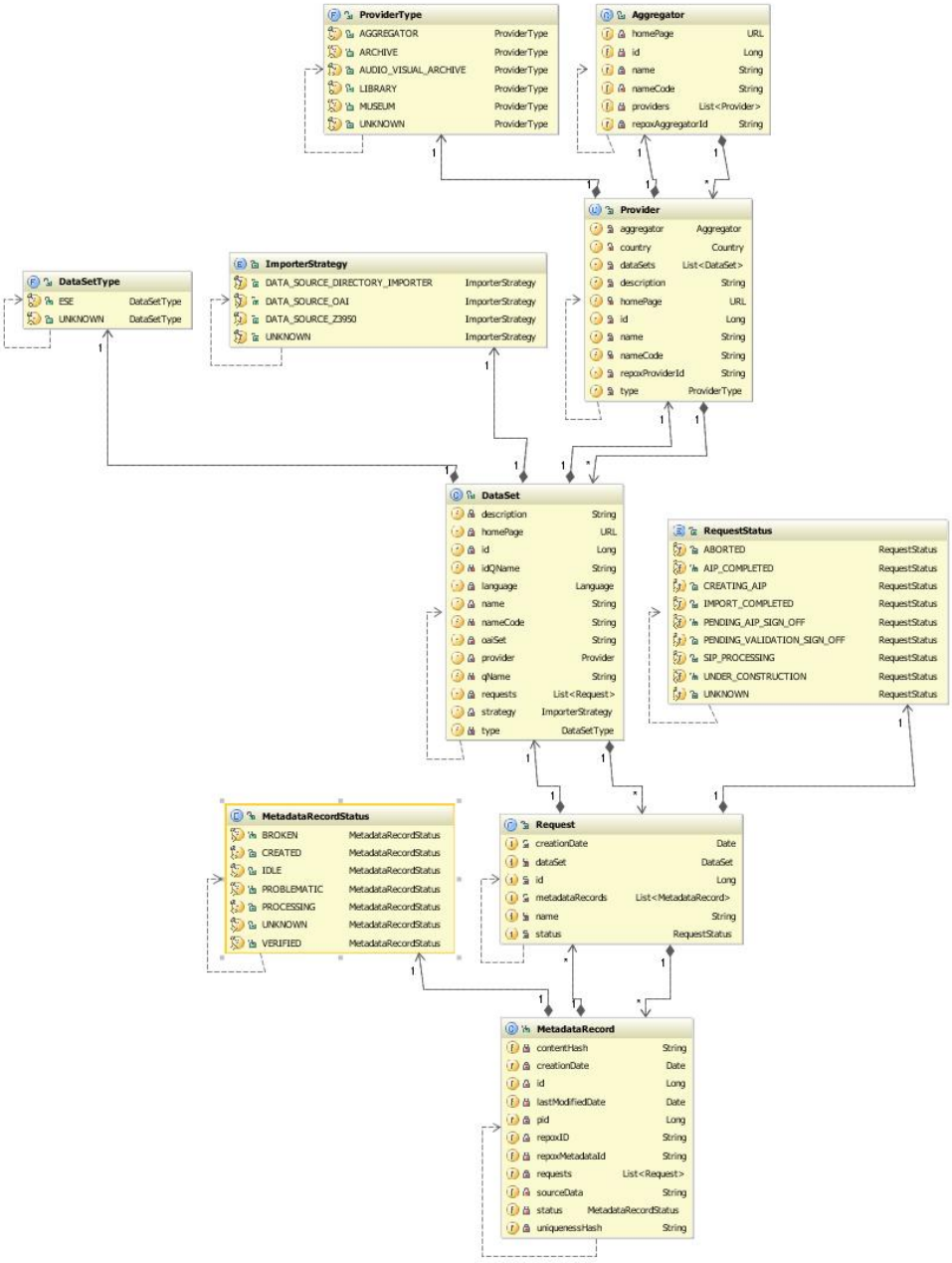
The field status can assume the following values
created - (1) the record is just created not processed
uri verified - (2) the uri responds and returns an OK
object downloaded - (3)
full_doc generated - (4)
brief_doc generated - (5)
completed - (6)
failed - (7) something went wrong, see err_msg for details in this msg also should be logged what step failed

Component diagram (in progress)



The diagram in the figure is in progress and should be used to define the overall integration architecture. Components should be detailed and checked. The goal of the diagram is to have a visual representation of components in order to discuss the detailed architecture. Probably what in the figure appears as interfaces could become methods of a single interface, and some components would be classes of a single component. At the end of this activity a correct UML component diagram will be produced.

Sip2Repox component class diagram



Repox activity

In the figure 2 the UML state diagram is used to describe the overall activity of the Repox component.

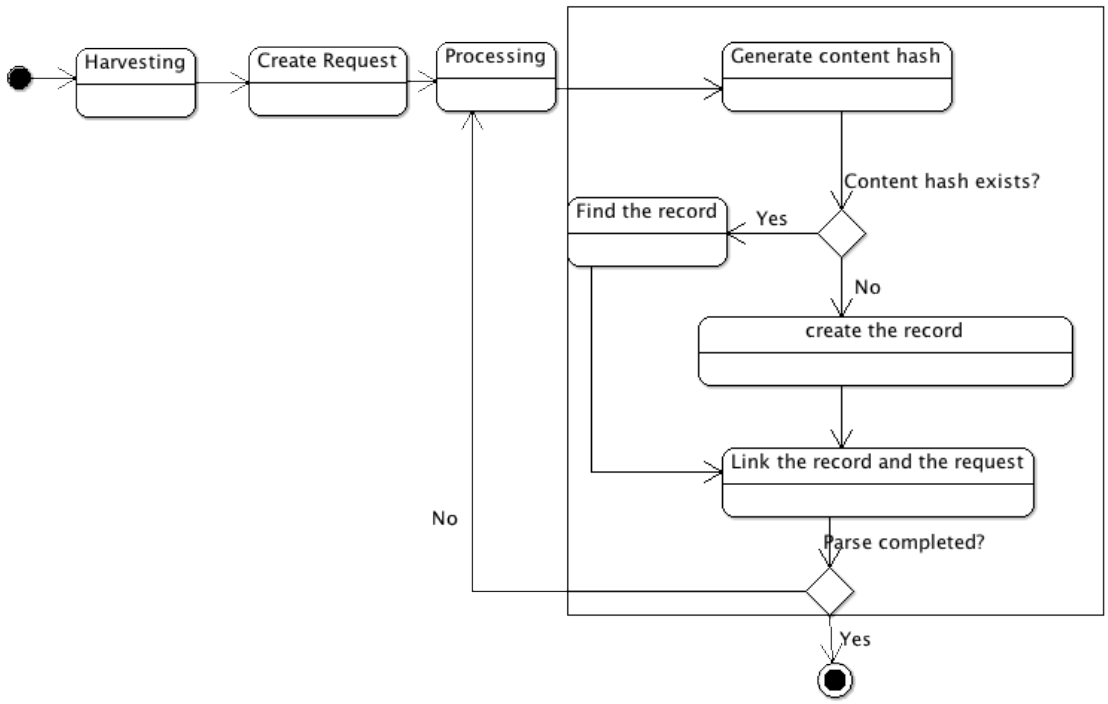


Figure Repox Statechart

As "Processing" it is intended the phase of extracting records from the harvested Data set. If the harvesting is made using OAI-PMH, records downloaded from data providers must be stored in the DB and in the file system for logging purposes. When the Data Set is downloaded via file transfer (FTP, email attachment etc) the processing phase will extract records.

For each ESE record Repox should:

a. Generate content-hash
b. if item identified by content-hash is not found in the database create a new record in the table MDRecord and fill it with content-hash and xml dump of ese record
c. if an item with the same content hash value is present get its id
d. Link the record to the request by adding a record in the table Request_Md_Record