Refactoring a library's legacy catalog: a case study

Joachim Ganseman
Library, Koninklijk Conservatorium Brussel,
School of Arts, Erasmus University College
Brussels, Belgium
Email: joachim.ganseman@ehb.be

***Abstract*—It is generally deemed safe practice to not change a working IT system, but at some point, driven by new advances in computing, an upgrade cannot be avoided anymore. This paper tells the story of a custom configured, pre-MARC catalog from the early 1990s. Well-tailored to the needs of a music library, it proved solid and had been in use without changes ever since, amassing almost 60,000 manually entered records by the end of 2014. With full MS-DOS support disappearing from recent OSes, it urgently needed to be ported to a modern web-accessible OPAC. This confronted us with several challenges: finding a robust way to convert records without losing information, dealing with corrupt or erroneous data entered in several languages, deciding on the functionality of a new catalog and lending library management system, etc. All while not losing sight of the road ahead: eventually, we want to be able to link the catalog to sizeable repositories of digitized scores and audio recordings that have been collected internally over the years, such that it could grow out to effectively fulfill a bridge function between scores and performances. The database conversion process is presented here, paying special attention to the specific requirements of a music research library and correction of encountered errors. This is a useful case study for anyone who ever needs to reintegrate a legacy digital catalog into a modern library system.**

I. Introduction

The library of the Royal Conservatories in Brussels has amassed approximately 1 million volumes in the almost 200 years of its existence. A significant part of it is of venerable age: at the moment of writing, the RISM database has 16415 entries under its siglum *B-Bc*. Up to this day, a traditional paper index card catalog is still often used for day-to-day search and retrieval, as it is the most complete index of the collection that is available.

In 1993 the library sought to start its first digital catalog. A license for the *Allegro-C* library database system [1], originally developed by the University Library of TU Braunschweig, was bought. The system was initially configured for the Lemmens Institute of Leuven and adopted by the Brussels Conservatories' library. In the second half of 1994 the system was rolled out. Some configuration changes were made around 1998, but no version upgrade was ever attempted. The need to update to a modern Integrated Library System became really pressing.

The exercise to import the old catalog into a modern Integrated Library System is not simple. Over 20 years, over 50 collaborators have entered data in the catalog, without much input validation, authority control or correction policies. While there was always a strict set of input conventions, mostly based on ISBD rules [11], different stylistic habits can be observed in records entered by different contributors. Over the years, some input conventions were changed, while older records remained unchanged. As a result, almost half of the present records contained errors, and a major cleanup operation was needed.

In this paper we'll look mostly at the kind of errors that were discovered, and efficient ways to fix them. For conversion of the internal data format towards MARC21 [9], dedicated software was written in which some of these correction rules were hard-coded [2]. Until it has been decided which library system will be adopted in the future, we use the Koha Integrated Library System [3] as migration target to test the conversion routine.

II. Background

At the time the decision to start a digital catalog was made, some alternatives were on the table. The main potential alternative was VUBIS. Originally developed at Vrije Universiteit Brussel and later commercialized jointly with TU Eindhoven, the company is currently called Infor and their product line V-smart [4].

In 1993, VUBIS was deemed too expensive and not flexible enough for the needs of a conservatory library. A license for Allegro-C was two orders of magnitude cheaper, and Allegro-C offered some features that were ahead of their time: notably powerful full-text indexing, and exceptional content-independence. It could be configured to store any kind of ASCII data in any number of fields.

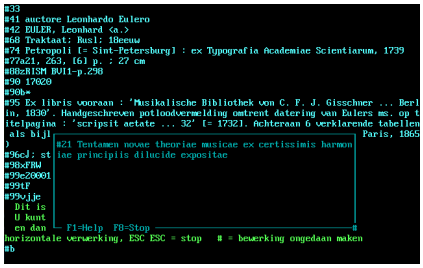
The choice was therefore quickly made, but was initially intended as only a temporary solution. The plan was to use Allegro-C only until dedicated ILS software (like VUBIS) was configurable enough to easily accomodate the needs of a music library. By 2003, most other users of Allegro-C (the Lemmens Institute of Leuven and Conservatories of Antwerp and Ghent) had integrated their catalogs with the library systems of their associated university colleges or universities. In Brussels, this step was never set [10].

The Allegro-C database, version 14, is a set of 16-bit programs written for MS-DOS. As Microsoft's Windows XP is the last OS to natively support 16-bit MS-DOS programs, and official support for Windows XP was ended on April 8, 2014, a migration to a more recent system could not be postponed any longer. First of all, we needed urgent solutions to enable us to continue working on the old database with newer OSes, while a new database was being researched.

The NTVDM (an MS-DOS Virtual Machine) in Microsoft Windows 7 can run Allegro-C v14, but unfortunately does not have full support for all keys on the AZERTY keyboards used

(a) Opening screen

(b) Record display



(c) Record editor

Fig. 1: A few screenshots of the Allegro-C v14 database system

in Belgium. To remedy this, an AZERTY keyboard driver for the Windows 7 NTVDM MS-DOS was written [5], by adapting a driver for a German QWERTZ keyboards. This temporary solution enabled the library staff to safely use the old database on currently supported OSes. Still, the NTVDM is not without problems - e.g. the software does not run in full screen anymore.

For even more recent computers (Microsoft Windows 8) and other OSes (Linux, Mac OS X), using DOSBox [6] was attempted. DOSBox' preferences file was extended with a small script to mount the database's program and data directories, to run at maximum speed, and to start Allegro-C automatically at the start of DOSBox. However, we encountered difficulties in writing to the data files on the local network drive. Some of the characters with diacritical marks were also not displayed correctly. We suspect this has to do with erroneous handling of the encoding somewhere. To avoid corruption of the data files, we only use this setup at workstations where read-only access to the database is sufficient.

Our old version of Allegro-C (v.14) is not used as OPAC, has no enforced input validation, does not use any authority records or authorized lists, and does not have Z39.50 or similar functionalities to link to outside library catalogs. Recent versions of Allegro-C have all these functionalities [7] However, mostly since its documentation is only available in German and it was unclear how easy it would be to translate the interfaces to Dutch and French (for use in Belgium), we decided to explore the migration to a web-based alternative first, and picked the open source Koha project [3] as initial migration target, mostly because of its very complete suite of functionality.

III. Accessing and Exporting Data

Data can be extracted by from Allegro-C by writing an export parameter file which lists the fields that need to be exported and defines the formatting details of the output (tabs, line delimiters, etc). This allows us to extract all data in a CSV (comma-separated values) format. The text encoding used in the database (and the resulting file) is Code Page 850 (in some editors known as IBM850 or MS-DOS Latin-1). Awareness of this encoding is important in order to preserve diacritical marks. If a new database system uses ANSI or UTF-8 encoding, at some point a conversion needs to be made. Advanced plaintext editors (jEdit, Notepad++, ...) have the ability to do this.

The export file prints out all relevant fields, separated by a tab, effectively creating a CSV file. The fields used in the database are defined in a configuration file (`*.cfg`). The export file (`*.apr`) looks like this (it was adapted from a similar one with comments in German), with `#t2` indicating a tab:

```
[...]
   -------- Konstanten
zl=0    Zeilenlnge unbegrenzt
ks=4    Beginn beim ersten Zeichen hinter #
ke=""   Kategorie-Ende = Code 0
as=h0   Aufnahme-Start: Hierarchiekennung + 0
ae=13 10   Aufnahme-Ende: CR / LF
dx=1    beim Testen Steuerzeichen auf Bild
1=13 10
2=" "
3=": "
4="; "
5=""
   -------- Anweisungsteil
#99n
#t2
#99t
#t2
#00
#t2
#90
#t2
#21
#t2
#41
[...]
```

Here we experience a first, unfortunately unpreventable, loss of data. During manual input or later corrections, the field number sometimes needed to be typed explicitly. Typing errors at that point then lead to data entered in a field that is not part of the agreed configuration (e.g., data in an non-existing field `#211` instead of the correct `#21`). These 'ghost' fields (we cannot see their existence in the database client) cannot be exported. Also, if any field contained a tab character, its exported record did not line up correctly with the columns. Approximately 10 such records were detected that needed manual correction first.

A. Common errors in the database

The resulting CSV file is, after conversion of the character set to UTF-8, easily imported into a spreadsheet program. Through sorting and filtering we can get a first rough idea of common errors in the database, especially in those fields that were supposed to only have a

limited number of possibilities (like, a checkmark to indicate whether an item can be borrowed or not).

After running a database for 20 years without enforced input validation, the list of potential errors is extensive. In September 2014, with the database containing approximately 58000 valid records, we found the following errors during a first pass-through (excluding any errors in capitalization, titles or names):

- 12 records contained an invalid (non-numeric) character in the *creation date* or *correction date* fields. [1]
- 394 records had a missing *creation date*. [2]
- 212 records contained no *record creator* code. [3]
- At least 60 records contained visibly missing or erroneous data on *instrumentation*. [4]
- 95 records contained *language* codes in the *instrumentation* field. [5]
- 182 records contained *tonality* info in the *instrumentation* field. [6]
- 68 records contained *language* data in the *tonality* field. [7]
- 4 records contained *tonality* data in the *language* field. [8]
- 4 records contained an *opus* number in the *tonality* field.
- 39 records contained *tonality* errors, ranging from not adhering to conventions (e.g. writing h-moll instead of b-minor) to typos (e.g. a sonata in p-major).
- 28 records contained errors in the *language* field, mostly misspellings.
- 85 records contained errors in the *ISBN* field. 45 of those were superfluous mentions of ISBN-10 or ISBN-13. 21 books turned out to have an invalid ISBN-number printed inside.
- 165 records contained an *ISSN* number in the *ISBN* field. [9]
- 79 records had *comment* data in the *medium* field.
- 59 records had general typing errors in the *medium* field
- 150 records contained only a number in the *material specification* field without further specification (this field encodes the extent of the item: nr of pages, dimensions etc).
- Approximately 450 records contained no status code (used to indicate the type of a record).
- At least 16 records contained a shortcut command to edit or delete a field, as textual data in a field instead. [10]
- ...

Finally, it is internal convention to use ISBD rules to encode lists of items (separating them with semicolons etc.). Quite a few records had errors in ISBD interpunction: we found 84 erroneous *tonality* fields, 264 erroneous *language* fields, 86 erroneous *ISBN* fields, 81 erroneous *medium* fields, 380 erroneous *material specification* fields etc.

Note that these were just errors that were in plain sight at the very beginning of the project. Most of these were manually fixed by library staff. More rigourous record validation and correction, based on a more detailed parsing of contents, was developed in the software created to convert the exported CSV file MARC21 [2].

IV. Conversion to MARC21

The data structure of the old Allegro-C database is MARC-like, but it's not MARC21 - since MARC21 was only published in 1999 as a unified standard derived from its predecessors USMARC and CAN/MARC, we call Allegro-C *pre-MARC* here. Allegro-C encodes any kind of data as plain text in a field with a 3-character code of choice. To convert the data to MARC21 (or Unimarc), a mapping needs to be made from the current Allegro-C configuration to MARC21 fields and subfields. Sometimes several fields in Allegro-C contain information that needs to be aggregated into a single MARC21 field. But more often, MARC21 is more detailed, and the info currently in one Allegro-C field needs to be parsed and divided among many MARC21 subfields.

We chose to write some dedicated software to assist in this conversion [2], enabling us to apply different validation and correction rules to each field. The software accepts a CSV file with the current records as input - making it also useful to integrate some separate catalogs for library subcollections that had been created by a few researchers in Microsoft Excel. It also requires a small configuration file that defines the mapping from columns in that CSV file to MARC21 fields and subfields. Internally, each field is a proper class with its own validation methods (to check for errors) and print methods (to output). The entire routine prints to a mnemonic MARC21 representation, which in turn can be compiled into machine-readable MARC21 with tools such as MarcEdit [8].

Whenever a new field of the old data is processed, its information is passed to the *update()* function of possibly several MARC21 Field objects in the corresponding MARC21 Record being built up. These *update()* functions contain the detailed parsing and storage routines, and also need to take into account potentially previously entered information which might be overwritten. Especially when information from several CSV columns is merged into one MARC21 field, the order in which the fields are processed may be important. In our local mapping file, we usually load the column containing the most complete information first as "default data", only to be later appended, or possibly overwritten, with additional info from other columns as necessary. The Field classes can be subclassed and its methods can be overridden to define different validation and correction procedures, should the software ever needed to be adapted for another project.

In what follows, we give a short overview of important fields in our catalog, the errors we encountered in them and how they were processed. We concentrate on bibliographic MARC records, augmented with a bit of holdings data.

A. Header information

The MARC21 leader field that we create for every record is mostly filled with default values - we assume that all records are complete (full level) and new. The bibiographic level defaults to monographs/items. If an ISSN is present somewhere, this is later overwritten to be a serial.

More difficult is the processing of the item type, since it was stored in more detail in Allegro-C's *material* field than the MARC21

[1] Since these numbers are automatically generated, these errors must have been introduced during a manual correction of the record.

[2] Idem. Also, the amount of missing data may indicate this was the result of a faulty macro applied to a batch of records once, or a maybe a software bug in the save routine.

[3] Each contributor to the database is supposed to enter their personal code in each record they add.

[4] The encoding system for instrumentation being very complex, this field was later found to contain hundreds of more erroneous entries.

[5] This is a legacy from the earliest database configuration, when language, tonality and instrumentation were still recorded in the same field. This was later split up in separate fields, but the records encoded in the old configuration were only partially, and sometimes erroneously, updated.

[6] idem

[7] idem

[8] idem

[9] This too is a legacy from the first configuration, when ISBN, ISSN and other standard numbers were encoded in the same field.

[10] Result of not having pressed enter after the previous edit command, combined with not checking the final result.

header allows for. As a general rule, we detect whether the *material* field contains `text, cd, dvd, vinyl, score...` to define whether this record should receive the code for a textbook, a score, or a recording. If additionally it contains the term `manus*` (in any language), it is updated to be a manuscript.

Problems arise with records that describe an item containing different types, like a textbook with a CD or DVD attached, or a vinyl record with a booklet. As we can only give one high-level code here, an order of precedence was defined as follows, from most to least important (MARC21 header code in []):

- Music: [c] (if manuscript: [d])
- Text: [a] (if manuscript: [t])
- Recording (CD, LP, cassette, tape, vinyl): [j]
- Film (VHS, DVD, photographic slides): [g]
- Pictures (photos, drawings, paintings): [k]
- Machine-readable items (computer files, microfilm): [m] [11]
- 3D Items (kits, statues, realia): [r]

Some MARC21 types remain unused. They are:

- $[e]$ and $[f]$ for cartographic material, as we do not have any.
- $[i]$ for non-music recordings. As it would be too complicated to automatically decide from its description whether a CD contains speech or music, and since we convert a music catalog, we default to musical audio ($[j]$) for all recordings.
- $[o]$ and $[p]$ for kits and sets of mixed materials: these are rare in the collection. We default to 3D artefacts [r] since that can just as well describe kits and boxes of mixed materials.

A record encompassing items of several types will receive the code for the most important item according to this prioritization. This results in textbooks with accompanying CDs to be classified as books, but also that CDs with described booklets risk to be classified as books. The latter is however less common in the library collection, and also easier to detect as erroneous afterwards.

A distinction between CD, LP (vinyl) and tape is made on the level of item type definitions in the Koha catalog. Based on the callnumber which is formatted in a particular way, these items with the same header code receive an internal Koha Item Type code to distinguish them in the OPAC. Similarly, archival material is split off from regular textual materials and marked as archive. This is a feature of the migration target, makin that search can be refined based on more item types than recorded in the header.

B. Callnumbers and inventory numbers

Items have both a location number (callnumber) and an inventory number. Usually, these are equal and unique, yet 620 callnumbers in the database occurred multiple times. Often this were items of which digital scans had been made, which then had been entered in a separate record bearing the same callnumber. Others were series of books, in which the separate volumes had not received individual callnumbers. In about 50 items the inventory number or callnumber needed to be updated since it had been either entered erroneously or copied from a previously entered record without being updated.

We envision a new Integrated Library System to rely on unique callnumbers to properly organize automated borrowing and return of items, effectively equating the callnumber to the barcode. As a result, all callnumbers needed to be made unique. Many were fixed manually. Those that remain, will automatically get an additional number appended to the callnumber in order to make them unique.

[11] Although the MARC21 specification says microfilms should get the same code as the reproduced item, regarding it as machine-readable corresponds better with current library policy, as this would group all scanned items together.

C. Authors

Authors had been encoded as one semicolon-separated list of names in standard form, with eventually a function and one or more dates appended. These were split up between MARC21 fields `100` and `700`, with subfield `$a` used to store the names, `$d` for the dates and `$e` for the functions. These last *relator terms* has always been limited to 33 valid functions (composer, author, editor, ...), abbreviated in Dutch.

The old database did not enforce authority control. There was a system to copy data from existing records, but few catalographers ever used it, since it involved typing a long list of keyboard shortcuts. As a result, many names exist in different spellings. About 10 different renderings of C.P.E. Bach could be found, depending on whether the initials had dots between them or not, or whether the first names were fully written out or not (and correctly spelled or not). The resulting messy author index caused major problems in search and retrieval.

To normalize this data, a system of authority control needs to be enforced in the new database, yet since there is a lot of cleaning up to do, this is a longer term endeavour. In the meantime, we have extracted a list of over 15000 unique names in the old database, to be checked against correctness - unfortunately a mostly manual work. In the final conversion, this should allow us to replace erroneously spelled names by the correct version.

Most of the other errors encountered here were typing mistakes: no closing brackets, the wrong kind of brackets (mixing up `()`, `[]`, `<>`), etc. Several hundred records contained incorrect relator terms. Most of these were spelling mistakes, sometimes the French version of a term was used instead of the Dutch terminology in this field. Many of these could be corrected by simple detection and replacement in the conversion software.

Whether we will convert from the old list of relator codes to the more extensive *MARC Code List for Relators* is still to be decided. We also hope at some point to link our list of names semi-automatically to an external Authority File like VIAF.

D. Music-specific data

For a music library, being able to search based on musical attributes is of major importance. One of the major strengths of the old Allegro-C database was that it had been configured to record musically meaningful information in great detail. One could say that at the time this was pioneering. Consequently, we wish to preserve as much as possible of this information as we can.

1) Instrumentation: The encoding system for instrumentation in the old database was pretty complicated, with many rules and exceptions. We discover errors in almost half of all records containing instrumentation info, if we parse the contents of the instrumentation field strictly. Roughly speaking, the system is as follows:

- First describe the total number of performers, *N* if unknown.
- After a colon: one or more of 13 possible sections (voices, strings, woodwinds, orchestra, ...), in a fixed order, with the number of performers in each section attached to it.
- In between <> brackets: for each previously mentioned section a subdivision per instrument, in the same order.

A string quartet is thus encoded as *4: str4* ⟨*vi2 av1 vc1*⟩. For a chamber orchestra, this becomes quickly more complicated. As an exception, soloists are mentioned separately. Initially, we've decided to detect the following errors:

- erroneous or not mentioned number of performers
- incorrect section abbreviations or section ordering
- incorrect interpunction

and we postponed checking for instrument abbreviations, ordering, and whether all the numbers actually add up. [12]

In MARC21, the (RDA-inpired [13]) instrumentation field `382` has several subfields that allow to split up the information that was encoded as one text string the old database: `$s` for the total number of performers, `$b` for soloists, `$a` for other instruments, `$n` for the numbers of these instruments. [13]

The main question remaining is what vocabulary to use. There exists a vocabulary in MARC21 containing 99 2-letter items, the *MARC Instruments and Voices code list*. However, it lacks sufficient granularity (e.g. distinguishing cantus firmus, viola di bordone, SATB saxophone types) for use in a specialized music library. Alternatively, the current specification of UNIMARC field 146, managed by IAML, contains a list of 281 unique 3-letter codes for 655 instruments [12]. However, UNIMARC's use of several possible suffixes to these codes does not contribute to readability. Finally, in 2014 the US Library of Congress launched its own list with over 800 instrument names, the LC Medium of Performance Thesaurus (LCMPT).

Whatever standard is adopted, this would require a translation of all old self-defined instrument codes to their more standardized counterparts. A final decision on this matter is still to be taken, as first getting the errors out of the existing data is a priority. In the meantime, the old database vocabulary is still in use, and just copied without change into the `$a` subfield.

2) Tonality: Regarding tonality (MARC21 field `384`), there are also several encoding options: one could adopt the Plaine-and-Easie code as used in MARC21 field `031` for musical incipits, write it out fully as it appears on the item, or adopt the list of codes from UNIMARC field 128. The system from the old database corresponds closely to UNIMARC. Just like with instrumentation, the readability can be a consideration: providing a controlled list of tonalities fully written out is certainly an option.

As for errors in the existing data, simple typos were most commonly encountered. Some encoders had recorded "H" as tonality if that appeared in German-language editions, even though it is policy to transcribe this to the equivalent "B". Typos in the names of modes also occurred a few times. Sometimes this field had a number in it, e.g. when the encoder mixed it up with the field code for opus number. Since the possible values in this field are limited, all of these errors were easily fixed. Also, only 608 records had tonality information actually encoded separately.

3) Opus: Converting the opus numbers (MARC21 field `383`) posed some specific challenges. Subfield `$a` encodes a general numeric designation; in the old database this was meant to be encoded as part of the title. This policy had however not always been adhered to, especially by encoders that were initially unfamiliar with music. Therefore, in the new database field `383$a` remains unused, and whenever the string *"nr"* was encountered in the old data, the conversion routine would spit out a warning. At the same time, the old database's field for opus numbers contained just a text string, while in the new database we want to split it up into 3 useful subfields: `$b` for opus numbers, `$d` for thematic catalog names and `$c` for thematic catalog numbers.

[12]Checking whether numbers add up is a useful validation tool, but should be able to handle some exceptions in this particular case. When multiple performers play four-handed piano, it is encoded in the old database as *2: toets1* ⟨*pf1*⟩ (*toets* being the category for keyboard instruments, *pf* meaning pianoforte).

[13]The choice for field `382` over field `048` is inspired by the recommendations that can be found in [13].

Opus numbers itself could also be encoded in a variety of ways: with prefixes *"op"*, *"opus"*, *"op."*, *"op-"* and variations thereof. Some encoders insisted on writing *"oeuvre"*, or *"opera"* in the plural case. With the necessary find-and-replace operations, this was a much as possible normalized such that only one prefix (*"op."* followed by a space) remained. Everything that followed and started with a number or with *"anh."* (for *Anhang*), was taken to be the actual opus number.

Since the same text string could also contain thematic catalog numbers, an order of processing was defined: first lifting out anything that looked like an opus number as described above, then treat whatever is remaining as a thematic catalogue. The names of the thematic catalog were not always consistently capitalized, and sometimes incorrectly spelled. It would take quite some time to hardcode checks on catalog names in the conversion routine, while only about 920 records had opus information encoded in them. Manually browsing all catalog abbreviations and correcting any errors encountered was the quickest approach to correct the data here. There might be time to implement more intricate checks (like checking that no Mozart item has a BWV number encoded with it) in the future.

E. Publisher information

The old database had publisher information stored in 2 different ways: either the publisher's name, location, year were encoded into 1 field, or otherwise into several other fields (the addition of separate fields for this information had only been done in 1998, after the database had been in use for 4 years). Whenever this info was encoded in several fields, it gets copied to the relevant MARC21 field `260` subfields: `$a, $b, $c` for the publisher and `$e, $f, $g` for the printer.

To accomodate for the legacy way of encoding, where all this info was contained in just one field, we needed to provide a workaround in the `update()` function of the appropriate class in the conversion software. Only when the data fed to the object came from that legacy field, it should be parsed in more detail to be distributed among the subfields. At the start of software development it was thought that all data going into one MARC21 subfield was going to be formatted reasonably similarly such that only one `update()` function was necessary. Hence, we actually ended up with a problem on the level of the software engineering the initial design of the software wasn't foreseen to facilitate this scenario.

As we needed a solution fast, for these cases a quick workaround was coded instead of e.g. an additional layer of abstraction. We ended up defining a *mock subfield "L"* to which this legacy field was mapped. In the code, this passes the parsing to an `updateLegacy()` function that handles these older types of records properly. Note that from a software engineering perspective, cleaner, more generic and more scalable solutions than this workaround can certainly be devised. This will however need to wait until a next refactoring iteration.

The legacy way of encoding had been often used, even after encoding in separate fields became an option. In total, at least 6113 records contained all the publisher's information as one string in one field. This field was ISBD-encoded in the format *place:publisher,year(printerplace:printername,printeryear)*, with multiple items separated by semicolons. Unfortunately, some catalographers would interpret this by, in the case of multiple publishers, writing *place;place:publisher;publisher*, while others would write *place:publisher;place:publisher*. Hunderds of records used either system, but most of them the former. Hence, all records using the latter system were detected and manually fixed by re-encoding the data it contained in the separate fields instead.

In the end, the following order was defined to parse this data:

- Everything up to the first colon is a place name

- Everything else up to the first comma is the publisher's name
- Everything else up to the first open bracket is a year
- Everything else up to the next colon is the printer's place, etc.

This also allows for publisher's names to contain colons (which often occurs). As an additional sanity check for potential errors, a warning was generated whenever an extracted year of publication would contain more than just numbers, the letter *'x'*, or phrases like *'s.a.', 'cop.', 'dep.'*.

One of the main things still lacking is a system to deal with translations of placenames. Often, old or Latin versions of placenames are translated by the catalographer (e.g. *"Anvers [=Antwerpen]"*). Currently the entire string, including possible translations appended to it, is copied into the corresponding MARC21 field. As a result, this string as a whole is regarded as a placename and indexed as such in the new database as well. Other library software than Koha, our current migration target, may have ways to resolve this. For the time being, dealing with translations (not only in this field but also in titles, keywords, etc) is probably the single most important issue left hanging in the air in this entire conversion operation.

F. Series information

It should be one of the more straightforward fields to process, yet it wasn't all rosy. Convention was that, in one field, a series title and the volume number would be separated by semicolon. The most occurring errors were in interpunction, like using colons instead of semicolons. Close second were errors in spelling the series name properly (result of the lack of validation), leading to one series appearing as several groups of distinct series with only 1 letter difference in the name. Whenever the series title had colons or semicolons in them, the issue was further complicated as this clashed with ISBD interpunction.

As a general rule, everything after the last occurring semicolon was regarded as the volume and number information, to be encoded in MARC21 field `490$v`. Everything before was regarded as the series statement (`490$a`). In general, this lead to satisfactory results, though ideally an additional verification of correctness would be welcome. To avoid too much manual work, we may look into trying to semi-automatically link the titles to an external list that is under authority control, as with authors.

G. Comments and notes

Most of the fields that contain remarks and comments can be immediately ported to a suitable `5xx` MARC21 field. Some of the fields in the old database that do not have an immediate counterpart in MARC21 are assigned to comment fields of which the definition comes reasonably close.

There are a few fields in the old database that contain information for which there is not immediately a MARC21 counterpart with a closely corresponding definition available. Most notably, this was the case for the name of the catalographer, which we store within each record - as far as we could see, MARC21 only stores the name of the institution. For these purposes fields `59x` were used.

One so far unresolved issue is the treatment of lists of movements or chapters. Many records contain such a list in one of the comment fields, presenting a first problem: there was no specific field dedicated for this info. The comment fields also may contain information on the time signature, tempo markings or generic comments. Also, there was never a strict formal way defined to encode lists of movements, therefore every catalographer did it their own way, leading to a plethora of numbering and interpunctions systems being found. Ideally, this information is reformatted and put in MARC field `505`. As with the instrumentation, at the moment we found it easier to copy everything as a general comment in field `500`, and leave extraction and reformatting of this data according to the MARC21 definition for a later stage.

H. Links between records

Since a music library contains many compilation albums containing various works, quite a few records describe monograph parts or book chapters. These are recognizable by the word *"onderdeel"* (i.e. "part" in Dutch) in their callnumber. As the very last step in the entire conversion routine, when all records have been made and the list of callnumbers is complete, these records are processed and have their bibliographic level changed in the MARC21 header accordingly, while also adding MARC fields `773$w` and `774$w` to establish parent-child relationships.

Since Allegro-C did not require a parent record to be present in case a part of a work was being described, we initially found 3207 descriptions of parts, for which a *host item* was not found automatically in the other records. For the largest collections, some parent records were manually created. For anything still left to process, a `773$w` empty parent record is created automatically, to be filled in later.

V. Configuration of the migration target

Koha [3] accepts MARC records for import, but internally uses MySQL as storage system. While the raw MARC data is preserved, most of the data gets copied into MySQL tables. Notably, core bibliographic data, edition data, and holdings data are stored in different tables (*biblio*, *biblioitems* and *items* respectively), hence there is a FRBR-like concept in place in the backend.

We generally assume that every bibliographic record in the old database also equates one item, whether lendable or not. Combined with the assumption that all callnumbers are unique and can be made into a barcode or identifier, this facilitates our conversion task significantly, as we can purely focus on the MARC format and do not need to do any special processing for multiple items of the same record anymore. This did require that before the conversion, the callnumbers were made unique.

Holdings data for use within Koha can be imported by adding a field `952` (equivalent to field `852` of the MARC21 standard) to the MARC record for every item, and populating it with the necessary information. At least the holding library and the item type (codes as defined in the local Koha installation, not the MARC institutional codes or header type codes) need to be provided for the built-in circulation system to function.

VI. Conclusion

With this entire conversion process from CSV files to MARC21, we've been able to make a transition from an old MS-DOS based stand-alone catalog to a modern web-accessible OPAC. In the process, several clean-up efforts have weeded out a large number of errors that existed in the database. In fact, if we retroactively apply the current conversion routine to the database as it existed in September 2014 (with approximately 58000 valid records), we obtain a total of 1734 errors that could not be automatically fixed. Another 1447 items had double callnumbers and 669 part descriptions had no parent record, for a total of 3850 errors.

Throughout the process of manually correcting these, often adjacent records were also updated whenever an error was noticed in them that had slipped through the detection rules, e.g. for spelling or interpunction errors. Between September 2014 and May 2015, a total of 14493 records was updated manually, while 5637 had been added in the same period.

Most interpunction and capitalization errors, as well as often repeated errors against supposedly controlled lists, are remedied by pretty trivial search-and-replace operations in the conversion routine, leaving only the less trivial errors to be corrected manually. Errors in semantics are much harder to find automatically and thus remain mostly undetected for now. Our lowest estimate is that at least 15% of the records in the old database contained such nontrivial errors, of which quite a few will now have been corrected. We can however not guarantee that the records in the new catalog are error-free.

The entire conversion process was implemented in C++ [2], as a command line program that accepts a CSV file as input and generates a MARC21 mnemonic file as output. This makes it suitable to convert any catalog in Excel to MARC21. At this point however, conversion rules are mostly hardwired into the code, and adapting it for use elsewhere would first require some significant refactoring: adding a layer of abstraction will be necessary to facilitate extension. Which seamlessly brings us to:

VII. Future Work

Several major decisions still need to be taken regarding some of the vocabularies that will be used, e.g for tonality, instrumentation, etc. Additionally, the complete configuration and list of fields in the new Integrated Library System remains under review. Since newer ILS software also offers more possibilities, and MARC21 is a vastly more extensive format than the previous customly configured one, there are options to introduce new fields (e.g. to store URLs), or to delete certain fields if they have grown out of use.

Most of the numeric data (opus numbers, dates of birth, dates of composition etc.) have been exported as text strings and imported into the new catalog as they were. This makes that at the moment of writing, search based on numeric ordering is not yet possible. In fact, so many of the numeric fields happened to contain non-numeric characters (question marks, different ways of formatting dates, even plain comments added by the catalographer) that this will still require a major cleanup effort.

The Brussels Conservatories Library also digitizes materials on request. These may in the nearby future be linked to the records by recording an URI in the MARC21 `852$u` field. The adoption of Linked Open Data would be a next step to work on, yet more time to investigate the how is needed. This might need custom development, at least when Koha remains the migration target, for which there are currently no resources available.

A. On Language

The ultimate goal of a library catalog should be that for every query asked, the set of returned results should be the same, regardless of the language. Since a lot of the data is however significantly language-dependent, this can only be accomplished through structured translation efforts. In the particular case of the Brussels Conservatories Library, the database contains roughly as many predominantly French as predominantly Dutch records, while the structured vocabularies in use having been Dutch-based.

Multilingual support is one of the main issues still remaining. Keywords, titles, placenames, ... are sometimes accompanied by a catalographer-added translation, but more often not. MARC21 is rather lacking in support for multilingual records, e.g. the keyword field `650` has no subfield that indicates what language the keyword is in. As a result, when searching for a *string quartet*, one is likely to miss the dozens of records that have *quatuor á cordes* as keyword.

Expanding the use of controlled vocabularies could remedy this to some extent. However, maintaining several sets of vocabularies for each language would be daunting to keep synchronized. Adding the same term several times, once in each language, would also multiply the time needed to create the record. A more streamlined workflow would be to decide on one common language to be used throughout the catalog, and have any necessary translations only done in the user interface when necessary - however, such decision would also need internal (political) support.

B. On the Software

Finally, let us note that the dedicated software that was written for this project [2], definitely has future work cut out for it already. Better generics, facilitating extendability, more documentation etc. are a must before it is ready to be used by other parties. It was written in C++ because of the author's familiarity with that language, but Python is at least as suitable for a text-processing intensive project like this one: a port to another programming language would be worthwhile to consider.

We are happy to report that the conversion software does successfully creates correct MARC21 mnemonic files from the data extracted in CSV-format from the old catalog. These were compiled into MARC21 proper and bulk-imported into a Koha installation, configured for use at the Brussels Conservatories Library. At the moment of writing, the beta-version of the new catalog is available online for testing by the general public [14].

This represents a significant upgrade from the old MS-DOS based catalog, sporting a multilingual interface, full record search with fuzzy matching and highlighting of results, advanced search on item type, editor, ISBN, ... , filtering, etc. Functionality for circulation, adding records, validation and authority control etc will first need to be thoroughly tested before those will be activated on this new public OPAC.

References

[1] B. Eversberg, *Allegro-C 14 Systemhandbuch*. Braunschweig, Germany: Universittsbibliothek der TU Braunschweig, 1995. ISBN 3-927115-25-8.

[2] J. Ganseman, *csv2marc*. Available online: https://github.com/jganseman/csv2marc

[3] Koha Community, *Koha Library Software*. Available online: http://koha-community.org/

[4] Infor Library Systems, *V-smart*. Available online: http://go.infor.com/libraries/solutions/infor-v-smart/

[5] J. Ganseman, *Azerty driver for Windows 7 NTVDM*. Available online: https://github.com/jganseman/azerty

[6] P. Veenstra, S. van der Berg, T. Frssman and U. Wohlers, *DOSBox*, Available online: http://www.dosbox.com/

[7] UB Braunschweig, *allegro-C: Software fr Bibliotheken*. Available online: http://www.allegro-c.de

[8] T. Reese, *MarcEdit* Available online: http://marcedit.reeset.net

[9] Library of Congress, Network Development and MARC Standards office, *MARC Standards* Available online: http://www.loc.gov/marc/

[10] Jan Dewilde, *Conservatoriumbibliotheken: de Assepoesters van het muziekerfgoedbeleid*, In: Bibliotheek & Archiefgids, vol. 86, nr. 1, pp. 9-14, jan. 2010.

[11] Federatie van Organisaties op het gebied van het Bibliotheek-Informatie- en Dokumentatiewezen (FOBID), *Regels voor catalogusbouw gedrukte muziek*, Den Haag, The Netherlands: Nederlands Bibliotheek en Lektuur Centrum, 1989. ISBN 90-6252-051-0

[12] International Association for Music Libraries, *List of codes for medium of performance*, Available online: http://www.iaml.info/sub-commission-unimarc

[13] C. Mullin, M. Huismann, D. Iseminger, N. Lorimer. D. Paradis, R. Schmidt, H. Vermeij, *Best Practices for Music Cataloging Using RDA and MARC21*, v1.1, RDA Music Implementation Task Force, Music Library Association, feb. 2015. Available online: http://bcc.musiclibraryassoc.org/bcc.html

[14] Available at http://catalog.b-bc.org