Metadata:
Batch editing of MARC records
(work area D)

Mark Philips
Serhiy Polyakov

May 2, 2007
Draft 1
# Table of Contents

1. Introduction .................................................................................................................................................. 1
2. MARC Import/Export ...................................................................................................................................... 1
3. Batch editing of MARC records .................................................................................................................... 1
   3.1.1. Mapping North Richland Hills Public Library's Historical Photographs collection set .............. 1
   3.1.2. Tools and software for batch editing of MARC records................................................................. 3
4. Upgrading metadata records for North Richland Hills Public Libraries ....................................................... 3
   4.1. General information about the MARC record structure ..................................................................... 3
   4.2. Decomposing and upgrading MARC records ..................................................................................... 3
5. References .................................................................................................................................................. 5

Appendix A. PHP script for upgrading records of North Richland Hills Public Library's Historical Photographs collection ................................................................. 5
1. Introduction
This document describes workflows developed for the conversion of two sets of metadata records and serves the following objectives:
- Research different avenues for the batch import and export of MARC records from popular integrated library systems.
- Identify tools/software which aids in batch editing of MARC records.
- Document workflow for batch editing MARC records.

2. MARC Import/Export
Most Integrated Library Systems includes capabilities of importing/exporting MARC records. For example, Innovative Interfaces ILS gives libraries full MARC record support including interactive or batch export and import in MARC format.

For the systems that do not support export/import of MARC records MARC Specialized Tools <http://www.loc.gov/marc/marctools.html > at the Library of Congress website can be used to facilitate these procedures. This page includes any software program that provides enhanced usability to MARC 21 records and systems. Specifically, BIBLIObase, BookWhere, and Validator TM Subjects and Names Authority Database provide export/import features for some systems that do not support these features natively.

3. Batch editing of MARC records
3.1.1. Mapping North Richland Hills Public Library’s Historical Photographs collection set
For the purposes of interoperability, THDI has agreements upon methods for describing archival images, display images, and thumbnails in the object metadata. This should be done differently in every metadata scheme.

According THDI Metadata Practices document, in MARC, all links should be handled in the 856 field that should be included in the main bibliographic record. The first indicator for this field will almost always be 4, for http. A second indicator value of 1 indicates that the record refers to a version, not to the original resource, which is appropriate for digitized resources and thumbnails generally. A second indicator value of 2 should be used when the 856 field links to a related resource, such as cover art or an item record.

Subfield 3 specifies "what portion or aspect of the resource the electronic location and access information applies" to. This is commonly used for online tables of contents, but for THDI it will also be used to indicate an image thumbnail. THDI members should select from the following set of tags for the contents of subfield 3:
- Thumbnail image -- used for low resolution images
- Display image -- used for the primary access copy
- Archival image -- used for the archival or master copy (TIFF), although this does not have to be included and best practices may suggest that it should not be included
- Full text -- used for both PDFs and plain text
- Item URL -- link to the item record on the home or local system
- Item PURL -- permanent link to the item record on the home or local system for systems that have this option enabled
- Cover art -- used for low resolution associated images
One of the deliverables of the Uplift project is to upgrade set of MARC records from North Richland Hills Public Library's Historical Photographs collection to meet THDI requirements. Field 962 has been used in original records to describe links to the thumbnails, archival, and display images. The following example of MARC record illustrates the upgraded record. For the purpose of THDI this description will be handled within field 856.

The following is the description of mapping between the fields 962 and 856 for the set of 863 records from North Richland Hills Public Library's Historical Photographs collection.

Each record contains two fields 962. However, there is one record that contains one field 962 and several records that do not contain field 962.

If field 962 contains subfields $a, $t, $u, $v, the content of subfield $u describes link to the archival (tiff) image. This is also noted in subfield $t with wording like "Download Full Image (TIFF)" or similar.

If field 962 contains subfields $a, $e, $t, $u, $v, the content of subfield $u describes link to the display (jpg) image. This is also noted in subfield $t with wording like "Click to View" or similar. Subfield $e contains link to the thumbnail image.

Each field 962 which contain four subfields $a, $t, $u, $v is mapped into field 856 like:

-856 41$3Archival image$uhttp://image.library.nrhtx.com/images/www/history/volume_0030/069u.tiff

Indicators 4 and 1 are used as recommended by THDI Metadata Practices document. Content of subfield 865$u equals content of 962$u. Subfield $3 uses Archival Image value. Note that content of subfields $a, $t, $v is not used in new 856 field.

Each field 962 which contain five subfields $a, $e, $t, $u, $v is mapped into two fields 856 like:

-856 41$3Archival Image$uhttp://image.library.nrhtx.com/images/www/history/volume_0030/069u.tiff
Content of the first field 865 pertains to content of the original 962$e, which is the link to thumbnail. Value “Thumbnail image” is used in subfield $3 of 856 field.

Content of the second field 865 pertains to content of the original 962$u, which is the link to display image. Value “Display image” is used in subfield $3 of 856 field.

Fields 856 need to be added at the end of each record.

3.1.2. Tools and software for batch editing of MARC records

Several software options for performing batch editing of MARC records have been investigated.

First one is free MarcEdit software that includes MarcEdit Script Maker tool. This tool is able to add fields to the set of MARC records based on multiple conditions. However, this tool is unable to use content of existing fields for newly added fields what is needed for performing upgrade procedure. MarcEdit was not used in batch editing because of this deficiency.

Second option was to use custom script that would read marc records from the text file, modify them, and write into output file. Such script was developed using PHP scripting language. The source of the script is provided in Appendix A.

4. Upgrading metadata records for North Richland Hills Public Libraries

4.1. General information about the MARC record structure

Two standards, ANSI/NISO Z39.2 and ISO 2709, define the structure of the MARC record. Various publications from the Library of Congress, Network Development and MARC Standards Office, including MARC 21 Specifications for Record Structure, Character Sets, and Exchange Media provide official descriptions of the structure and semantics of the content designation used in MARC record.

The MARC record is in the form of an ASCII text record with a very specific structure. The primary component parts of the MARC record structure are:

- Leader
- Directory
- Control Fields
- Data Fields

In MARC 21, the control and data fields are indicated by a three digit numeric Field Tag. Fields may have one or more subfields, indicated by a subfield code delimiter and a subfield code. Indicators are additional structures at the field level that precede subfield data.

4.2. Decomposing and upgrading MARC records

Several structural elements of the MARC have to be modified as a result of addition of 856 at the end of the record.

List of structural elements that need to be modified:

Leader
• 00-04 - Record length: The computer-generated, five-character numeric string that specifies the length of the entire record. The number is right justified, and each unused position contains a zero.

• 12-16 - Base address of data: The computer-generated, five-character numeric string that indicates the first character position of the first variable control field in a record. The number is right justified, and each unused position contains a zero.

Directory
Entries for each new 856 field should be added according to the following structure. Note that addition of these entries affects base address of data and record length.

• 00-02 - Tag: Three numeric characters that identify an associated field.

• 03-06 - Field length: Four numeric characters that indicate the length of the field, including indicators, subfield codes, data, and the field terminator. The number is right justified, and each unused position contains a zero.

• 07-11 - Starting character position: Five numeric characters that indicate the starting character position of the field relative to the Base address of data (Leader/12-16) of the record. The number is right justified, and each unused position contains a zero.

Data fields
A set of new 856 fields should be added at the end of the record.

The following is a sample MARC record. In fact, MARC is a continuous ASCII text string without line breaks. However, for presenting this sample here, line breaks have been added at position 90. Also, nonprinting characters have been substituted as following:

\ for ASCII: 29(Dec)/1D(Hex) MARC 21: Record Terminator
^ for ASCII: 30(Dec)/1E(Hex) MARC 21: Field Terminator
$ for ASCII: 31(Dec)/1F(Hex) MARC 21: Delimiter
# for ASCII: 32(Dec)/20(Hex) MARC 21: Blank
• for ASCII: 32(Dec)/20(Hex) where it does not have a special meaning

New or altered parts of the record are presented in bold.

PHP script (see Appendix A) developed for the records upgrade performs the following:

• reads records from the file one by one
• performs decomposition of a record at the required level of granularity
• assembles the new 856 fields based on the mapping algorithm described above
• assembles new directory entries for new 865 fields
• calculates new base address of data
• calculates starting character positions and length of new 865 fields
• calculates new record length
• assembles a record with updated and new structures
• writes a record into output file

5. References

Innovative Interfaces: Millennium: Cataloging (Innovative Interfaces)  
<http://www.iii.com/mill/catalog.shtml>

MARC 21 Specifications for Record Structure, Character Sets, and Exchange Media. (Library of Congress)  
<http://www.loc.gov/marc/specifications/specrecstruc.html>

<http://www.loc.gov/marc/marctools.html>

MarcEdit Homepage: Your Complete Free MARC Software (Terry Reese)  


Appendix A. PHP script for upgrading records of North Richland Hills Public Library's Historical Photographs collection

<?php
//Open files
$FName="NRH_Historic.out";
$fp=fopen($FName, "rb") or die ("Could not open the file");

$FileOut="NRH_Historic_upgraded.out";
$fh=fopen($FileOut, "w");

//Max number of records to read from file
$MaxRecords=1000;

//Main "for"
for ($ng=0; $ng<=$MaxRecords; $ng=$ng+1)
{
  //reads record in to $RecArr
  unset($RecArr);
  $RecLen=fopen($fp, 5);
  //JUST READ FIRST 5 CHARACTERS
  if (feof($fp))
  {
    break;
  }
  $RestLen=$RecLen - 5;
  $RecRest=fopen($fp, $RestLen);
  //JUST READ ALL RECORD AFTER FIRST 5 CHARACTERS
  if (feof($fp))
  {
    break;
  }
}
$RecArr=$RecLen.$RecRest;

//JUST READ a RECORD INTO $RecArr

//START: Work with a record

//(1) For Leader:
unset($LeaderArr);

$LeaderArr[0]=substr($RecArr,0,5); //WILL CHANGE - record length
$LeaderArr[1]=substr($RecArr,5,7);
$LeaderArr[2]=substr($RecArr,12,5); //WILL CHANGE - base address of data
$LeaderArr[3]=substr($RecArr,17,7);

//(2) For Directory:
unset($BaseAddrArr);
unset($DirLenArr);
unset($DirNumbFieldsArr);
unset($DirectArr);

$BaseAddrArr=substr($RecArr,12,5); //WILL CHANGE
$DirLen=$BaseAddrArr-24;
$DirNumbFieldsArr=($DirLenArr=$BaseAddrArr-24)/12; //1 - to exclude dir (-field) terminator - chr(30);
for ($k=0; $k<=$($DirNumbFieldsArr-1); $k+=$k+1) //fields within a MARC record
{
$DirectArr[$k][0]=$k+1; //FieldCounter
$DirectArr[$k][1]=substr($RecArr,24+($k*12),3); //00_02_Tag
$DirectArr[$k][2]=substr($RecArr,27+($k*12),4); //03_06_FieldLength
$LastFieldLength = $DirectArr[$k][2];
$LastFieldAddr = $DirectArr[$k][3];
}

//(3) For Control_001_008 new:
unset($Control_001_008Arr);

$k=0;
$ControlLength=0;
while ($DirectArr[$k][1] < 10)
{

$ControlLength=$ControlLength + $DirectArr[$k][2];
$k+=$k+1;
}

$Control_001_008Arr=substr($RecArr,$BaseAddrArr,$ControlLength-1); //without field terminator - chr(30)

//(7) For Data_010_999:
unset($Data_010_999ArrRaw);
unset($Data_010_999TagArr);
unset($Data_010_999Ind1Arr);
unset($Data_010_999Ind2Arr);
unset($Data_010_999SubCodeArr);

$i=0;
for ($k=0; $k<=$($DirNumbFieldsArr-1); $k+=$k+1)
{
if ($DirectArr[$k][1]>-10)
{
$t=$DirectArr[$k][1];
//without field terminator = chr(30)
$Data_010_999ArrRaw[$i]=$t.substr($RecArr,$BaseAddrArr+$DirectArr[$k][3],$DirectArr[$k][2]-1);

$i=$i+1;

for($i=0; $i<=(count($Data_010_999ArrRaw)-1); $i=$i+1)
{
    $Data_010_999TagArr[$i]=substr($Data_010_999ArrRaw[$i],0,3);
    $Data_010_999Ind1Arr[$i]=substr($Data_010_999ArrRaw[$i],3,1);
    $Data_010_999Ind2Arr[$i]=substr($Data_010_999ArrRaw[$i],4,1);
    $Data_010_999SubCodeArr[$i]=explode(chr(31), substr($Data_010_999ArrRaw[$i],6));
}

//END: Work with a record

//UPGRADE:

$Archival = '';
$Thumb = '';
$Display = '';

foreach ($Data_010_999TagArr as $FieldKey => $FieldTagValue) {
    $ftype = 'tif'; //
    if ($FieldTagValue === '962') {
        foreach ($Data_010_999SubCodeArr[$FieldKey] as $SubKey => $SubValue) {
            if (substr($SubValue, 0, 1) === 'e') {
                $ftype = 'thumb_display';
            }
        }
        if ($ftype === 'tif') {
            foreach ($Data_010_999SubCodeArr[$FieldKey] as $SubKey => $SubValue) {
                if (substr($SubValue, 0, 1) === 'u') {
                    $Archival = $SubValue;
                }
            }
        }
        elseif ($ftype === 'thumb_display') {
            foreach ($Data_010_999SubCodeArr[$FieldKey] as $SubKey => $SubValue) {
                if (substr($SubValue, 0, 1) === 'e') {
                    $Thumb = 'u'.substr($SubValue, 1);
                }
                if (substr($SubValue, 0, 1) === 'u') {
                    $Display = $SubValue;
                }
            }
        }
    }
}

elseif ($ftype === 'thumb_display') {
    foreach ($Data_010_999SubCodeArr[$FieldKey] as $SubKey => $SubValue) {
        if (substr($SubValue, 0, 1) === 'e') {
            $Thumb = 'u'.substr($SubValue, 1);
        }
        if (substr($SubValue, 0, 1) === 'u') {
            $Display = $SubValue;
        }
    }
}

AddField_1 = ''; AddField_2 = ''; AddField_3 = '';

if ($Archival != '')
{
    AddField_1 = '41'.chr(31).'3Archival image'.chr(31).$Archival.chr(30);
}

if ($Thumb != '')
{
    AddField_2 = '41'.chr(31).'3Thumbnail image'.chr(31).$Thumb.chr(30);
}

if ($Display != '')
{
    AddField_3 = '41'.chr(31).'3Display image'.chr(31).$Display.chr(30);
}

// Calculate increase in record length:
$fieldLengthAdd = strlen(AddField_1.$AddField_2.$AddField_3);

$l1 = 0;
$l2 = 0;
$l3 = 0;

if ($AddField_1 != '')
{$l1 = 1;}
if ($AddField_2 != '')
{$l2 = 1;}
if ($AddField_3 != '')
{$l3 = 1;}

$numOfFieldsAdd = $l1 + $l2 + $l3;
$dirLengthAdd = $numOfFieldsAdd * 12;
$recordLengthAdd = $dirLengthAdd + $fieldLengthAdd;
$newRecLength = $LeaderArr[0] + $recordLengthAdd; // new
$newRecLength = sprintf("%05s", $newRecLength);

// Increase in base address of data
$baseAddressOfDataAdd = $dirLengthAdd;
$newBaseAddressOfData = $LeaderArr[2] + $baseAddressOfDataAdd;
$newBaseAddressOfData = sprintf("%05s", $newBaseAddressOfData);

// Length of each dir entry for new fields:
$dirLength_1 = strlen($AddField_1);
$dirLength_1 = sprintf("%04s", $dirLength_1);
$dirLength_2 = strlen($AddField_2);
$dirLength_2 = sprintf("%04s", $dirLength_2);
$dirLength_3 = strlen($AddField_3);
$dirLength_3 = sprintf("%04s", $dirLength_3);

// Starting char positions for new fields:
$newField1_addr = $LastFieldAddr + $LastFieldLength;
$newField1_addr = sprintf("%05s", $newField1_addr);

$newField2_addr = $newField1_addr + $dirLength_1;
$newField2_addr = sprintf("%05s", $newField2_addr);

$newField3_addr = $newField2_addr + $dirLength_2;
$newField3_addr = sprintf("%05s", $newField3_addr);
//START ASSEMBLING:

$NewLeader = $newRecLength.$LeaderArr[1].$newBaseAddressOfData.$LeaderArr[3];

$numbFields=$DirNumbFieldsArr;  //Will change
$newDir = '';
for($k=0; $k<$numbFields; $k=$k+1)
{
    $newDir = $newDir.$DirectArr[$k][1].$DirectArr[$k][2].$DirectArr[$k][3];
}
//$newDir=$newDir.chr(30);

if ($dirLength_1 > 0)
{
    $newDir=$newDir.'856'.$dirLength_1.$newField1_addr;
}
if ($dirLength_2 > 0)
{
    $newDir=$newDir.'856'.$dirLength_2.$newField2_addr;
}
if ($dirLength_3 > 0)
{
    $newDir=$newDir.'856'.$dirLength_3.$newField3_addr;
}
$newDir=$newDir.chr(30);

$newControl = $Control_001_008Arr.chr(30);

$newField='';
for($i=0; $i<=(count($Data_010_999ArrRaw)-1); $i=$i+1)
{
    $newField = $newField.$Data_010_999Ind1Arr[$i].$Data_010_999Ind2Arr[$i];
    for($s=0; $s<=(count($Data_010_999SubCodeArr[$i])-1); $s=$s+1)
    {
        $newField = $newField.chr(31).$Data_010_999SubCodeArr[$i][$s];
    }
    $newField = $newField.chr(30);
}
if ($dirLength_1 > 0)
{
    $newField = $newField.$AddField_1;
}
if ($dirLength_2 > 0)
{
    $newField = $newField.$AddField_2;
}
if ($dirLength_3 > 0)
{
    $newField = $newField.$AddField_3;
}

//RECOSTRUCT
$newRecord = $NewLeader.$newDir.$newControl.$newField.chr(29);
fwrite($fh, $newRecord);
}
fclose($fp);
fclose($fh);
?>