# Augmenting In-house and Vendor-supplied MARC Records

## Automating Batch-Derive of ETD records by XSLT

Lucas Mak
Metadata & Catalog Librarian
Michigan Sate University Libraries

Library Technology Conference 2010 (March 17-18, 2010, St. Paul, MN

# Thesis & Dissertation cataloging at MSU

- Cataloged theses & dissertations in print and microfiche formats only until 2009
  - Subject analysis (LCSH)
  - Authority control of author names
  - Both single and separate approaches

# ETD cataloging at MSU

- In 2009, started to catalog full-text electronic theses & dissertations available in ProQuest
  - Separate record approach
  - Derive from print records
    - Copy and paste permanent URLs from ProQuest
    - For current cataloging only
  - Not systematic in retrospective

# Recent Task

- To catalog MSU's ETDs (1997 to current) with full-text access in ProQuest
  - Total: ~8300
  - Rationale
    - Access to full-text ETDs originated from CIC institutions (1997 to current)
    - Plan to share ETD MARC records among CIC institutions
  - Deduplication necessary

# Option 1: Manual Derive

- Advantages:
  - Established procedure
  - Same look & feel compared to existing ETD records
    - Consistency in subject & name access
- Disadvantages:
  - Slow & repetitive
    - Copy & paste of permanent URLs one at a time
    - Key in identical & similar information repeatedly
      - Can be mitigated by using constant data
  - Coordination required if more than one staff involved

# Option 2: Batch Loading ProQuest MARC Records

- Advantages:
  - Data elements
    - Permanent URL to individual ETD record page
    - Unique identifiers
      - Useful for deduplication for future incremental loads
    - Basic description: Title, Author main entry, Degree type, Date, Pagination for the main text
    - Abstract by author
  - Time-saving
    - Batch processing, e.g. adding GMD

# Option 2: Batch Loading ProQuest MARC Records

- Disadvantages
  - Inconsistent look & feel and/or access between print & electronic
    - No LCSH, only broad descriptors available, e.g. American studies
    - No authority control of names
    - No Dept./Program name in 502
    - Typos in title
    - Title transcription not following AACR2 rules
      - Auto correction of typo instead of using [sic] or [i.e. …]
      - Inputting special characters, symbols, or superscripts as is
    - Incorrect degree type

# Option 2: Batch Loading ProQuest MARC Records

- Disadvantages
  - Lacking certain data elements
    - 245$c → can be added by manipulating 100
    - 006, 007, GMD → can be added in batch
  - 245$a & 245$b merged into one 245$a
    - Can be separated by conditional processing
  - “Junk fields”, e.g. school code, descriptor code
    - Can be removed in batch

=LDR 02323nam 2200325 4500

=001 AAI9734166

=005 20091116134751.5

=008 091116s1997\\\\||||||||||||||||\||eng\d

=020 \\$a9780591441529

=035 \\$a(UMI)AAI9734166

=040 \\$aUMI$cUMI

=100 1\$aPaino, Troy Dale.

=245 14$aThe end of nostalgia: A cultural history of Indiana high school basketball during the Progressive Era.

=300 \\$a220 p.

=500 \\$aSource: Dissertation Abstracts International, Volume: 58-05, Section: A, page: 1881.

=502 \\$aThesis (Ph.D.)--Michigan State University, 1997.

=520 \\$aThe process of negotiation between producers and consumers created a varied and complex set of cultural meanings in the community support for high school basketball in the three Indiana cities …

=590  \\$aSchool code: 0128.
=650  \4$aAmerican Studies.
=650  \4$aHistory, United States.
=650  \4$aHistory, Modern.
=690  \\$a0323
=690  \\$a0337
=690  \\$a0582
=710  2\$aMichigan State University.
=773  0\$tDissertation Abstracts International$g58-05A.
=790  \\$a0128
=791  \\$aPh.D.
=792  \\$a1997
=856  \\$uhttp://gateway.proquest.com/openurl?url_ver=Z39.88-2004&rft_val_fmt=info:ofi/fmt:kev:mtx:dissertation&res_dat=xri:pqdiss&rft_dat=xri:pqdiss:9734166

# Option 3: Matching & Batch-derive

- Advantages
  - Consistency in look and feel, and access
    - Deriving from print records
  - Efficient
    - Deriving in batch
    - Automatic carry over corresponding permanent URL
- Challenges
  - To identify suitable technology
  - To find match point(s)

# Technology

- XSLT (Extensible Stylesheet Language Transformation)
  - Within the family of XML
    - Case sensitive
    - Current version: 2.0
  - “Transformation” means:
    - Manipulation of XML documents by creating a new document based on the original document
  - Common usages in library context
    - Web display
      - e.g. converting EAD into HTML for display
    - Metadata crosswalking
      - Data selection and manipulation

# Technology

- XSLT
  - Multiple inputs and outputs
  - Comparing data from multiple inputs
    - document ( )
    - key ( )

# Match Strategy

- Characteristics of thesis & dissertation records
  - Identical titles uncommon
  - Special characters (e.g. scientific notations) not uncommon in title info
  - Full name in statement of responsibility
  - Multiple contributions by one author
  - Affecting uniqueness and reliability of match points

# Match Strategy

- MSU record
  - 001 (OCLC & Skyriver no.)
  - 100 (Author name)
  - 245$a (Title proper)
  - 245$b (Other title info)
  - 245$c (statement of responsibility)
  - 260$c (Date on t.p.)
  - 502 (Degree type, Date of acceptance)
- ProQuest record
  - 001 (UMI no.)
  - 020 (ISBN)
  - 100 (Author name)
  - 245$a (Title proper & other title info)
  - 502 (Degree type, Date)
  - 791 (Degree type)
  - 792 (Date)

# Match Strategy

- No common unique identifier
- “Common” data
  - Author name
    - No Authority control
  - Title
    - Merged 245$a & 245$b
    - Typos and different transcription rule
  - Date
    - Nature of date uncertain
  - Degree type

# Match Strategy

- String matching
  - Matching with single criterion
    - Useful for matching unique values
    - Possibility of false hit
  - Matching with multiple criteria
    - Useful for matching values that are not unique by themselves but unique when combined, e.g. author name + date
    - Possibility of false drop caused by error in data

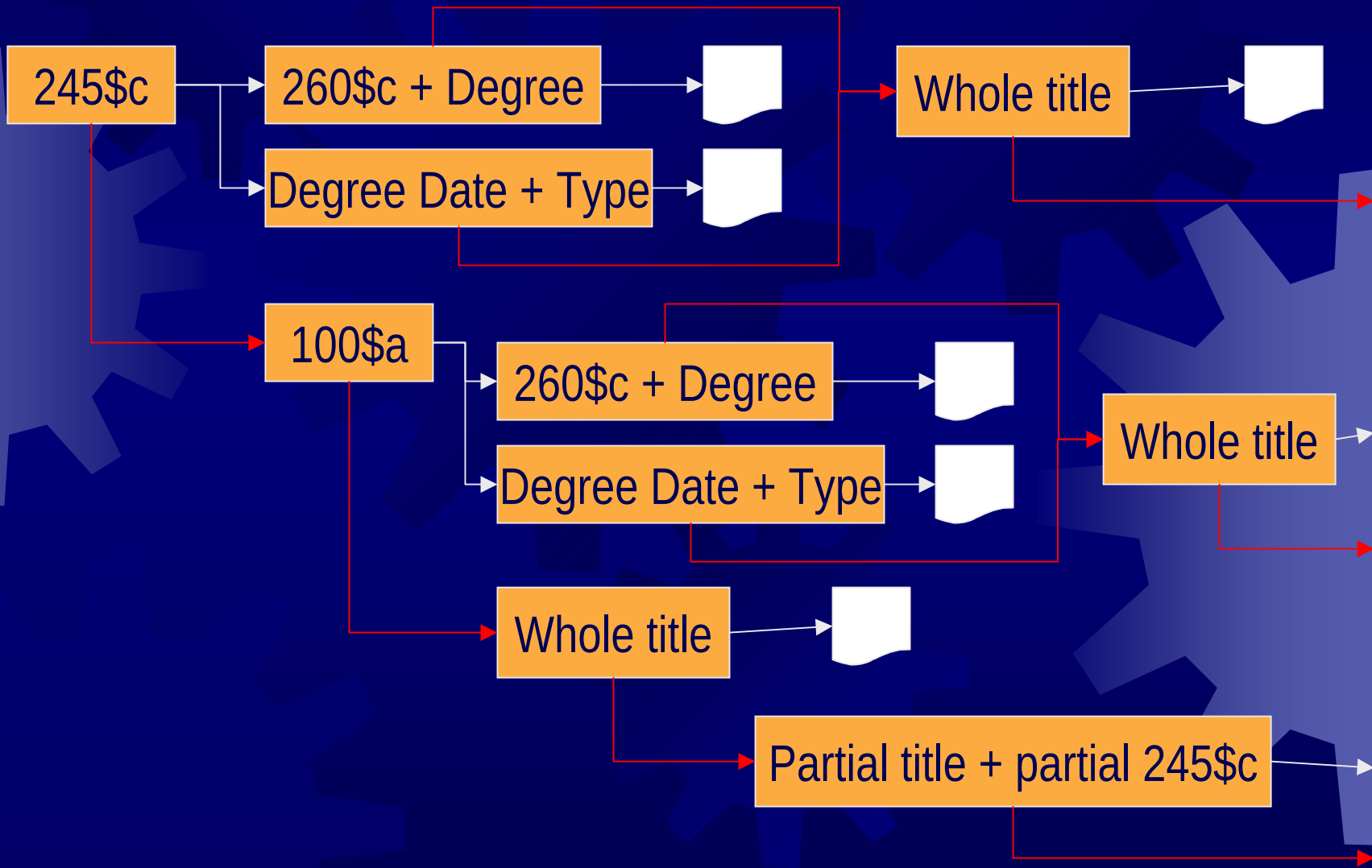
# Match Strategy

- String matching
  - Exact match
    - Pattern specified equals to the string
    - Will fail when different in capitalization, punctuation, spacing, etc.
      - Requires pre-match string normalization
  - Fuzzy match
    - Existence of certain pattern in any part of a string
    - Prone to false hit
      - Requires additional matching
  - Multi-step matching
    - Serialization of conditional processing, i.e. series of True/False tests and corresponding action(s)

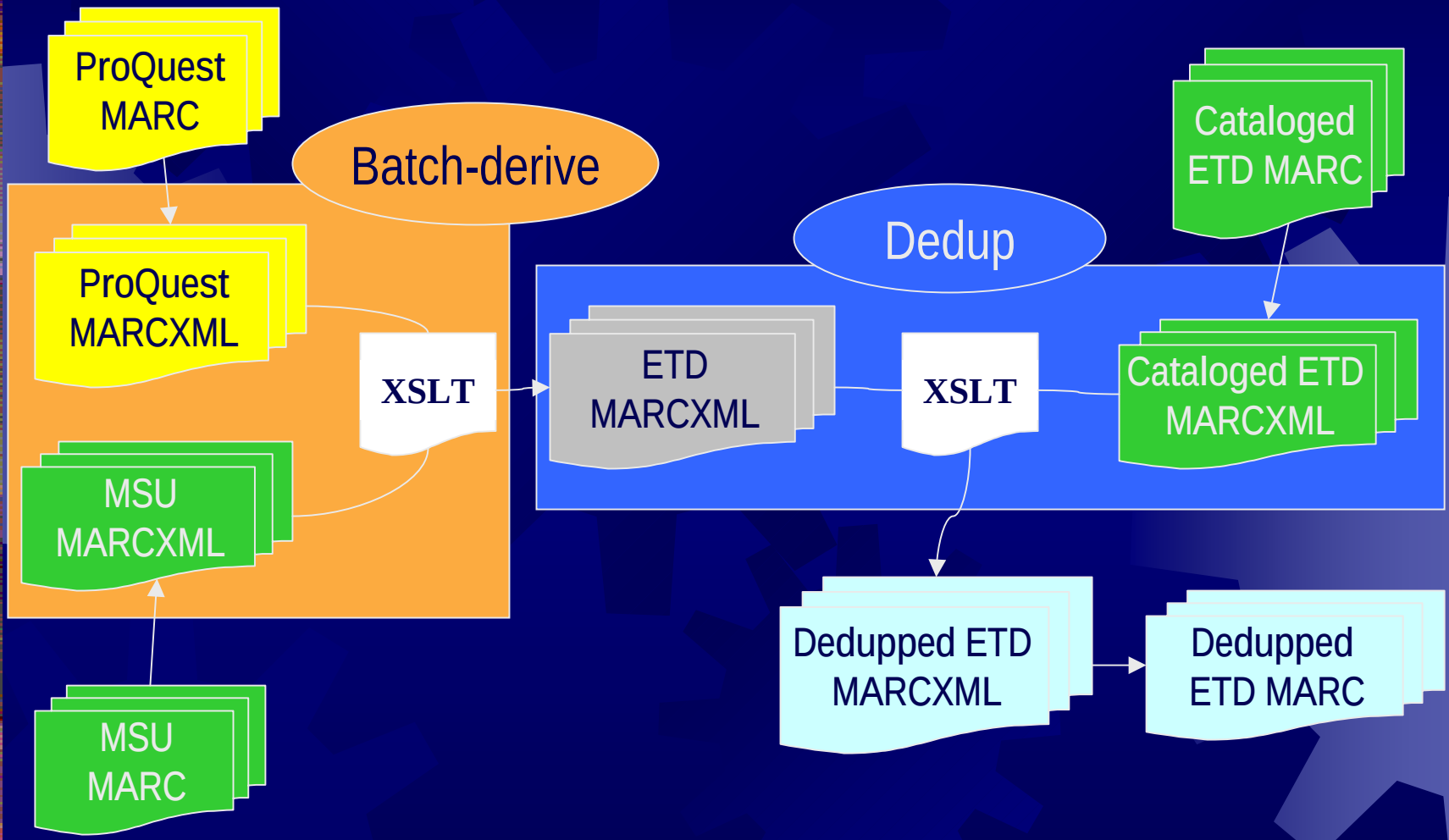
# Match Strategy

- Matching criteria implemented
  - Different combinations of Title (complete/partial string), Name (complete/partial string), Date (complete string), Degree type (partial string)
  - Create new matching point from existing data
    - Flip MARC 100 in ProQuest records into direct order → used to compare with 245$c from MSU print records

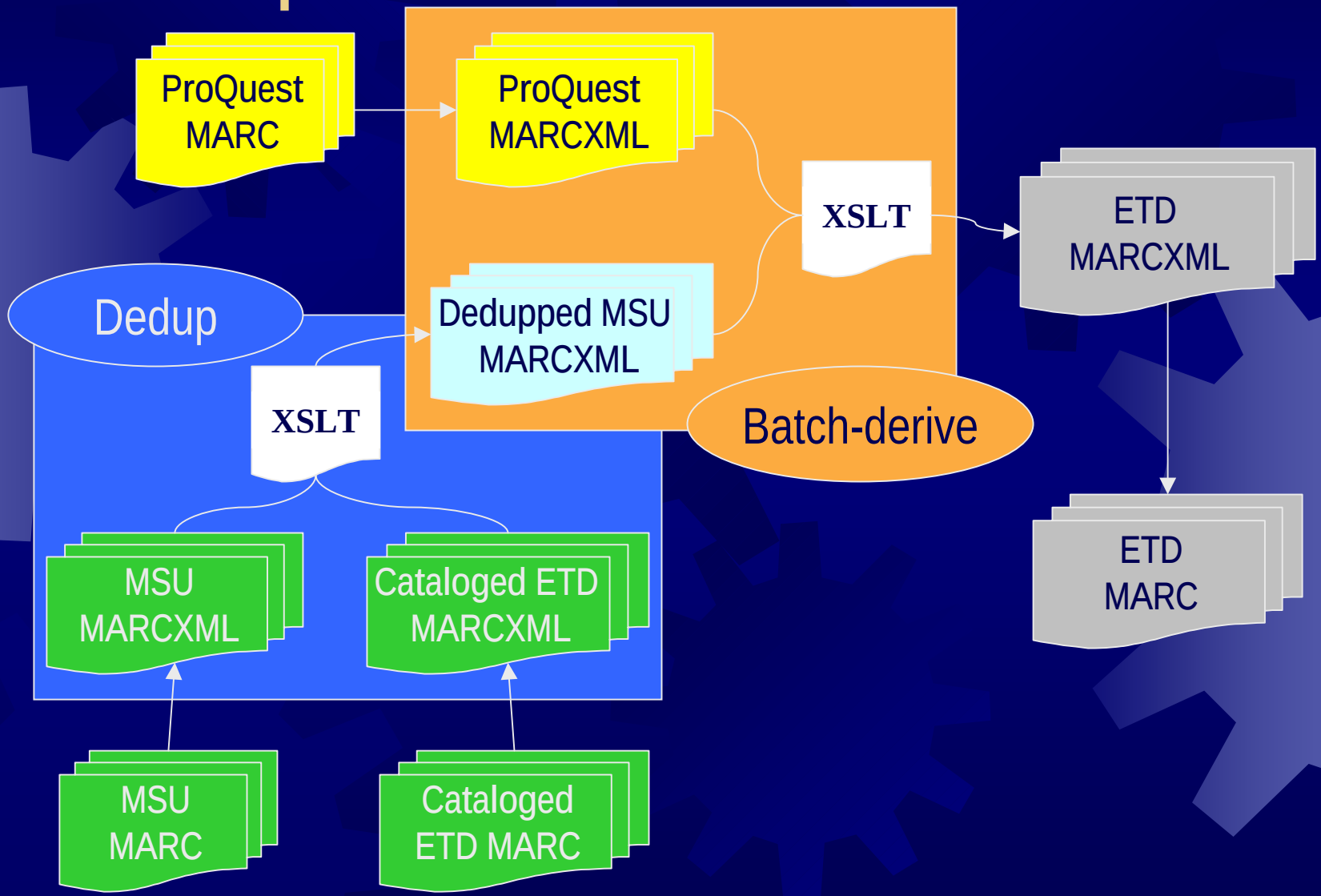
# Match Strategy



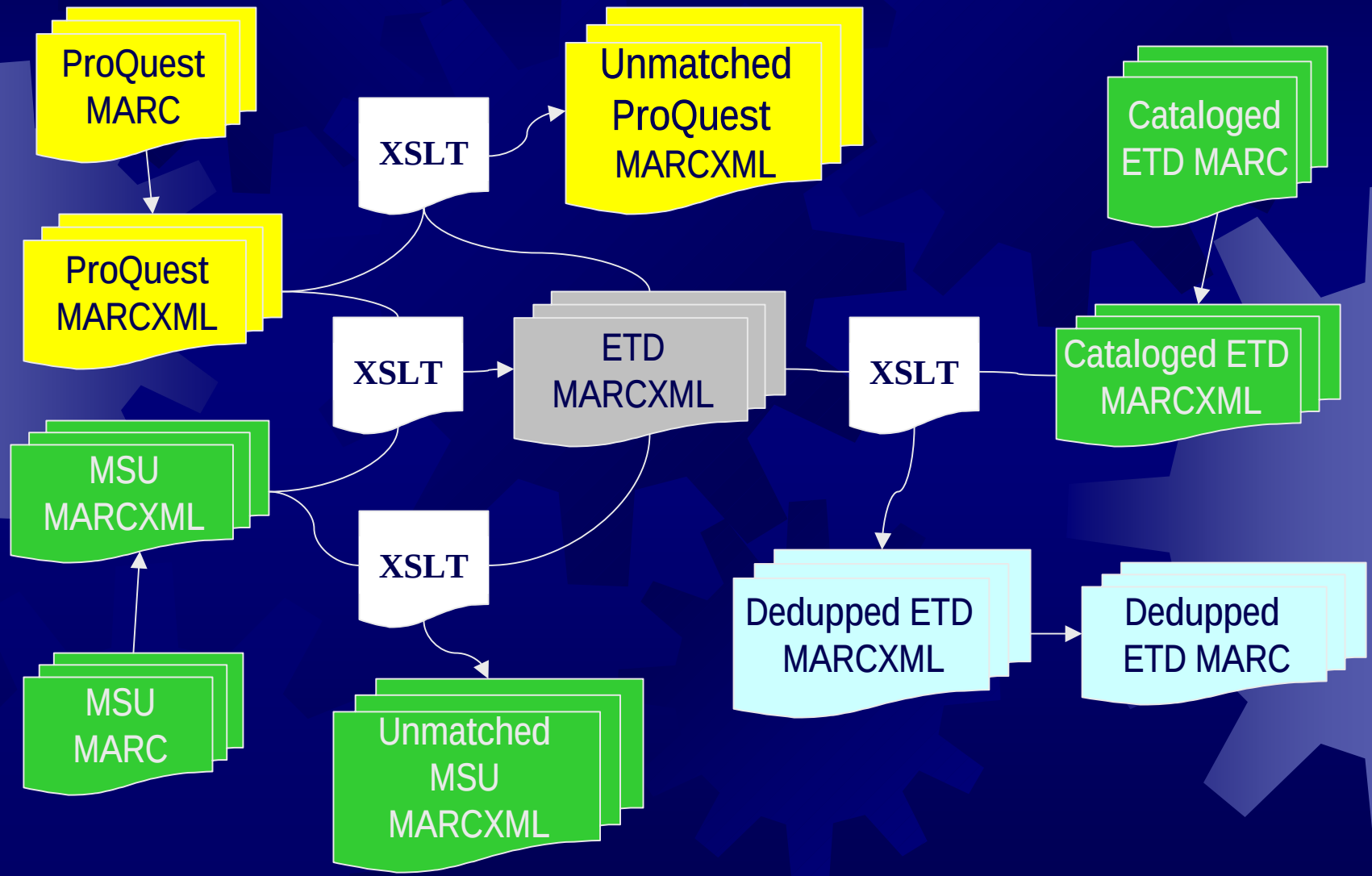
# Batch-derive & Dedup Process



# Alternative Batch-derive & Dedup Process



# Batch-derive & Dedup Process



# XSLT for Batch-derive Action

- Three templates
  - Matching template
    - To compare data elements between MSU and ProQuest MARCXMLs
    - Once matched:
      1. Derive from MSU print MARCXML
      2. Copy data from ProQuest MARCXML
  - Derive template
    - To derive MARC fields and data from print counterpart (MSU print thesis MARCXML)
    - To insert new data into output
    - To try correcting data
  - Copy template
    - To copy data from ProQuest records
      - ISBN, UMI no., Permanent URL

# XSLT for Dedup Action

- Dedup Template
  - To compare MARC field 245 between the output “ETD MARCXML” from batch-derive action and “Cataloged ETD MARCXML”
  - Output unique titles only

# Matching Template

- Conditional processing
  - Specify matching criteria and corresponding action(s)
  - XSLT elements: <xsl:choose> in combination with <xsl:when> & <xsl:otherwise>
  - XSLT functions:
    - document( )
      - Introduce an external XML document into the current context document
      - Need to keep track which the current context document is
    - contains( ), matches( ), starts-with( ), ends-with( )
      - Pre-match string manipulation
    - boolean( )
      - Existence of a node (i.e. MARC field or subfield)

# Matching Template

```
<xsl:when test="document('MSUprintThesis.xml')/marc:record[normalize-
    space(substring-after(translate(translate(lower-case(marc:datafield[@tag=245]
    /marc:subfield[@code='c']),$symbols,$spaces),$apos,' '),'by '))=$doc1flipped100]">
  <xsl:for-each select="…">
    <xsl:choose>
      <xsl:when test="substring(replace(translate(translate(replace(marc:datafield[@tag=
      260]/marc:subfield[@code='c'],'c',''),$symbols,$spaces),'[]',' '),' ',''),1,4)=$doc1Year">
        <xsl:choose>
          <xsl:when test="lower-case(substring(substring-after(marc:datafield[@tag=502]
            /marc:subfield[@code='a'],'('),1,1))=$doc1Degree">
            <marc:record>
              <xsl:call-template name="print2Electronic"/>
                <xsl:for-each select="$ProQuestPosition">
                  <xsl:call-template name="ProQuestElements"/>
                </xsl:for-each>
            </marc:record>
            …
      </xsl:when>
      <xsl:otherwise>
      <xsl:choose>
        <xsl:when test="$doc2normalized502Date=$doc1Year">
```

Key in MSU XML

Select MSU XML when matched

Insert <marc:record> and trigger derive & copy templates

Select original ProQuest XML

Execute tests under <xsl:otherwise> when <xsl:when> fails

# Derive Template

- Direct copy data from print records
  - MARC leader, 041, 043, 100, 260, 546
  - <xsl:copy-of>
  - <xsl:copy-of select="*[not(self::marc:controlfield[001 <= @tag and @tag <= 009])][not(self::marc:datafield[010 <= @tag and @tag <= 040])][not(self::marc:datafield[049 <= @tag and @tag <= 099])][not(self::marc:datafield[@tag=245])][not(self::marc:datafield[@tag=300])][not(self::marc:datafield[@tag=500])][not(self::marc:datafield[@tag=502])][not(self::marc:datafield[@tag=504])][not(self::marc:datafield[@tag=530])][not(self::marc:datafield[@tag=533])][not(self::marc:datafield[@tag=590])][not(self::marc:datafield[@tag=856])][not(self::marc:datafield[@tag=866])][not(self::marc:datafield[@tag=867])][not(self::marc:datafield[900 <= @tag and @tag <= 999])]"/>

# Derive Template

- Hard-code (i.e. writing) text into output document
  - =003  SKY
  - =006  m\\\\\\\d\\\\\\\
  - =007  cr\bn\\a
  - =040  \\$aEEM$cEEM
  - =049  \\$aQEMO$aEEMT
  - =099  \\$aMSU ONLINE THESIS
  - =538  \\$aMode of access: World Wide Web.
  - =500  \\$aDescription based on print version record.
  - =655  \0$aElectronic dissertations.

# Derive Template

- Example:

```
<marc:datafield tag="049" ind1=" " ind2=" ">
    <marc:subfield code="a">QEMO</marc:subfield>
    <marc:subfield code="a">EEMT</marc:subfield>
</marc:datafield>
```

# Derive Template

- Merge new text into existing string
  - <xsl:value-of>
    - Copying data from a node
  - <xsl:text>
    - Hard-coding text into output
  - Substring( ), substring-before( ), substring-after( )
    - Selecting part of a string
  - Example
    - 008
      - Insert “s” in byte 23 (Form), “b” in byte 24 (Cont), “m” in byte 25
      - =008  010918s2000\\\\xx\a\\\\sbm\\\000\0\eng\d

# Derive Template

```
<marc:controlfield tag="008">
    <xsl:value-of
select="substring(marc:controlfield[@tag=008],1,23)"/>
    <xsl:text>sbm</xsl:text>
    <xsl:value-of
select="substring(marc:controlfield[@tag=008],27)"/>
</marc:controlfield>
```

# Derive Template

- Replace existing data with new data
  - replace( ), translate( )
    - Useful for data correction
  - Example:
    - Replace “PH.D.” with “Ph. D.” in 502

```
<xsl:for-each
select="marc:datafield[@tag=502]/marc:subfield[@code='a']">
    <marc:datafield tag="502" ind1=" " ind2=" ">
        <marc:subfield code="a">
            <xsl:value-of select="replace(.,'PH.D.','Ph. D.')"/>
        </marc:subfield>
    </marc:datafield>
</xsl:for-each>
```

# Derive Template

- Conditional processing
  - Action(s) based on fulfillment of certain condition(s)
  - XSLT elements: <xsl:if>, <xsl:choose> in combination with <xsl:when> & <xsl:otherwise>
  - XSLT functions: contains( ), matches( ), starts-with( ), ends-with( ), boolean( )
- Example 1
  - Inserting 245$h [electronic resource]
    - Mix of single record and separate record approaches
    - Existence of $b → punctuation after $h

# Derive Template

```
<xsl:choose>
  <xsl:when test="marc:subfield[@code='h']">
        …
  </xsl:when>
    <xsl:otherwise>
      <marc:subfield code='a'><xsl:value-of select="normalize-
      space(substring(normalize-space(marc:subfield[@code='a']),1,string-
      length(marc:subfield[@code='a'])-1))"/></marc:subfield>
        <xsl:choose>
          <xsl:when test="marc:subfield[@code='b']">
            <marc:subfield code="h">[electronic resource] :</marc:subfield>
          </xsl:when>
          <xsl:otherwise>
            <marc:subfield code="h">[electronic resource] /</marc:subfield>
          </xsl:otherwise>
        </xsl:choose>
        <xsl:copy-of select="marc:subfield[@code!='a']"/>
    </xsl:otherwise>
</xsl:choose>
```

Existence of $h

Existence of $b

# Derive Template

- Example 2
  - Insert correct OCLC no. into 776$w

```
<marc:subfield code='w'>
   <xsl:text>(OCoLC)</xsl:text>
   <xsl:choose>
     <xsl:when test="count(preceding-sibling::marc:controlfield
    [@tag=001])&gt;=2">
       <xsl:for-each select="preceding-sibling::marc:controlfield
    [@tag=001][contains(lower-case
    (self::marc:controlfield[@tag=001]),'paper')]">
            <xsl:value-of select="normalize-space(substring-before(.,'('))"/>
       </xsl:for-each>
     </xsl:when>
     <xsl:otherwise>
       <xsl:value-of select="preceding-sibling::marc:controlfield
    [@tag=001]"/>
     </xsl:otherwise>
   </xsl:choose>
</marc:subfield>
```

Existence of Multiple 001

Select 001 tagged "paper"

# Copy Template

- Moving data into a different MARC field
  - To move UMI no. from 001 to 028

Create a new field

```
<marc:datafield tag="028" ind1="5" ind2="0">
   <marc:subfield code="a">
      <xsl:value-of select="substring-after
(marc:controlfield[@tag=001],'AAI')"/>
   </marc:subfield>
   <marc:subfield code="b">UMI</marc:subfield>
</marc:datafield>
```

Copy data from 001

# Copy Template

- Create EZProxy link
  - <xsl:value-of>, <xsl:text>

```
<marc:datafield tag="856" ind1="4" ind2="0">
  <marc:subfield code="u">
    <xsl:text>http://ezproxy.msu.edu:2047/login?url=</xsl:text>
    <xsl:value-of
    select="marc:datafield[@tag=856]/marc:subfield[@code='u']"/>
  </marc:subfield>
  <marc:subfield code="z">Connect to online resource - MSU
    authorized users</marc:subfield>
</marc:datafield>
```

Insert EZProzy prefix

Copy link from ProQuest record

# Dedup Template

- Compare MARC 245 between Output from batch-derive action and “Cataloged ETD MARCXML”
- Output unique titles only

# Dedup Template

```
<xsl:template match="marc:record">
    <xsl:variable name="doc1Position" select="self::marc:record"/>
    <xsl:variable name="catalogedETD245"
    select="marc:datafield[@tag=245]/marc:subfield[@code='a']"/>
    <xsl:choose>
        <xsl:when
    test="document('CatalogedETD.xml')//marc:record[marc:datafiel
    d[@tag=245]/marc:subfield[@code='a']=$catalogedETD245]"/>
        <xsl:otherwise>
            <xsl:copy-of select="$doc1Position"/>
        </xsl:otherwise>
    </xsl:choose>
</xsl:template>
```

Set current context as a variable

No action when matched

Copy when not matched

# Implementation Issues

- Inconsistency in cataloging practice
  - Single vs Separate record approach
    - Need to build in extra steps to account for exceptions
  - Transcription approach in ProQuest records
    1. Auto correction of typo in title
    2. Special characters
    3. Random translation of terms, e.g. U.S. → United States
    - Hard to predict and account for in XSLT
    - Need to rely on other matching criteria, e.g. Name + Date

# Implementation Issues

- Mistakes in ProQuest data
  1. Typos in title
  2. Incorrect date info
  3. Incorrect degree type, e.g. M.U.P. vs M.U.R.P.
     - Matching by 1st character → false hit possible
  4. Wrong spacing/ word division
     - Normalize string to take out white space

# Implementation Issues

- Mistakes in MSU data
  - Missing subfield code, e.g. 245$c
  - Typos in names
    - e.g. Michael Jay Renner → Michael Jan Renner
- Mismatch in scope of data
  - ProQuest
    - Published in Dissertation & Thesis Abstract & Index between 1997 & 2008
    - Full-text only
  - MSU
    - 260$c or Degree date (502)
    - Not all print theses between 1997 & 2008 are received and/or cataloged
  - No corresponding targets to match against

# Limitations of XSLT

- True/False matching
  - Not approximate string matching
  - Zero tolerance to differences between targets
    - No way to set a threshold of how close two matched
- Case-sensitive
  - Normalization needed
    - Implication on processing time

# Limitations of XSLT

- Computer processing power
  - > 60 min.
    - Intel Core 2 Duo 2.00GHz, 4 GB RAM, oXygen
    - Large XML file sizes
      - 42 MB in XML, but 24 MB in .mrc
      - Open 2 40 MB files and write a new 40 MB file
    - Pre-test string manipulation & multi-step string matching
    - Should use stand-alone command line XSLT processor?

# Possible adaptations

- Original ETD cataloging workflow
  1. Batch process ProQuest records
  2. Enhanced and corrected by catalogers
  3. Batch derive records for print and microfiche

# Possible adaptations

- Batch-derive from print to electronic
  - Hathi Trust records
    - Text-delimited file with OCLC no. & “volume identifier” pair
    1. Batch search records of print version by OCLC no. in ILS or Connexion
    2. Batch derive from print records
    3. OCLC no. as match point → plug-in the “volume identifier” in new record
    4. Create URL by appending “volume identifier” to handle prefix (http://hdl.handle.net/2027/)

# Reflections

- Unique identifiers vital for matching
  - Copied ISBN and UMI no. from ProQuest records
- Effective matching requires familiarity of source data → normal pattern & exceptions
- Efficiency of perfecting XSLT to cover all exceptions??

# Questions?

Lucas Mak
makw@mail.lib.msu.edu