Common Terminology Mapping Experiment II with 400,000 MARCXML records of University of Illinois at Urbana-Champaign (UIUC) Library

(Boaz) Sunyoung Jin
October 15, 2014

To examine performance of the developed Common Terminology (CT) in achieving and improving metadata interoperability, empirical evaluations are planned and progressing with Harvard (MARC), MIT (QDC), and UIUC (MARCXML) records through cooperation of three universities. As a prototype and a case study, a Common Terminology of MARC, MODS, DC and QDC is developed to achieve and improve metadata interoperability among very different degree of specificity and generality standards. As a bridge terminology, CT aims to embrace many standards that many communities are using according to their needs, but to provide uniformity to search.

To convert (Q)DC of MIT records to CT, first, a conversion with Python language is designed and developed. It measures transfer and non-transfer rates, and lexical and semantic match rates. As a result of the conversion of mapping experiments, total transfer rate from (Q)DC of MIT to CT is 99.9%. Lexical and semantic match rates are 98.7% and 100%. Loss of information rate is extremely lower as 0.00463%. CT maximizes lexical and semantic interoperability reducing significantly the gaps of different degrees of generality or specificity. CT minimizes considerably loss of information at multiple levels.

Along with the successful result, another mapping experiment is done with 400,000 MARCXML records of UIUC library. The conversion for MARCXML to CT mapping is developed in Python language during September 2014. For the conversion, MARC to CT crosswalk in csv form is developed based on MARC to CT 1-1 crosswalk. A special data structure for MARC tag, indexes, and subfield is designed by encoding and decoding methods. Several methods are suggested to measure transfer, non-transfer, degree of tag match rates, and semantic match rate. Remarkably, in the result, CT shows very high performance in the MARCXML to CT mapping experiment, although CT has only 12 common terms (less than Dublin Core) and 58 qualifiers (many fewer than MARC tags). Through the result of the experiment, we conclude that CT shows higher performance in achieving and improving metadata interoperability, minimizing loss of information and preserving the specificity and precision of the source metadata records.

1. MARC to CT crosswalk

First of all, the development of MARC to CT crosswalk is fundamental for the mapping experiment of MARC to CT. The development bases on MARC to CT 1.1 crosswalk. The MARC to CT 1.1 crosswalk does not include detail mappings with indexes and subcodes. However, for the actual mappings with the conversion program, the detail mappings with indexes and subcodes should be specified clearly. MARC to CT crosswalk in csv form clarifies the mapping with both MARC tag and indexes and subcodes. Since MARC has many tags with subcodes, the mapping of MARC to CT that has 12 common terms with 58 qualifiers should be generalized to preserve information as much as possible. Thus, the generalized mapping with

only tag number regardless specific indexes and subcodes is also included in the MARCtoCTcrosswalk. For example, 046 tag, special coded dates, has several mappings into CT with different types (subproperty) depending on subcodes, to specify what kinds of date is used. 046 with subcode ‘c, e, k, or l’ is mapped into ct:date type=”issued.” 046 with subcode ‘j’ is mapped into ct:date type=”modified.” 046 with subcode ‘m or n’ is mapped into ct:date type=”available.” Also, for the generalized mapping regardless indexes and subcodes, 046 into ct:date mapping is included in the crosswalk. This is a part of MARC to CT crosswalk. The full crosswalk is found in MARCtoCTcrosswalk.

MARC tag	Index 1	Index 2	subcode	Common Terminology 1.1
leader				description type="recordinfo"
001				identifier type="controlNumber" source="UIUC"
003				identifier type="controlNumber" source="UIUC"
005	.....			description type="recordinfo"
020				identifier type="isbn"
022				identifier type="issn"
022			a	identifier type="issn"
024	2		a	identifier type="ismn"
024				identifier type="identifierOther"
028	.....			identifier type="issueNumber"
046				date
046			c	date type="issued"
046			e	date type="issued"
046			k	date type="issued"
046			l	date type="issued"
046			j	date type="modified"
046			m	date type="available"
046	….		n	date type="available"

2. The Designed Conversion Program in Python Programming Language

2.1. Defining MARC tag Class

With the developed MARCtoCTcrosswalk, settling a data structure to describe all MARC tags including various indexes and subcodes was not simple. Thanks to advice of Professor Dubin, I decided to encode MARC tags adding 'I' for expressing index 1, 'J' for index 2, and 'S' for subcodes. For example, 264I1J0Sc means MARC tag number is 264; index 1 is 1; index 2 is 0; and subcode is c. To decode it, Queue data structure is used. The class for defining MARC tag with indexes and subcode is the below with Python language.

```
from collections import deque
class marcTag:
    def __init__(self, tag, value):
```

```
self.tag=tag
self.value=value
if 'leader' in tag:
   self.tagnum='leader'
   self.ind1="
   self.ind2="
   self.subcode="
else:
   qtag=deque(tag)
   self.tagnum="
   self.ind1="
   self.ind2="
   self.subcode="
   for i in range(0,3):
      self.tagnum+=str(qtag.popleft())
   #print self.tagnum
   for i in range(0, len(qtag),2):
      x=qtag.popleft()
      if x=='I':
         self.ind1=qtag.popleft()
      elif x=='J':
         self.ind2=qtag.popleft()
      elif x=='S':
         self.subcode=qtag.popleft()
```

2.2. Leader and Control Field

The conversion program has mainly three parts to convert MARCXML to CT: leader; control field (e.g., 001, 005, 008); and data field parts for other tags, because UIUC records consist of these three parts. Leader and control field parts are converted into CT and their some information is also interpreted into several CT to preserve much information. For example, the content of leader is '01026cam a2200301Ka 4500' in the below UIUC record. The 'leader' information is transferred into 'description type="recordinfo"' of CT. And its content according to 6th and 7th column is interpreted with typeGenre with authority, Library of Congress MARC type, and with description type="issuance."

```
<description type="recordinfo">01026cam a2200301Ka 4500</description>
<typeGenre authority="LCMARCtype">text</typeGenre>
<typeGenre authority="LCMARCtype">Monograph/Item</typeGenre>
<description type="issuance">multipart monograph</description>
```

Whenever a statement is transferred into CT, it is counted as transferred. But, if there is no mapping in the developed MARCtoCTcrosswalk, it is counted as non-transferred. The interpreted information into several CTs such as leader, 007 or 008 tags isn't counted as transferred. In above example, only the first statement converted from leader is counted as transferred.

2.3. Data Fields

According to the MARCtoCTcrosswalk, the datafields parts in UIUC records are transferred into CT. The program, first, retrieves information of tag number, index 1, index 2, subfield code, and

value from two different types of datafields. The information of two types is retrieved by two kinds of regular expressions:

p3=re.compile(r'<marc:datafield tag="(.*)" ind1="(.*)" ind2="(.*)"><marc:subfield code="(.*)">(.*)<') #defining a regular expression to find tag, ind1, ind2, and subcode pattern
p4=re.compile(r'<marc:subfield code="(.*)">(.*)<') #defining subfield pattern.

The p3 pattern searches and matches the first and second lines of the below UIUC records. The p4 pattern searches and matches the third line with the same tag and indexes information of p3 pattern.

```
<marc:datafield tag="041" ind1="0" ind2=" ">
        <marc:subfield code="a">rus</marc:subfield>
        <marc:subfield code="b">eng</marc:subfield>
</marc:datafield>
```

These information is transferred into CT like the below. CT shows the generalization of language code, 041 tag. That is, they are mapped into ct:language with authority iso 639-2 or iso 639-3 nevertheless subcode $a indicates a language code of text/sound track or separate title, and subcode $b indicates a language code of summary or abstract. CT does not indicate detail information whether the language code is for title or summary.

```
<language authority ="iso639-2">rus</language>
<language authority ="iso639-3">eng</language>
```

But, if the information according to subfields is critical to preserve information, we can specify the detail information in the transferred content. Actually, the conversion program translates tag 852 information like the below, because UIUC records use the tag often and I feel we need to specify the detail for content values.

```
def translate852(mfield):
   if mfield.subcode=='a':
      content='Location-'+str(mfield.value)
   elif mfield.subcode=='b':
      content='Sublocation-'+str(mfield.value)
   elif mfield.subcode=='h':
      content='Coded location-'+str(mfield.value)
   elif mfield.subcode=='i':
      content='Item part-'+str(mfield.value)
   elif mfield.subcode=='p':
      content='Piece designation-'+str(mfield.value)
   elif mfield.subcode=='t':
      content='Copy number-'+str(mfield.value)
   else:
      content=mfield.value
   return content
```

The full version of the conversion program is in the **Designed Conversion Program**.

3. Methodology to Measure transfer, non-transfer, Tag Match, and Semantic Match Rates

The designed conversion program also measures transfer rate, non-transfer rate (loss of information rate), MARC tag match rate, and semantic match rates in the mapping experiment with MARCXML records of UIUC.

3.1. Measuring Transfer and Non-transfer Rates

To measure transfer and non-transfer rates, total transferred tags with different subcodes and indexes are counted. For example, in the below MARCXML record, leader, 001 tag, 005, 008, 035 subcode a, and 035 subcode 9 are checked whether each of them is transferred (counted as transferred) or not (counted as non-transferred) into the developed CT.

```
<leader>00655nam  2200193 i 4500</leader>
<controlfield tag="001">200001</controlfield>
<controlfield tag="005">20020415161359.0</controlfield>
<controlfield tag="008">770519s1965    ilu          00000 eng d</controlfield>
<datafield tag="035" ind1=" " ind2=" ">
   <subfield code="a">(OCoLC)ocm02977467</subfield>
</datafield>
<datafield tag="035" ind1=" " ind2=" ">
   <subfield code="9">AAV-1954</subfield>
</datafield>
```

If a statement of a MARCXML record is not transferred into CT, the non-transfer variable is increased and the tag is added into noMatchtagDic dictionary showing how many times it isn't transferred in UIUC records.

3.2. Measuring Tag Match Rate

Since matching with all information with MARC tag, indexes and subcode is very rarely happen, to measure the tag match rate, three methods are used:

- Perfect match rate matched by all of MARC tag, indexes and subcode
- Subcode match rate matched by tag, and any subcodes or subcode 'a' as a default
- General match rate matched by MARC tag number only

The perfect match means the encoded tag with all information of MARC tag, indexes and subcode such as 086I0Sa is mapped into CT according to MARCtoCTcrosswalk, For example, 043 field, geographic area code, with subcode 'a' (043Sa) is perfectly matched with ct:subject type="spatial" based on MARC to QDC crosswalk. 260Sa, 260Sb, 260Sc, 533Sb, 533Sd, and 533Se encoded tags show perfect match.

If the encoded tag with all information is not matched into CT based on MARCtoCTcrosswalk, only MARC tag and subcode information except indexes is retrieved from the encoded tag. And subcode match is investigated. There are two kinds of subcode match: MARC tag and subcode match except indexes; and MARC tag and subcode 'a' match as a default except indexes. As a first subcode match example, 245 field with subcode 'b,' 245Sb, is matched with ct:title

type=”subtitle.” 245Sb, 245Sc, 650Sz are the first subcode match tags. The second subcode match examples with subcode default ‘a’ are 020, 035, 040, 100, 245, 300, 440, 490, 504, 650, 830, 852, etc.

If the encoded tag is not perfect matched nor subcode matched, the general match is considered. The general match means that only MARC tag number is matched. As an example of general match, 300 field is matched with ct:format type=”extent,” regardless indexes and subcode information. 035, 040, 082, 084, 111,300, 490, 700, 776, 8, 99430, 852, etc. are general matched tags.

In the conversion program, these tag match rates are investigated in the findMatch module like:

```
def findMatch(tag, content,mfield,marcCTDic,fhandle2,transfer, ntransfer, tstatement,pmatch,
smatch,gmatch,exactMatch,broadMatch):
    roleTags=['100','110','111','700','710','711','720']
    if tag in marcCTDic:#perfect match by all tag number, subcode, and indexes
        key=marcCTDic[tag]
        if mfield.tagnum in roleTags and mfield.subcode=='e' :
            key=key+' role="'+str(content)+'" authority="LCMARCrelators"'
        transfer, tstatement=writingconversion(key,content,fhandle2, transfer, tstatement)
        pmatch+=1
        exactMatch,broadMatch=semanticMatch(exactMatch,broadMatch, tag) #find semantic match rates
    elif mfield.tagnum in marcCTDic and mfield.subcode!='':#subcode match by tag number and subcode
        ntag=str(mfield.tagnum)+'S'+str(mfield.subcode)
        if ntag in marcCTDic:
            key=marcCTDic[ntag]
            if mfield.tagnum in roleTags and mfield.subcode=='e' :
                key=key+' role="'+str(content)+'" authority="LCMARCrelators"'
            transfer, tstatement=writingconversion(key,content,fhandle2, transfer, tstatement)
            smatch+=1
            exactMatch,broadMatch=semanticMatch(exactMatch,broadMatch, ntag) #find semantic match rates
        elif mfield.subcode=='a': #subcode match by tag number and subcode ‘a’ as a default
            key=marcCTDic[mfield.tagnum]
            transfer, tstatement=writingconversion(key,content,fhandle2, transfer, tstatement)
            smatch+=1
            exactMatch,broadMatch=semanticMatch(exactMatch,broadMatch, mfield.tagnum) #find semantic match
rates
        else: #general match by only tag number
            for mkey in marcCTDic.keys():
                if mfield.tagnum in mkey or mkey in mfield.tagnum:
                    key=marcCTDic[mkey]
                    if mfield.tagnum in roleTags and mfield.subcode=='e' :
                        key=key+' role="'+str(content)+'" authority="LCMARCrelators"'
                    transfer, tstatement=writingconversion(key,content,fhandle2, transfer, tstatement)
                    exactMatch,broadMatch=semanticMatch(exactMatch,broadMatch, mkey) #find semantic match rates
                    gmatch+=1
                    break
    elif mfield.tagnum in marcCTDic: #general match by only tag number
        for mkey in marcCTDic.keys():
            if mfield.tagnum in mkey or mkey in mfield.tagnum:
                key=marcCTDic[mkey]
                transfer, tstatement=writingconversion(key,content,fhandle2, transfer, tstatement)
                gmatch+=1
```

```
            exactMatch,broadMatch=semanticMatch(exactMatch,broadMatch, mkey) #find semantic match rates
            break
    else:#if there is no-matched by tag number, indexes, and subcode, increase non-transfer variable
        ntransfer,tstatement=notransfer(mfield.tagnum, ntransfer,tstatement)

    return transfer,ntransfer, tstatement, pmatch, smatch,gmatch,exactMatch,broadMatch
```

3.3. Measuring Semantic Match Rates

To measure semantic match rate, I investigate again how fields interpreted by MODS elements are semantically related to CT, although semantic relations were sincerely considered when MARCtoCTcrosswalk was built and when CT was designed and developed. This time, the semantic relation is measured by SKOS mapping concept schemes.

First, semantic relations are investigated for each MARCXML to CT mapping based on SKOS mapping concept schemes (e.g., skos:exactMatch, skos:closeMatch, skos:broadMatch, skos:narrowMatch, and skos:relationMatch). The investigated semantic relation is added in the MARC to CT crosswalk. For example, MARC 041 field, language code, is mapped into ct:language with different authorities according to subcodes (e.g., iso 639-2, rfc1766, rfc3066, rfc 4646, and iso 639-3). It is semantically exactly matched mapping. Thus, the mapping is considered as semantic exactMatch relation. MARC 044 field, country of publishing/producing entity code, is mapped into ct:publisher; 044 with subcode ‘a’, into publisher type=”place”; 044 with subcode ‘c’ into publisher type=”place” authority=”iso3166.” Since the mapping with subcode is much exactly mapped with sub-property type=”place” that specifies the published place, the mapping is considered as exactMatch. On the other hand, 044 without subcode is generally mapped into ct:publisher. Thus, this mapping is considered as broadMatch. The below table is a part of MARC to CT crosswalk with SKOS semantic relations.

tag	Index1	Index2	subcode	Common Terminology (CT)	
041				language authority ="iso639-2"	exactMatch
041			2_rfc1766	language authority ="rfc1766"	exactMatch
041			2_rfc3066	language authority ="rfc3066"	exactMatch
041			2_rfc4646	language authority ="rfc4646"	exactMatch
041			2_iso639-3	language authority ="iso639-3"	exactMatch
042	….			description type="authentication" authority="LCMARCauthentication"	exactMatch
044				Publisher	broadMatch
044			a	publisher type="place"	exactMatch
044	….		c	publisher type="place" authority="iso3166"	exactMatch

In the defined semantic relations, I found mainly two matches are used: broadMatch and exactMatch. In the conversion program, thus, these matches are defined as broadMatch and exactMatch dictionaries. Simply, semanticMatch module examines in which dictionary a field mapping is included, and increases exactMatch or broadMatch variable.

```
def semanticMatch(exactMatch,broadMatch, field):
    if field in exactMatchDic:
```

```
        exactMatch+=1
    elif field in broadMatchDic:
        broadMatch+=1
    return exactMatch,broadMatch
```

4. The Results of the Mapping Experiment with UIUC(MARCXML) Metadata Records

The result of the MARCXML to CT conversion shows amazing performance of CT that has only 12 common terms that are less than DC core 15 elements, and 58 qualifiers that are many fewer than MARC tags and subcodes. **Showing very high mapping rate and very low loss of information rate is founded** on that we developed **CT based on MARC tag usage in records and in search engines. And the Common Terminology concept, a set of common terms of commonly used standards as a bridge terminology, is very effective to achieve interoperability** among different standards (even very different degree of specificity and generality such as MARC, DC).

4.1. An example of the converted CT from MARCXML of UIUC records

As we can see from the example, CT description is easily readable and understandable, and very simple.

```
<?xml version="1.0" encoding="UTF-8"?>
<!--filename:UIU5.xml  records:33961  totalUIUrecords:321359-->
<CT xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://courseweb.lis.illinois.edu/~sunjin/CT/1.1/
http://courseweb.lis.illinois.edu/~sunjin/CT/1.1/ct.xsd"
        xmlns="http://courseweb.lis.illinois.edu/~sunjin/CT/1.1/">
        <identifier type="controlNumber" source="UIUC">4863288</identifier>
        <description type="recordinfo">20080912002420.0</description>
        <date type="issued">1975</date>
        <publisher type="place">ja </publisher>
        <typeGenre type="genre" authority="LCMARCgenre">fiction</typeGenre>
        <language authority="iso639-2">eng</language>
        <description type="recordinfo">00653cam a2200205Ia 4500</description>
        <typeGenre authority="LCMARCtype">text</typeGenre>
        <typeGenre authority="LCMARCtype">Monograph/Item</typeGenre>
        <description type="issuance">multipart monograph</description>
        <identifier type="controlNumber">(OCoLC)ocm56626489</identifier>
        <description type="recordinfo">Cataloging Source-UIU</description>
        <language authority ="iso639-2">eng</language>
        <language authority ="iso639-3">jpn</language>
        <contributor name="personal">Bellow, Saul.</contributor>
        <title>Dangling man /</title>
        <rights type="holder">Saul Bellow ; edited with notes by the Taiyosha Editorial Department
(unabridged).</rights>
        <publisher  type="place">[Tokyo?] :</publisher>
        <publisher>Taiyosha Press,</publisher>
        <date type="issued">1975.</date>
        <format type="extent">186 p. ;</format>
        <format type="extent">19 cm.</format>
```

<language>Introduction and notes in Japanese.</language>
<identifier type="identifierOther">X0</identifier>
<identifier type="identifierOther">UIU</identifier>
<identifier type="collection">Location-IU</identifier>
<identifier type="collection">Sublocation-Rare Book & Manuscript Library [non-circulating]</identifier>
<identifier type="collection">Coded location-KRAUS</identifier>
<identifier type="collection">Item part-813 B417d1975a</identifier>
<identifier type="collection">Piece designation-30112062501694</identifier>
<identifier type="collection">Copy number-1</identifier>

4.2. Performance Results

As a result along with the converted CT from MARCXML of UIUC records, performance of CT is as follows:

- Total transfer rate from MARCXML of UIUC to CT is 95.2709%.
- Non-transfer rate, loss of information rate from MARC records to CT is 4.729%. **It is very low rate, considering that CT has only 12 common terms (less than Dublin Core) and 58 qualifiers (many fewer than MARC tags).**
- Semantic match rate by SKOS concept is 100%
  - exactMatch rate: 55.337% and
  - broadMatch: 44.6628%.
- Tag Match Rate
  - The perfect match rate matched by all of MARC tag, indexes and subcode is 16.1834%;
  - subcode match rate by tag and subcodes is 43.068%; and
  - general match rate for MARC tag number only is 40.748%.

4.3. Non-transferred MARC tags

The dictionary for non-transferred tags shows which tags have no mapping in the MARCtoCTcrosswalk. The most often used but not transferred tag is 049 and it is for local holdings for OCLC. 092 tag is for local Dewey call number of OCLC. But, these non-transferred tags do not critically affect loss of information rate. Total loss of information rate, non-transfer rate, is only 4.729% for 400.000 MARCXML records of UIUC. The noMatchtag dictionary shows the sorted non-transferred tags with frequency of use.

Sorted **noMatchtagDic**: OrderedDict([('696', 1), ('315', 1), ('793', 1), ('526', 1), ('361', 1), ('512', 1), ('514', 1), ('565', 1), ('956', 1), ('946', 1), ('098', 2), ('695', 2), ('939', 2), ('935', 2), ('996', 2), ('790', 2), ('917', 2), ('556', 2), ('952', 2), ('697', 3), ('544', 3), ('592', 3), ('263', 3), ('791', 4), ('302', 4), ('656', 4), ('866', 4), ('380', 5), ('261', 5), ('913', 7), ('960', 7), ('598', 8), ('096', 9), ('524', 9), ('346', 9), ('945', 10), ('599', 11), ('853', 13), ('547', 14), ('902', 16), ('257', 17), ('581', 18), ('069', 20), ('585', 21), ('596', 22), ('863', 27), ('545', 29), ('017', 31), ('400', 32), ('027', 33), ('765', 36), ('886', 41), ('351', 48), ('692', 49), ('055', 51), ('767', 52), ('344', 54), ('753', 60), ('270', 71), ('563', 72), ('911', 85), ('256', 89), ('777', 116), ('212', 132), ('350', 132), ('030', 140), ('586', 150), ('799', 206), ('070', 214), ('410', 260), ('570', 265), ('088', 354), ('525', 370), ('006', 392), ('301', 404), ('693', 523), ('074', 538), ('963', 552), ('025', 583), ('699', 683), ('012', 850), ('254', 853), ('949', 886), ('503', 963), ('691', 1346), ('938', 1574), ('840', 1832), ('099', 2162), ('305', 2754), ('539', 2795), ('936', 2824), ('247', 3041), ('011', 3098), ('508', 3192), ('588', 3265), ('891', 3552), ('262', 4678), ('590', 4895), ('850', 5692), ('910', 7005), ('306', 8154), ('066', 8421), **('037', 10974), ('265', 11254), ('048', 16695), ('987', 23711), ('090', 39528), ('690', 85477), ('092', 128942), ('880', 142277), ('049', 204944)])**

4.4. The Result of the Conversion Program

>>>
filename: UIU1.xml 135915 totalUIUrecords: 135915
totalUIUrecords: 135915 transfer: 4566186 ntransfer: 52187 tstatement: 4618373 pmatch: 644183 smatch: 2125140 gmatch: 1796863
exactMatch: 2592894 broadMatch: 1971693

filename: UIU2.xml 37437 totalUIUrecords: 173352
totalUIUrecords: 173352 transfer: 6157477 ntransfer: 106886 tstatement: 6264363 pmatch: 840925 smatch: 2804648 gmatch: 2511904
exactMatch: 3427368 broadMatch: 2727148

filename: UIU3.xml 56647 totalUIUrecords: 229999
totalUIUrecords: 229999 transfer: 8509113 ntransfer: 182095 tstatement: 8691208 pmatch: 1289332 smatch: 3680828 gmatch: 3538953
exactMatch: 4656453 broadMatch: 3848390

filename: UIU4.xml 57399 totalUIUrecords: 287398
47829 th record has an error with empty content of 008 content:
totalUIUrecords: 287398 transfer: 10756274 ntransfer: 394669 tstatement: 11150943 pmatch: 1721344 smatch: 4632441 gmatch: 4402489
exactMatch: 5935324 broadMatch: 4815423

filename: UIU5.xml 33961 totalUIUrecords: 321359
totalUIUrecords: 321359 transfer: 12231743 ntransfer: 542800 tstatement: 12774543 pmatch: 1964110 smatch: 5273914 gmatch: 4993719
exactMatch: 6764377 broadMatch: 5461435

filename: UIU6.xml 55907 totalUIUrecords: 377266
totalUIUrecords: 377266 transfer: 14163307 ntransfer: 678292 tstatement: 14841599 pmatch: 2299075 smatch: 6108226 gmatch: 5756006
exactMatch: 7811853 broadMatch: 6345313

filename: UIU7.xml 17650 totalUIUrecords: 394916
totalUIUrecords: 394916 transfer: 15004437 ntransfer: 744785 tstatement: 15749222 pmatch: 2428238 smatch: 6462177 gmatch: 6114022
exactMatch: 8299340 broadMatch: 6698430

total records of UIU: 394916
total transferred and non transferred 15749222
transfer: 15004437 , **transfer rate: 95.2709727503**
notransfer: 744785 , **notransfer rate: 4.72902724973**

perpect match matched by all tag, indexes, and subcode: 2428238 , **perpect match rate: 16.183466264**
subcode match matched by tag and subcode: 6462177 , **subcode match rate: 43.0684403553**
tag match matched by only tag: 6114022 , **tag match rate: 40.7480933806**

total semantic matched:exactMatch or broadMatch 14997770
semantic exactMatch: 8299340 , **semantic exactMatch rate: 55.3371601245**
semantic broadMatch: 6698430 , **semantic broadMatch rate: 44.6628398755**
>>>

5. Conclusion

To examine the developed Common Terminology, the mapping experiments have been done with MIT(QDC) records and UIUC(MARCXML) records. The paper reports 400,000 refined MARCXML records of UIUC to CT mapping experiment. Considering complex MARC records' information with indexes and subcodes, encoding and decoding methods is used. Several mapping methods are used. Surprisingly, the result of the MARCXML to CT conversion proves amazing performance of CT, although CT has only 12 common terms that are less than 15 core elements of DC, and 58 qualifiers that are many fewer than MARC tags and subcodes. Total transfer rate is 95.27%; **Non-transfer rate, loss of information rate is only 4.729%**. Semantic match rate by SKOS concept is 100%.

In light of these results and the result of MIT(QDC) to CT mapping experiment (transfer 99.9%, lexical 98.7%, semantic match rate 100%, and **loss of information rate 0.00463%**), **we conclude that CT shows higher performance in achieving and improving metadata interoperability, minimizing loss of information and preserving the specificity and precision of the source metadata records**.

Showing very high performance and very low loss of information rate is founded on which we developed **CT based on MARC tag usage in Harvard, UIUC, and WorldCat records and in search interfaces.** Over 50% used tags are considered as common terms or qualifiers. Also, **the Common Terminology concept, a set of common terms of commonly used standards as a bridge terminology, is very effective to achieve interoperability** among different standards (even very different degree of specificity and generality such as MARC, MODS and DC & QDC). Through the experiments, it is proved that finding and reusing commonly used terms among existing standards is a very crucial way to build interoperability, instead of creating new schema.