Media structure modifications in Invenio

Student : BOULMIER Anthony
Supervisor : Jean-Yves Le Meur

Summer 2012

Table of contents

Introduction

As part of my internship at CERN, I had the task to carrying out a project to modify the structure of storage media (photos) in the software Invenio and solve problems linked theses changes.

The purpose of this project was primarily to participate in the development of the Invenio software and improve my knowledge in managing and participating in projects of large size.

In the remainder of this paper, I will describe the project and the methodology that allowed me to solve the problem.

Project Analysis

MARC the standard

MARC is a standard used to trade bibliographic data in a system, like a library.

This standard is very used by librarian, because it allows a lot of field, very useful, to have a good description of a media.

MARC is used in the Invenio system to trade all the bibliographic data with the user and the database. This format is also converted to xml, latex and JSON.

Why would we change the structure ?

In the following pictures, each square is a way to see a record in the database.

A record is a way to store a picture or an album in the database.

First media structure (Pictures)

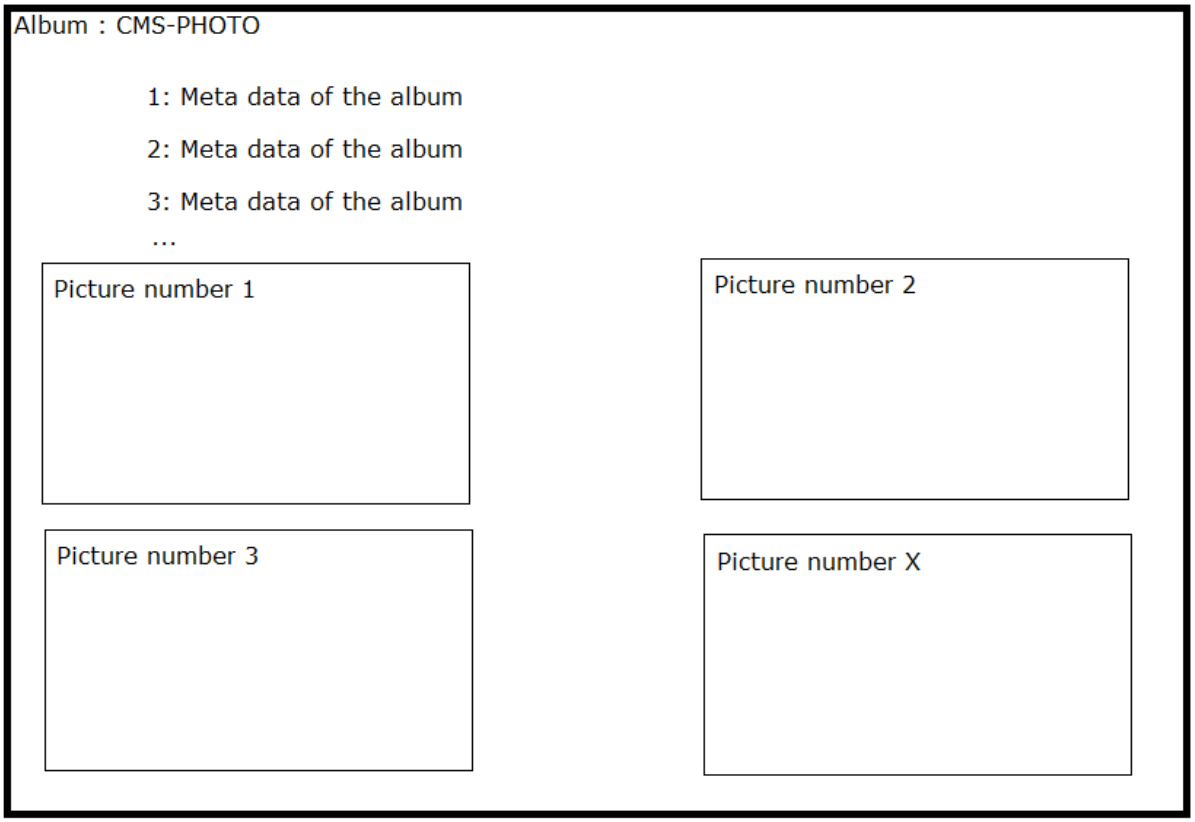


Figure 1 : First media structure

As you can see, an album can contains some metadata about only the album, the pictures doesn't have any meta data.

It can be a problem when there is two different photograph in the same album per example.

The main goal of the structure modification is to allow pictures to have meta data.

The new media structure (Pictures)

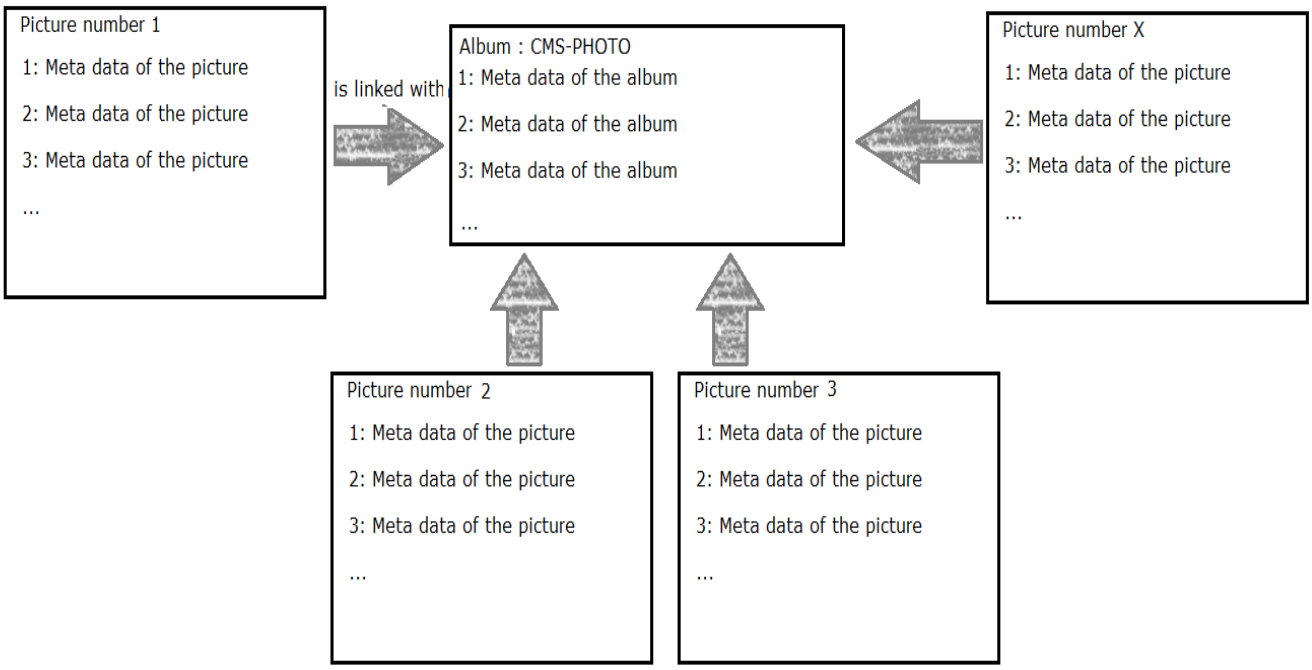


Figure 2 : The new media structure

The new structure allows meta data for the album and for the pictures.

To arrive to this structure we need to split the existing records and restore it in the database with the new structure with a script which I had to develop.

What will be changed ?

When the structure will be updated in the system, some changes must be done to allow the system to understand the new structure.

Per example, when the software displays the content of a record in the website, he needs to understand the new structure to display also the content of the subrecords (Subrecords are the pictures linked with the album record), otherwise there going to be a problem for the user or the system.

- The data research in a record. We need to find some data in the subrecord because it won't be in the album record anymore.
- The indexing of the keywords of a record have to be careful because when we want to search a record by typing a keyword of a subrecord he need to understand the structure.
- The research in the website.
- The way to add a new record in the system, must be improved to allow the understanding of some new tags and to keep the new structure during the time.

How to keep the link with host and subrecord

With this new structure, we need to keep the link between the host and the sub records.

With the MARC21 we can add a field (datafield) and a subfield with code and tag to display the record ID of the host record.

With this approach, we can easily find all the subrecords in the database and merge each subrecord with their host record. It is a good way to be very generic.

The tags

We choosed the tag number "773" to keep the link with the host record and the code "a".

Example of link with host record:

```
"
<datafield tag="773" ind1="" ind2="">
     <subfieldcode="a">1547821<subfield>
</datafield>
"
```

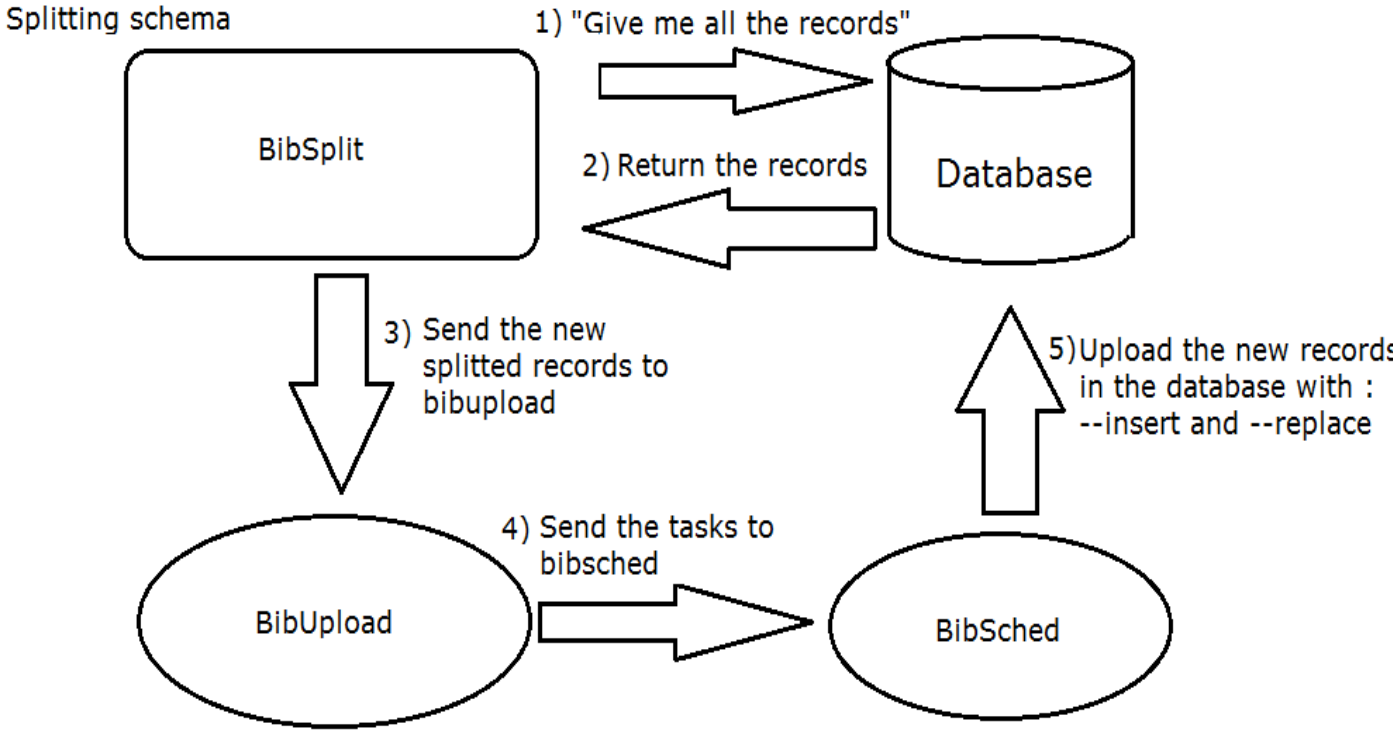
Methodology

How to change the structure

To change the structure we need to get all the record contained in the database and convert it into the new structure by a specific library created for this work.

The new library is named BibSplit, it is called one time to convert each record into a collection of host records and subrecords. BibSplit contains also a config file where each code, tag and specific field are describe to be called by any other library in the system.

We can see the splitting system like this :



Display the new record structure

The second step was to allow the display system to understand the new structure.

I created a library named BibAlbum to have a good representation of a host and his subrecord and permit some actions like get all the BibDocFile or all the url of an album, with some methods and properties.

This library is used during the display.

Format field

Invenio works with a format field (number 980) used to specify the type of display that we want with this kind of record.

Per example, for a host record the format type ALBUM was add in a config file and in all the host records.

With this approach, we can display records in different ways.

Display link

To keep a virtual structure during the display a link that allow the users to come back to the host record when they see a subrecord to have a view of the whole album.

Conclusion

This project was a great opportunity for me to get more experiences in large project management and improve my competences in Python development.

It was a project with many dependencies, so it was a good amount of analysis and reflexions about the consequences of each actions that I am doing on each part of the project.

The new structure is a good way to make the Invenio system a bit more generic, maintainable and more extendable.

I hope that the BibAlbum library will grow and it will be improved, because I think that it is a good way to make a part of the system more easy to understand.

The next step for the future projects will be to upgrade the export library to understand the actual structure and then to make the possibility to export the records with different subrecord.