marcxml2mods

Release 0.2.5

November 16, 2016

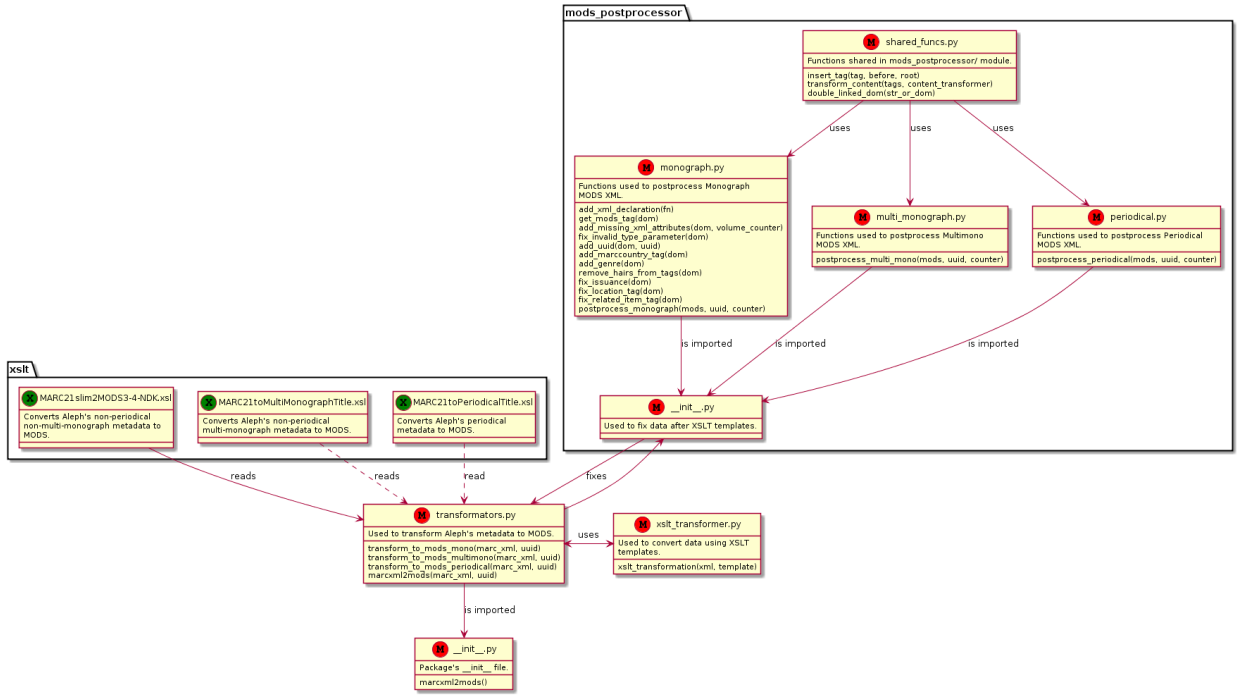
Contents

The package is used for conversion of bibliographic data from MARC XML or OAI to MODS format.

CHAPTER 1

Package structure

Relations between files in this package are captured at following image:



1.1 API

`/api/marcxml2mods`:

1.1.1 transformators submodule

This module is used to transform Marc XML from Aleph to MODS. MODS is defined by XSLT templates.

`marcxml2mods.transformators.`**`_absolute_template_path`**(*fn*)

Return absolute path for filename from local `xslt/` directory.

Parameters **`fn`** (*`str`*) – Filename. `MARC21slim2MODS3-4-NDK.xsl` for example.

Returns Absolute path to *fn* in `xslt` dicretory..

Return type str

marcxml2mods.transformators.**_apply_postprocessing**(*marc_xml*, *xml*, *func*, *uuid*, *url*)
Apply *func* to all <mods:mods> tags from *xml*. Insert UUID.

Parameters

- **marc_xml** (*str*) – Original Aleph record.
- **xml** (*str*) – XML which will be postprocessed.
- **func** (*fn*) – Function, which will be used for postprocessing.
- **uuid** (*str*) – UUID, which will be inserted to *xml*.
- **url** (*str*) – URL of the publication (public or not).

Returns List of string with postprocessed XML.

Return type list

marcxml2mods.transformators.**transform_to_mods_mono**(*marc_xml*, *uuid*, *url*)
Convert *marc_xml* to MODS data format.

Parameters

- **marc_xml** (*str*) – Filename or XML string. Don't use \n in case of filename.
- **uuid** (*str*) – UUID string giving the package ID.
- **url** (*str*) – URL of the publication (public or not).

Returns Collection of transformed xml strings.

Return type list

marcxml2mods.transformators.**transform_to_mods_multimono**(*marc_xml*, *uuid*, *url*)
Convert *marc_xml* to multimonograph MODS data format.

Parameters

- **marc_xml** (*str*) – Filename or XML string. Don't use \n in case of filename.
- **uuid** (*str*) – UUID string giving the package ID.
- **url** (*str*) – URL of the publication (public or not).

Returns Collection of transformed xml strings.

Return type list

marcxml2mods.transformators.**transform_to_mods_periodical**(*marc_xml*, *uuid*, *url*)
Convert *marc_xml* to periodical MODS data format.

Parameters

- **marc_xml** (*str*) – Filename or XML string. Don't use \n in case of filename.
- **uuid** (*str*) – UUID string giving the package ID.
- **url** (*str*) – URL of the publication (public or not).

Returns Collection of transformed xml strings.

Return type list

marcxml2mods.transformators.**type_decisioner**(*marc_xml*, *mono_callback*, *multi-mono_callback*, *periodical_callback*)
Detect type of the *marc_xml*. Call proper callback.

Parameters

- **marc_xml** (*str*) – Filename or XML string. Don't use \n in case of filename.
- **mono_callback** (*fn reference*) – Callback in case of monographic publications.
- **multimono_callback** (*fn reference*) – Callback used in case of multi-monographic publications.
- **periodical_callback** (*fn reference*) – Callback used in case of periodical publications.

Returns Content returned by the callback.

Return type obj

Raises ValueError – In case that type couldn't be detected.

marcxml2mods.transformators.**marcxml2mods**(*marc_xml*, *uuid*, *url*)

Convert *marc_xml* to MODS. Decide type of the record and what template to use (monograph, multi-monograph, periodical).

Parameters

- **marc_xml** (*str*) – Filename or XML string. Don't use \n in case of filename.
- **uuid** (*str*) – UUID string giving the package ID.
- **url** (*str*) – URL of the publication (public or not).

Returns Collection of transformed xml strings.

Return type list

1.1.2 xslt_transformer submodule

This module is used to transform XML using XSLT templates.

API

marcxml2mods.xslt_transformer.**_oai_to_xml**(*marc_oai*)

Convert OAI to MARC XML.

Parameters **marc_oai** (*str*) – String with either OAI or MARC XML.

Returns String with MARC XML.

Return type str

marcxml2mods.xslt_transformer.**_add_namespace**(*marc_xml*)

Add proper XML namespace to the *marc_xml* record.

Parameters **marc_xml** (*str*) – String representation of the XML record.

Returns XML with namespace.

Return type str

marcxml2mods.xslt_transformer.**_read_content_or_path**(*content_or_path*)

If *content_or_path* contains \n, return it. Else assume, that it is path and read file at that path.

Parameters **content_or_path** (*str*) – Content or path to the file.

Returns Content.

Return type str

Raises IOError – whhen the file is not found.

marcxml2mods.xslt_transformer.**_read_marcxml**(*xml*)
Read MARC XML or OAI file, convert, add namespace and return XML in required format with all necessities.

Parameters **xml** (*str*) – Filename or XML string. Don't use \n in case of filename.

Returns Required XML parsed with lxml.etree.

Return type obj

marcxml2mods.xslt_transformer.**_read_template**(*template*)
Read XSLT template.

Parameters **template** (*str*) – Filename or XML string. Don't use \n in case of filename.

Returns Required XML parsed with lxml.etree.

Return type obj

marcxml2mods.xslt_transformer.**xslt_transformation**(*xml*, *template*)
Transform *xml* using XSLT *template*.

Parameters

- **xml** (*str*) – Filename or XML string. Don't use \n in case of filename.
- **template** (*str*) – Filename or XML string. Don't use \n in case of filename.

Returns Transformed *xml* as string.

Return type str

/api/mods_postprocessor/mods_postprocessor:

1.1.3 monograph postprocessor

Module which defines postprocessing for monograph publications.

marcxml2mods.mods_postprocessor.monograph.**add_xml_declaration**(*fn*)
Decorator to add header with XML version declaration to output from FN.

marcxml2mods.mods_postprocessor.monograph.**get_mods_tag**(*dom*)
Find and return HTMLElement with <mods:mods> tag from the *dom*.

marcxml2mods.mods_postprocessor.monograph.**add_missing_xml_attributes**(*dom*, *volume_counter=0*)
Add *xmlns* and *ID* attributes to <mods:mods> tag.

Parameters

- **dom** (*HTMLElement*) – DOM containing whole document.
- **volume_counter** (*int, default 0*) – ID of volume.

marcxml2mods.mods_postprocessor.monograph.**fix_invalid_type_parameter**(*dom*)
"Make sure that <mods:placeTerm> has type="code" attribute.

marcxml2mods.mods_postprocessor.monograph.**add_uuid**(*dom*, *uuid*)
Add <mods:identifier> with *uuid*.

marcxml2mods.mods_postprocessor.monograph.**add_marccountry_tag**(*dom*)
Add <mods:placeTerm> tag with proper content.

marcxml2mods.mods_postprocessor.monograph.**add_genre**(*dom*)
Add <mods:genre> with *electronic volume* content into <mods:originInfo.

marcxml2mods.mods_postprocessor.monograph.**remove_hairs_from_tags**(*dom*)
Use remove_hairs() to some of the tags:

- mods:title
- mods:placeTerm

marcxml2mods.mods_postprocessor.monograph.**fix_issuance**(*dom*)
Fix <mods:issuance> for monographic tags from *monographic* to *single_unit*.

marcxml2mods.mods_postprocessor.monograph.**fix_location_tag**(*dom*)
Repair the <mods:location> tag (the XSLT template returns things related to paper books, not electronic documents).

marcxml2mods.mods_postprocessor.monograph.**fix_related_item_tag**(*dom*)
Remove <mods:relatedItem> tag in case that there is only <mods:location> subtag.

marcxml2mods.mods_postprocessor.monograph.**fix_missing_electronic_locator_tag**(*dom*, *url*)
In case that MODS contains no URL and the location is wrong (physical), add url from *url* parameter.

marcxml2mods.mods_postprocessor.monograph.**fix_missing_lang_tags**(*marc_xml*, *dom*)
If the lang tags are missing, add them to the MODS. Lang tags are parsed from *marc_xml*.

marcxml2mods.mods_postprocessor.monograph.**postprocess_monograph**(**args*, ***kwargs*)
Fix bugs in *mods* produced by XSLT template.

Parameters

- **marc_xml** (*str*) – Original Aleph record.
- **mods** (*str*) – XML string generated by XSLT template.
- **uuid** (*str*) – UUID of the package.
- **counter** (*int*) – Number of record, is added to XML headers.
- **url** (*str*) – URL of the publication (public or not).

Returns Updated XML.

Return type str

1.1.4 multi_monograph postprocessor

Module for postprocessing of multi monograph publications.

marcxml2mods.mods_postprocessor.multi_monograph.**postprocess_multi_mono**(**args*, ***kwargs*)
Some basic postprocessing of the multi-monograph publications.

Parameters

- **marc_xml** (*str*) – Original Aleph record.
- **mods** (*str*) – XML string generated by XSLT template.
- **uuid** (*str*) – UUID of the package.
- **counter** (*int*) – Number of record, is added to XML headers.

- **url** (*str*) – URL of the publication (public or not).

Returns Updated XML.

Return type str

1.1.5 periodical postprocessor

This module defines postprocessing for periodical publications.

marcxml2mods.mods_postprocessor.periodical.**postprocess_periodical**(**args*, ***kwargs*)

Some basic postprocessing of the periodical publications.

Parameters

- **marc_xml** (*str*) – Original Aleph record.
- **mods** (*str*) – XML string generated by XSLT template.
- **uuid** (*str*) – UUID of the package.
- **counter** (*int*) – Number of record, is added to XML headers.
- **url** (*str*) – URL of the publication (public or not).

Returns Updated XML.

Return type str

1.1.6 shared_funcs submodule

This module contains few functions shared across whole package.

marcxml2mods.mods_postprocessor.shared_funcs.**insert_tag**(*tag*, *before*, *root*)

Insert *tag* before *before* tag if present. If not, insert it into *root*.

Parameters

- **tag** (*obj*) – HTMLElement instance.
- **before** (*obj*) – HTMLElement instance.
- **root** (*obj*) – HTMLElement instance.

marcxml2mods.mods_postprocessor.shared_funcs.**transform_content**(*tags*, *content_transformer*)

Transform content in all *tags* using result of *content_transformer(tag)* call.

Parameters

- **tags** (*obj/list*) – HTMLElement instance, or list of HTMLElement instances.
- **content_transformer** (*function*) – Function which is called as `content_transformer(tag)`.

marcxml2mods.mods_postprocessor.shared_funcs.**double_linked_dom**(*str_or_dom*)

Create double linked DOM from input.

In case of string, parse it, make it double-linked. In case of DOM, just make it double-linked.

Parameters **str_or_dom** (*str/HTMLelement*) – String or HTMLelement instance.

Returns HTMLelement with parsed, double-linked content from *str_or_dom*.

Return type obj

`/api/xslt/xslt`:

1.1.7 MARC21slim2MODS3-4-NDK.xsl

This file is used to transform **monographic** publications.

1.1.8 MARC21toMultiMonographTitle.xsl

This file is used to transform **multi monographic** publications.

```
<xsl:stylesheet xmlns:mods="http://www.loc.gov/mods/v3" xmlns:marc="http://www.loc.gov/MARC21/slim"
        xmlns:xlink="http://www.w3.org/1999/xlink" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
        exclude-result-prefixes="xlink marc" version="1.0">
        <xsl:output encoding="UTF-8" indent="yes" method="xml"/>
        <xsl:strip-space elements="*"/>
        <xsl:namespace-alias stylesheet-prefix="mods" result-prefix="mods"/>

        <!-- Clone for generating multipart monograph TITLE element -->
        <!-- Author: Filip Majda, 2013 -->

        <!-- Maintenance note: For each revision, change the content of <mods:recordInfo><mods:recor
        MARC21slim2MODS3-4 (Revision 1.76) 20120201

REvision 1.77.NDK - Added full qualification mods: elements
Revision 1.76.NDK - Field 015 as CCNB
Revision 1.76 - Fixed 242 - 2012/02/01 tmee
Revision 1.75 - Fixed 653 - 2012/01/31 tmee
Revision 1.74 - Fixed 510 note - 2011/07/15 tmee
Revision 1.73 - Fixed 506 540 - 2011/07/11 tmee
Revision 1.72 - Fixed frequency error - 2011/07/07 and 2011/07/14 tmee
Revision 1.71 - Fixed subject titles for subfields t - 2011/04/26 tmee
Revision 1.70 - Added mapping for OCLC numbers in 035s to go into <mods:identifier type="oclc"> 2011/
Revision 1.69 - Added mapping for untyped identifiers for 024 - 2011/02/27 tmee
Revision 1.68 - Added <mods:subject><mods:titleInfo> mapping for 600/610/611 subfields t,p,n - 2010/1
Revision 1.67 - Added frequency values and authority="marcfrequency" for 008/18 - 2010/12/09 tmee
Revision 1.66 - Fixed 008/06=c,d,i,m,k,u, from dateCreated to dateIssued - 2010/12/06 tmee
Revision 1.65 - Added back marcsmd and marccategory for 007 cr- 2010/12/06 tmee
Revision 1.64 - Fixed identifiers - removed isInvalid template - 2010/12/06 tmee
Revision 1.63 - Fixed descriptiveStandard value from aacr2 to aacr - 2010/12/06 tmee
Revision 1.62 - Fixed date mapping for 008/06=e,p,r,s,t - 2010/12/01 tmee
Revision 1.61 - Added 007 mappings for marccategory - 2010/11/12 tmee
Revision 1.60 - Added altRepGroups and 880 linkages for relevant fields, see mapping - 2010/11/26 tme
Revision 1.59 - Added scriptTerm type=text to language for 546b and 066c - 2010/09/23 tmee
Revision 1.58 - Expanded script template to include code conversions for extended scripts - 2010/09/2
Revision 1.57 - Added Ldr/07 and Ldr/19 mappings - 2010/09/17 tmee
Revision 1.56 - Mapped 1xx usage="primary" - 2010/09/17 tmee
Revision 1.55 - Mapped UT 240/1xx nameTitleGroup - 2010/09/17 tmee
MODS 3.4
Revision 1.54 - Fixed 086 redundancy - 2010/07/27 tmee
Revision 1.53 - Added direct href for MARC21slimUtils - 2010/07/27 tmee
Revision 1.52 - Mapped 046 subfields c,e,k,l - 2010/04/09 tmee
Revision 1.51 - Corrected 856 transform - 2010/01/29 tmee
Revision 1.50 - Added 210 $2 authority attribute in <mods:titleInfo type=”abbreviated”> 2009/11/23 tm
```

```
Revision 1.49 - Aquifer revision 1.14 - Added 240s (version) data to <mods:titleInfo type="uniform"><
Revision 1.48 - Aquifer revision 1.27 - Added mapping of 242 second indicator (for nonfiling characte
Revision 1.47 - Aquifer revision 1.26 - Mapped 300 subfield f (type of unit) - and g (size of unit) 2
Revision 1.46 - Aquifer revision 1.25 - Changed mapping of 767 so that <type="otherVersion>  2009/11/
Revision 1.45 - Aquifer revision 1.24 - Changed mapping of 765 so that <type="otherVersion>  2009/11/
Revision 1.44 - Added <mods:recordInfo><mods:recordOrigin> canned text about the version of this styl
Revision 1.43 - Mapped 351 subfields a,b,c 2009/11/20 tmee
Revision 1.42 - Changed 856 second indicator=1 to go to <mods:location><mods:url displayLabel="electr
Revision 1.41 - Aquifer revision 1.9 Added variable and choice protocol for adding usage="primary dis
Revision 1.40 - Dropped <mods:note> for 510 and added <mods:relatedItem type="isReferencedBy"> for 51
Revision 1.39 - Aquifer revision 1.23 Changed mapping for 762 (Subseries Entry) from <mods:relatedIte
Revision 1.38 - Aquifer revision 1.29 Dropped 007s for electronic versions 2009/11/18 tmee
Revision 1.37 - Fixed date redundancy in output (with questionable dates) 2009/11/16 tmee
Revision 1.36 - If mss material (Ldr/06=d,p,f,t) map 008 dates and 260$c/$g dates to dateCreated 2009
Revision 1.35 - Mapped appended detailed dates from 008/07-10 and 008/11-14 to dateIssued or DateCrea
Revision 1.34 - Mapped 045b B.C. and C.E. date range info to iso8601-compliant dates in <mods:subject
Revision 1.33 - Mapped Ldr/06 "o" to <mods:typeOfResource>kit 2009/11/16 tmee
Revision 1.32 - Mapped specific note types from the MODS Note Type list <http://www.loc.gov/standards
Revision 1.31 - Mapped 540 to <mods:accessCondition type="use and reproduction"> and 506 to <mods:acc
Revision 1.30 - Mapped 037c to <mods:identifier displayLabel=""> 2009/11/13 tmee
Revision 1.29 - Corrected schemaLocation to 3.3 2009/11/13 tmee
Revision 1.28 - Changed mapping from 752,662 g going to mods:hierarchicalGeographic/area instead of "
Revision 1.27 - Mapped 648 to <mods:subject> 2009/03/13 tmee
Revision 1.26 - Added subfield $s mapping for 130/240/730  2008/10/16 tmee
Revision 1.25 - Mapped 040e to <mods:descriptiveStandard> and Leader/18 to <mods:descriptive standard
Revision 1.24 - Mapped 852 subfields $h, $i, $j, $k, $l, $m, $t to <mods:shelfLocation> and 852 subfi
Revision 1.23 - Commented out xlink/uri for subfield 0 for 130/240/730, 100/700, 110/710, 111/711 as
Revision 1.22 - Mapped 022 subfield $l to type "issn-l" subfield $m to output identifier element with
Revision 1.21 - Mapped 856 ind2=1 or ind2=2 to <mods:relatedItem><mods:location><mods:url>  2008/07/0
Revision 1.20 - Added genre w/@auth="contents of 2" and type= "musical composition"  2008/07/01 tmee
Revision 1.19 - Added genre offprint for 008/24+ BK code 2  2008/07/01  tmee
Revision 1.18 - Added xlink/uri for subfield 0 for 130/240/730, 100/700, 110/710, 111/711  2008/06/26
Revision 1.17 - Added mapping of 662 2008/05/14 tmee
Revision 1.16 - Changed @authority from "marc" to "marcgt" for 007 and 008 codes mapped to a term in
Revision 1.15 - For field 630, moved call to part template outside title element  2007/07/10 tmee
Revision 1.14 - Fixed template isValid and fields 010, 020, 022, 024, 028, and 037 to output addition
Revision 1.13 - Changed order of output under cartographics to reflect schema  2006/11/28 tmee
Revision 1.12 - Updated to reflect MODS 3.2 Mapping  2006/10/11 tmee
Revision 1.11 - The attribute objectPart moved from <mods:languageTerm> to <mods:language>  2006/04/0
Revision 1.10 - MODS 3.1 revisions to language and classification elements (plus ability to find marc
Revision 1.09 - Subfield $y was added to field 242 2004/09/02 10:57 jrad
Revision 1.08 - Subject chopPunctuation expanded and attribute fixes 2004/08/12 jrad
Revision 1.07 - 2004/03/25 08:29 jrad
Revision 1.06 - Various validation fixes 2004/02/20 ntra
Revision 1.05 - MODS2 to MODS3 updates, language unstacking and de-duping, chopPunctuation expanded
Revision 1.03 - Additional Changes not related to MODS Version 2.0 by ntra
Revision 1.02 - Added Log Comment  2003/03/24 19:37:42  ckeith
        -->

        <xsl:template match="/">
                <!-- <mods:mods xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" version="3.4"
                        xsi:schemaLocation="http://www.loc.gov/mods/v3 http://www.loc.gov/standards/m
                <mods:mods>
                        <xsl:for-each select="//marc:record">
                        <xsl:call-template name="marcRecord"/>
                        </xsl:for-each>
                </mods:mods>
        </xsl:template>
```

```
        <xsl:template name="marcRecord">
                <xsl:variable name="leader" select="marc:leader"/>
                <xsl:variable name="leader6" select="substring($leader,7,1)"/>
                <xsl:variable name="leader7" select="substring($leader,8,1)"/>
                <xsl:variable name="leader19" select="substring($leader,20,1)"/>
                <xsl:variable name="controlField008" select="marc:controlfield[@tag='008']"/>
                <xsl:variable name="typeOf008">
                        <xsl:choose>
                                <xsl:when test="$leader6='a'">
                                        <xsl:choose>
                                                <xsl:when
                                                        test="$leader7='a' or $leader7='c' or $leader
                                                        >BK</xsl:when>
                                                <xsl:when test="$leader7='b' or $leader7='i' or $lead
                                        </xsl:choose>
                                </xsl:when>
                                <xsl:when test="$leader6='t'">BK</xsl:when>
                                <xsl:when test="$leader6='p'">MM</xsl:when>
                                <xsl:when test="$leader6='m'">CF</xsl:when>
                                <xsl:when test="$leader6='e' or $leader6='f'">MP</xsl:when>
                                <xsl:when test="$leader6='g' or $leader6='k' or $leader6='o' or $lead
                                        >VM</xsl:when>
                                <xsl:when test="$leader6='c' or $leader6='d' or $leader6='i' or $lead
                                        >MU</xsl:when>
                        </xsl:choose>
                </xsl:variable>

                <!-- titleInfo -->

                <xsl:for-each select="marc:datafield[@tag='245']">
                        <xsl:call-template name="createTitleInfoFrom245"/>
                </xsl:for-each>

                <!-- Genre -->

                <mods:genre>title</mods:genre>

                <!-- Origin Info -->

                <mods:originInfo>

                        <xsl:for-each
                                select="marc:datafield[@tag=260]/marc:subfield[@code='b' or @code='c
                                <xsl:choose>
                                        <xsl:when test="@code='b'">
                                                <mods:publisher>
                                                        <xsl:call-template name="chopPunctuation">
                                                                <xsl:with-param name="chopString" sel
                                                                <xsl:with-param name="punctuation">
                                                                        <xsl:text>:,;/ </xsl:text>
                                                                </xsl:with-param>
                                                        </xsl:call-template>
                                                </mods:publisher>
                                        </xsl:when>
                                </xsl:choose>
                        </xsl:for-each>
```

```
                    <xsl:for-each select="marc:datafield[@tag=250]/marc:subfield[@code='a']">
                            <mods:edition>
                                    <xsl:value-of select="."/>
                            </mods:edition>
                    </xsl:for-each>

            </mods:originInfo>


            <!-- Identifiers -->

            <!-- 015, 856, 020, 024, 022, 028, 010, 035, 037 -->

            <xsl:for-each select="marc:datafield[@tag='015']">
                    <xsl:if test="marc:subfield[@code='a']">
                <mods:identifier type="ccnb">
                   <xsl:value-of select="marc:subfield[@code='a']"/>
                </mods:identifier>
            </xsl:if>
             <xsl:for-each select="marc:subfield[@code='z']">
                <mods:identifier type="ccnb" invalid="yes">
                            <xsl:value-of select="."/>
                </mods:identifier>
             </xsl:for-each>
            </xsl:for-each>
            <xsl:for-each select="marc:datafield[@tag='020']">
      <xsl:if test="marc:subfield[@code='a'] and not(starts-with(marc:subfield[@code='a'],'('))">
         <mods:identifier type="isbn">
            <xsl:call-template name="chopPunctuation">
               <xsl:with-param name="chopString" select="marc:subfield[@code='a']"/>
            </xsl:call-template>
         </mods:identifier>
      </xsl:if>
      <xsl:for-each select="marc:subfield[@code='z']">
         <xsl:if test="not(starts-with(.,'('))">
            <mods:identifier type="isbn" invalid="yes">
               <xsl:call-template name="chopPunctuation">
                  <xsl:with-param name="chopString" select="."/>
               </xsl:call-template>
            </mods:identifier>
         </xsl:if>
      </xsl:for-each>
   </xsl:for-each>

   <xsl:for-each select="marc:datafield[@tag='024'][@ind1='0']">
      <xsl:if test="marc:subfield[@code='a']">
         <mods:identifier type="isrc">
            <xsl:value-of select="marc:subfield[@code='a']"/>
         </mods:identifier>
      </xsl:if>
   </xsl:for-each>
   <xsl:for-each select="marc:datafield[@tag='024'][@ind1='2']">
      <xsl:if test="marc:subfield[@code='a'] and not(starts-with(marc:subfield[@code='a'],'('))">
         <mods:identifier type="ismn">
            <xsl:call-template name="chopPunctuation">
               <xsl:with-param name="chopString" select="marc:subfield[@code='a']"/>
            </xsl:call-template>
         </mods:identifier>
```

```
         </xsl:if>
         <xsl:for-each select="marc:subfield[@code='z']">
            <xsl:if test="not(starts-with(.,'('))">
               <mods:identifier type="ismn" invalid="yes">
                  <xsl:call-template name="chopPunctuation">
                     <xsl:with-param name="chopString" select="."/>
                  </xsl:call-template>
               </mods:identifier>
            </xsl:if>
         </xsl:for-each>
      </xsl:for-each>
      <xsl:for-each select="marc:datafield[@tag='024'][@ind1='4']">
         <mods:identifier type="sici">
            <xsl:call-template name="subfieldSelect">
               <xsl:with-param name="codes">ab</xsl:with-param>
            </xsl:call-template>
         </mods:identifier>
      </xsl:for-each>
      <xsl:for-each select="marc:datafield[@tag='024'][@ind1='8']">
         <mods:identifier>
            <xsl:value-of select="marc:subfield[@code='a']"/>
         </mods:identifier>
      </xsl:for-each>

      <xsl:for-each select="marc:datafield[@tag='022']">
         <xsl:if test="marc:subfield[@code='a']">
            <mods:identifier type="issn">
               <xsl:value-of select="marc:subfield[@code='a']"/>
            </mods:identifier>
         </xsl:if>
         <xsl:for-each select="marc:subfield[@code='z']">
            <mods:identifier type="issn" invalid="yes">
               <xsl:value-of select="."/>
            </mods:identifier>
         </xsl:for-each>
         <xsl:for-each select="marc:subfield[@code='y']">
            <mods:identifier type="issn" invalid="yes">
               <xsl:value-of select="."/>
            </mods:identifier>
         </xsl:for-each>
         <xsl:for-each select="marc:subfield[@code='l']">
            <mods:identifier type="issn-l">
               <xsl:value-of select="."/>
            </mods:identifier>
         </xsl:for-each>
         <xsl:for-each select="marc:subfield[@code='m']">
            <mods:identifier type="issn-l" invalid="yes">
               <xsl:value-of select="."/>
            </mods:identifier>
         </xsl:for-each>
      </xsl:for-each>

               <xsl:for-each select="marc:datafield[@tag='010'][marc:subfield[@code='a']]">
                       <mods:identifier type="lccn">
                               <xsl:value-of select="normalize-space(marc:subfield[@code='a'])"/>
                       </mods:identifier>
               </xsl:for-each>
               <xsl:for-each select="marc:datafield[@tag='010'][marc:subfield[@code='z']]">
```

```
                <mods:identifier type="lccn" invalid="yes">
                        <xsl:value-of select="normalize-space(marc:subfield[@code='z'])"/>
                </mods:identifier>
        </xsl:for-each>

        <xsl:for-each select="marc:datafield[@tag='028']">
                <mods:identifier>
                        <xsl:attribute name="type">
                                <xsl:choose>
                                        <xsl:when test="@ind1='0'">issue number</xsl:when>
                                        <xsl:when test="@ind1='1'">matrix number</xsl:when>
                                        <xsl:when test="@ind1='2'">music plate</xsl:when>
                                        <xsl:when test="@ind1='3'">music publisher</xsl:when>
                                        <xsl:when test="@ind1='4'">videorecording identifier<
                                </xsl:choose>
                        </xsl:attribute>
                        <xsl:call-template name="subfieldSelect">
                                <xsl:with-param name="codes">
                                        <xsl:choose>
                                                <xsl:when test="@ind1='0'">ba</xsl:when>
                                                <xsl:otherwise>ab</xsl:otherwise>
                                        </xsl:choose>
                                </xsl:with-param>
                        </xsl:call-template>
                </mods:identifier>
        </xsl:for-each>

        <xsl:for-each
                select="marc:datafield[@tag='035'][marc:subfield[@code='a'][contains(text(),
                <mods:identifier type="oclc">
                        <xsl:value-of
                                select="normalize-space(substring-after(marc:subfield[@code=
                </mods:identifier>
        </xsl:for-each>

        <xsl:for-each select="marc:datafield[@tag='037']">
                <mods:identifier type="stock number">
                        <xsl:if test="marc:subfield[@code='c']">
                                <xsl:attribute name="displayLabel">
                                        <xsl:call-template name="subfieldSelect">
                                                <xsl:with-param name="codes">c</xsl:with-para
                                        </xsl:call-template>
                                </xsl:attribute>
                        </xsl:if>
                        <xsl:call-template name="subfieldSelect">
                                <xsl:with-param name="codes">ab</xsl:with-param>
                        </xsl:call-template>
                </mods:identifier>
        </xsl:for-each>


        <!-- 1.51 tmee 20100129-->
        <xsl:for-each select="marc:datafield[@tag='856'][marc:subfield[@code='u']]">
                <xsl:if
                        test="starts-with(marc:subfield[@code='u'],'urn:hdl') or starts-with
                        <mods:identifier>
                                <xsl:attribute name="type">
                                        <xsl:if
```

```
                                                test="starts-with(marc:subfield[@code='u'],'u
                                                >doi</xsl:if>
                                        <xsl:if
                                                test="starts-with(marc:subfield[@code='u'],'u
                                                >hdl</xsl:if>
                                </xsl:attribute>
                                <xsl:value-of
                                        select="concat('hdl:',substring-after(marc:subfield[(
                                />
                        </mods:identifier>
                </xsl:if>


                <xsl:if
                        test="starts-with(marc:subfield[@code='u'],'urn:hdl') or starts-with
                        <mods:identifier type="hdl">
                                <xsl:if test="marc:subfield[@code='y' or @code='3' or @code=
                                        <xsl:attribute name="displayLabel">
                                                <xsl:call-template name="subfieldSelect">
                                                        <xsl:with-param name="codes">y3z</xsl
                                                </xsl:call-template>
                                        </xsl:attribute>
                                </xsl:if>
                                <xsl:value-of
                                        select="concat('hdl:',substring-after(marc:subfield[(
                                />
                        </mods:identifier>
                </xsl:if>
        </xsl:for-each>

        <xsl:for-each select="marc:datafield[@tag=024][@ind1=1]">
                <mods:identifier type="upc">
                        <xsl:value-of select="marc:subfield[@code='a']"/>
                </mods:identifier>
        </xsl:for-each>


        <!-- 1.51 tmee 20100129-->
        <xsl:for-each select="marc:datafield[@tag='856'][marc:subfield[@code='u']]">
                <xsl:if
                        test="starts-with(marc:subfield[@code='u'],'urn:hdl') or starts-with
                        <mods:identifier>
                                <xsl:attribute name="type">
                                        <xsl:if
                                                test="starts-with(marc:subfield[@code='u'],'u
                                                >doi</xsl:if>
                                        <xsl:if
                                                test="starts-with(marc:subfield[@code='u'],'u
                                                >hdl</xsl:if>
                                </xsl:attribute>
                                <xsl:value-of
                                        select="concat('hdl:',substring-after(marc:subfield[(
                                />
                        </mods:identifier>
                </xsl:if>

                <xsl:if
                        test="starts-with(marc:subfield[@code='u'],'urn:hdl') or starts-with
```

```
                        <mods:identifier type="hdl">
                                <xsl:if test="marc:subfield[@code='y' or @code='3' or @code=
                                        <xsl:attribute name="displayLabel">
                                                <xsl:call-template name="subfieldSelect">
                                                        <xsl:with-param name="codes">y3z</xsl
                                                </xsl:call-template>
                                        </xsl:attribute>
                                </xsl:if>
                                <xsl:value-of
                                        select="concat('hdl:',substring-after(marc:subfield[(
                                />
                        </mods:identifier>
                </xsl:if>
        </xsl:for-each>

</xsl:template>

<!-- The rest is completely duplicite with MARC21slim2MODS3-4-NDK.xsl -->

<xsl:template name="displayForm">
        <xsl:for-each select="marc:subfield[@code='c']">
                <mods:displayForm>
                        <xsl:value-of select="."/>
                </mods:displayForm>
        </xsl:for-each>
</xsl:template>
<xsl:template name="affiliation">
        <xsl:for-each select="marc:subfield[@code='u']">
                <mods:affiliation>
                        <xsl:value-of select="."/>
                </mods:affiliation>
        </xsl:for-each>
</xsl:template>
<xsl:template name="uri">
        <xsl:for-each select="marc:subfield[@code='u']|marc:subfield[@code='0']">
                <xsl:attribute name="xlink:href">
                        <xsl:value-of select="."/>
                </xsl:attribute>
        </xsl:for-each>
</xsl:template>
<xsl:template name="role">
        <!-- Not used in NK/MZK
        <xsl:for-each select="marc:subfield[@code='e']">
                <mods:role>
                        <mods:roleTerm type="text">
                                <xsl:value-of select="."/>
                        </mods:roleTerm>
                </mods:role>
        </xsl:for-each>
         -->
        <xsl:for-each select="marc:subfield[@code='4']">
                <mods:role>
                        <mods:roleTerm authority="marcrelator" type="code">
                                <xsl:value-of select="."/>
                        </mods:roleTerm>
                </mods:role>
        </xsl:for-each>
</xsl:template>
```

```
        <xsl:template name="part">
                <xsl:variable name="partNumber">
                        <xsl:call-template name="specialSubfieldSelect">
                                <xsl:with-param name="axis">n</xsl:with-param>
                                <xsl:with-param name="anyCodes">n</xsl:with-param>
                                <xsl:with-param name="afterCodes">fgkdlmor</xsl:with-param>
                        </xsl:call-template>
                </xsl:variable>
                <xsl:variable name="partName">
                        <xsl:call-template name="specialSubfieldSelect">
                                <xsl:with-param name="axis">p</xsl:with-param>
                                <xsl:with-param name="anyCodes">p</xsl:with-param>
                                <xsl:with-param name="afterCodes">fgkdlmor</xsl:with-param>
                        </xsl:call-template>
                </xsl:variable>
                <xsl:if test="string-length(normalize-space($partNumber))">
                        <mods:partNumber>
                                <xsl:call-template name="chopPunctuation">
                                        <xsl:with-param name="chopString" select="$partNumber"/>
                                </xsl:call-template>
                        </mods:partNumber>
                </xsl:if>
                <xsl:if test="string-length(normalize-space($partName))">
                        <mods:partName>
                                <xsl:call-template name="chopPunctuation">
                                        <xsl:with-param name="chopString" select="$partName"/>
                                </xsl:call-template>
                        </mods:partName>
                </xsl:if>
        </xsl:template>
        <xsl:template name="relatedPart">
                <xsl:if test="@tag=773">
                        <xsl:for-each select="marc:subfield[@code='g']">
                                <mods:part>
                                        <mods:text>
                                                <xsl:value-of select="."/>
                                        </mods:text>
                                </mods:part>
                        </xsl:for-each>
                        <xsl:for-each select="marc:subfield[@code='q']">
                                <mods:part>
                                        <xsl:call-template name="parsePart"/>
                                </mods:part>
                        </xsl:for-each>
                </xsl:if>
        </xsl:template>
        <xsl:template name="relatedPartNumName">
                <xsl:variable name="partNumber">
                        <xsl:call-template name="specialSubfieldSelect">
                                <xsl:with-param name="axis">g</xsl:with-param>
                                <xsl:with-param name="anyCodes">g</xsl:with-param>
                                <xsl:with-param name="afterCodes">pst</xsl:with-param>
                        </xsl:call-template>
                </xsl:variable>
                <xsl:variable name="partName">
                        <xsl:call-template name="specialSubfieldSelect">
                                <xsl:with-param name="axis">p</xsl:with-param>
                                <xsl:with-param name="anyCodes">p</xsl:with-param>
```

```
                        <xsl:with-param name="afterCodes">fgkdlmor</xsl:with-param>
                </xsl:call-template>
        </xsl:variable>
        <xsl:if test="string-length(normalize-space($partNumber))">
                <mods:partNumber>
                        <xsl:value-of select="$partNumber"/>
                </mods:partNumber>
        </xsl:if>
        <xsl:if test="string-length(normalize-space($partName))">
                <mods:partName>
                        <xsl:value-of select="$partName"/>
                </mods:partName>
        </xsl:if>
</xsl:template>
<xsl:template name="relatedName">
        <xsl:for-each select="marc:subfield[@code='a']">
                <mods:name>
                        <mods:namePart>
                                <xsl:value-of select="."/>
                        </mods:namePart>
                </mods:name>
        </xsl:for-each>
</xsl:template>
<xsl:template name="relatedForm">
        <xsl:for-each select="marc:subfield[@code='h']">
                <mods:physicalDescription>
                        <mods:form>
                                <xsl:value-of select="."/>
                        </mods:form>
                </mods:physicalDescription>
        </xsl:for-each>
</xsl:template>
<xsl:template name="relatedExtent">
        <xsl:for-each select="marc:subfield[@code='h']">
                <mods:physicalDescription>
                        <mods:extent>
                                <xsl:value-of select="."/>
                        </mods:extent>
                </mods:physicalDescription>
        </xsl:for-each>
</xsl:template>
<xsl:template name="relatedNote">
        <xsl:for-each select="marc:subfield[@code='n']">
                <mods:note>
                        <xsl:value-of select="."/>
                </mods:note>
        </xsl:for-each>
</xsl:template>
<xsl:template name="relatedSubject">
        <xsl:for-each select="marc:subfield[@code='j']">
                <mods:subject>
                        <mods:temporal encoding="iso8601">
                                <xsl:call-template name="chopPunctuation">
                                        <xsl:with-param name="chopString" select="."/>
                                </xsl:call-template>
                        </mods:temporal>
                </mods:subject>
        </xsl:for-each>
```

```
        </xsl:template>
        <xsl:template name="relatedIdentifierISSN">
                <xsl:for-each select="marc:subfield[@code='x']">
                        <mods:identifier type="issn">
                                <xsl:value-of select="."/>
                        </mods:identifier>
                </xsl:for-each>
        </xsl:template>
        <xsl:template name="relatedIdentifierLocal">
                <xsl:for-each select="marc:subfield[@code='w']">
                        <mods:identifier type="local">
                                <xsl:value-of select="."/>
                        </mods:identifier>
                </xsl:for-each>
        </xsl:template>
        <xsl:template name="relatedIdentifier">
                <xsl:for-each select="marc:subfield[@code='o']">
                        <mods:identifier>
                                <xsl:value-of select="."/>
                        </mods:identifier>
                </xsl:for-each>
        </xsl:template>
        <!--tmee 1.40 510 isReferencedBy -->
        <xsl:template name="relatedItem510">
                <xsl:call-template name="displayLabel"/>
                <xsl:call-template name="relatedTitle76X-78X"/>
                <xsl:call-template name="relatedName"/>
                <xsl:call-template name="relatedOriginInfo510"/>
                <xsl:call-template name="relatedLanguage"/>
                <xsl:call-template name="relatedExtent"/>
                <xsl:call-template name="relatedNote"/>
                <xsl:call-template name="relatedSubject"/>
                <xsl:call-template name="relatedIdentifier"/>
                <xsl:call-template name="relatedIdentifierISSN"/>
                <xsl:call-template name="relatedIdentifierLocal"/>
                <xsl:call-template name="relatedPart"/>
        </xsl:template>
        <xsl:template name="relatedItem76X-78X">
                <xsl:call-template name="displayLabel"/>
                <xsl:call-template name="relatedTitle76X-78X"/>
                <xsl:call-template name="relatedName"/>
                <xsl:call-template name="relatedOriginInfo"/>
                <xsl:call-template name="relatedLanguage"/>
                <xsl:call-template name="relatedExtent"/>
                <xsl:call-template name="relatedNote"/>
                <xsl:call-template name="relatedSubject"/>
                <xsl:call-template name="relatedIdentifier"/>
                <xsl:call-template name="relatedIdentifierISSN"/>
                <xsl:call-template name="relatedIdentifierLocal"/>
                <xsl:call-template name="relatedPart"/>
        </xsl:template>
        <xsl:template name="subjectGeographicZ">
                <mods:geographic>
                        <xsl:call-template name="chopPunctuation">
                                <xsl:with-param name="chopString" select="."/>
                        </xsl:call-template>
                </mods:geographic>
        </xsl:template>
```

```
        <xsl:template name="subjectTemporalY">
                <mods:temporal>
                        <xsl:call-template name="chopPunctuation">
                                <xsl:with-param name="chopString" select="."/>
                        </xsl:call-template>
                </mods:temporal>
        </xsl:template>
        <xsl:template name="subjectTopic">
                <mods:topic>
                        <xsl:call-template name="chopPunctuation">
                                <xsl:with-param name="chopString" select="."/>
                        </xsl:call-template>
                </mods:topic>
        </xsl:template>
        <!-- 3.2 change tmee 6xx $v genre -->
        <xsl:template name="subjectGenre">
                <mods:genre>
                        <xsl:call-template name="chopPunctuation">
                                <xsl:with-param name="chopString" select="."/>
                        </xsl:call-template>
                </mods:genre>
        </xsl:template>

        <xsl:template name="nameABCDN">
                <xsl:for-each select="marc:subfield[@code='a']">
                        <mods:namePart>
                                <xsl:call-template name="chopPunctuation">
                                        <xsl:with-param name="chopString" select="."/>
                                </xsl:call-template>
                        </mods:namePart>
                </xsl:for-each>
                <xsl:for-each select="marc:subfield[@code='b']">
                        <mods:namePart>
                                <xsl:value-of select="."/>
                        </mods:namePart>
                </xsl:for-each>
                <xsl:if
                        test="marc:subfield[@code='c'] or marc:subfield[@code='d'] or marc:subfield[(
                        <mods:namePart>
                                <xsl:call-template name="subfieldSelect">
                                        <xsl:with-param name="codes">cdn</xsl:with-param>
                                </xsl:call-template>
                        </mods:namePart>
                </xsl:if>
        </xsl:template>
        <xsl:template name="nameABCDQ">
                <mods:namePart>
                        <xsl:call-template name="chopPunctuation">
                                <xsl:with-param name="chopString">
                                        <xsl:call-template name="subfieldSelect">
                                                <xsl:with-param name="codes">aq</xsl:with-param>
                                        </xsl:call-template>
                                </xsl:with-param>
                                <xsl:with-param name="punctuation">
                                        <xsl:text>:,;/ </xsl:text>
                                </xsl:with-param>
                        </xsl:call-template>
                </mods:namePart>
```

```
                <xsl:call-template name="termsOfAddress"/>
                <xsl:call-template name="nameDate"/>
        </xsl:template>
        <xsl:template name="nameACDEQ">
                <mods:namePart>
                        <xsl:call-template name="subfieldSelect">
                                <xsl:with-param name="codes">acdneq</xsl:with-param>
                        </xsl:call-template>
                </mods:namePart>
        </xsl:template>
        <xsl:template name="constituentOrRelatedType">
                <xsl:if test="@ind2=2">
                        <xsl:attribute name="type">constituent</xsl:attribute>
                </xsl:if>
        </xsl:template>
        <xsl:template name="relatedTitle">
                <xsl:for-each select="marc:subfield[@code='t']">
                        <mods:titleInfo>
                                <mods:title>
                                        <xsl:call-template name="chopPunctuation">
                                                <xsl:with-param name="chopString">
                                                        <xsl:value-of select="."/>
                                                </xsl:with-param>
                                        </xsl:call-template>
                                </mods:title>
                        </mods:titleInfo>
                </xsl:for-each>
        </xsl:template>
        <xsl:template name="relatedTitle76X-78X">
                <xsl:for-each select="marc:subfield[@code='t']">
                        <mods:titleInfo>
                                <mods:title>
                                        <xsl:call-template name="chopPunctuation">
                                                <xsl:with-param name="chopString">
                                                        <xsl:value-of select="."/>
                                                </xsl:with-param>
                                        </xsl:call-template>
                                </mods:title>
                                <xsl:if test="marc:datafield[@tag!=773]and marc:subfield[@code='g']">
                                        <xsl:call-template name="relatedPartNumName"/>
                                </xsl:if>
                        </mods:titleInfo>
                </xsl:for-each>
                <xsl:for-each select="marc:subfield[@code='p']">
                        <mods:titleInfo type="abbreviated">
                                <mods:title>
                                        <xsl:call-template name="chopPunctuation">
                                                <xsl:with-param name="chopString">
                                                        <xsl:value-of select="."/>
                                                </xsl:with-param>
                                        </xsl:call-template>
                                </mods:title>
                                <xsl:if test="marc:datafield[@tag!=773]and marc:subfield[@code='g']">
                                        <xsl:call-template name="relatedPartNumName"/>
                                </xsl:if>
                        </mods:titleInfo>
                </xsl:for-each>
                <xsl:for-each select="marc:subfield[@code='s']">
```

```
                        <mods:titleInfo type="uniform">
                                <mods:title>
                                        <xsl:call-template name="chopPunctuation">
                                                <xsl:with-param name="chopString">
                                                        <xsl:value-of select="."/>
                                                </xsl:with-param>
                                        </xsl:call-template>
                                </mods:title>
                                <xsl:if test="marc:datafield[@tag!=773]and marc:subfield[@code='g']">
                                        <xsl:call-template name="relatedPartNumName"/>
                                </xsl:if>
                        </mods:titleInfo>
                </xsl:for-each>
        </xsl:template>
        <xsl:template name="relatedOriginInfo">
                <xsl:if test="marc:subfield[@code='b' or @code='d'] or marc:subfield[@code='f']">
                        <mods:originInfo>
                                <xsl:if test="@tag=775">
                                        <xsl:for-each select="marc:subfield[@code='f']">
                                                <mods:place>
                                                        <mods:placeTerm>
                                                                <xsl:attribute name="type">text</xsl
                                                                <xsl:attribute name="authority">marcg
                                                                <xsl:value-of select="."/>
                                                        </mods:placeTerm>
                                                </mods:place>
                                        </xsl:for-each>
                                </xsl:if>
                                <xsl:for-each select="marc:subfield[@code='d']">
                                        <mods:publisher>
                                                <xsl:value-of select="."/>
                                        </mods:publisher>
                                </xsl:for-each>
                                <xsl:for-each select="marc:subfield[@code='b']">
                                        <mods:edition>
                                                <xsl:value-of select="."/>
                                        </mods:edition>
                                </xsl:for-each>
                        </mods:originInfo>
                </xsl:if>
        </xsl:template>

        <!-- tmee 1.40 -->

        <xsl:template name="relatedOriginInfo510">
                <xsl:for-each select="marc:subfield[@code='b']">
                        <mods:originInfo>
                                <mods:dateOther type="coverage">
                                        <xsl:value-of select="."/>
                                </mods:dateOther>
                        </mods:originInfo>
                </xsl:for-each>
        </xsl:template>
        <xsl:template name="relatedLanguage">
                <xsl:for-each select="marc:subfield[@code='e']">
                        <xsl:call-template name="getLanguage">
                                <xsl:with-param name="langString">
                                        <xsl:value-of select="."/>
```

```
                        </xsl:with-param>
                </xsl:call-template>
        </xsl:for-each>
</xsl:template>
<xsl:template name="nameDate">
        <xsl:for-each select="marc:subfield[@code='d']">
                <mods:namePart type="date">
                        <xsl:call-template name="chopPunctuation">
                                <xsl:with-param name="chopString" select="."/>
                        </xsl:call-template>
                </mods:namePart>
        </xsl:for-each>
</xsl:template>
<xsl:template name="subjectAuthority">
        <xsl:if test="@ind2!=4">
                <xsl:if test="@ind2!=' '">
                        <xsl:if test="@ind2!=8">
                                <xsl:if test="@ind2!=9 or marc:subfield[@code='2'] != ''">
                                        <xsl:attribute name="authority">
                                                <xsl:choose>
                                                        <xsl:when test="@ind2=0">lcsh</xsl:wh
                                                        <xsl:when test="@ind2=1">lcshac</xsl
                                                        <xsl:when test="@ind2=2">mesh</xsl:wh
                                                        <!-- 1/04 fix -->
                                                        <xsl:when test="@ind2=3">nal</xsl:whe
                                                        <xsl:when test="@ind2=5">csh</xsl:whe
                                                        <xsl:when test="@ind2=6">rvm</xsl:whe
                                                        <xsl:when test="@ind2=7 or @ind2=9">
                                                                <xsl:value-of select="marc:su
                                                        </xsl:when>
                                                </xsl:choose>
                                        </xsl:attribute>
                                </xsl:if>
                        </xsl:if>
                </xsl:if>
        </xsl:if>
</xsl:template>
<!-- 1.75
        fix -->
<xsl:template name="subject653Type">
        <xsl:if test="@ind2!=' '">
                <xsl:if test="@ind2!='0'">
                        <xsl:if test="@ind2!='4'">
                                <xsl:if test="@ind2!='5'">
                                        <xsl:if test="@ind2!='6'">
                                                <xsl:if test="@ind2!='7'">
                                                        <xsl:if test="@ind2!='8'">
                                                                <xsl:if test="@ind2!='7'">
                                                                        <xsl:if test="@ind2!=
                                                                                <xsl:if test=
                                                                                        <xsl
                                                                                        <xsl
                                                                                        <xsl
                                                                                        <xsl
                                                                                        <xsl
                                                                                        </xsl
                                                                                        </xsl
                                                                </xsl:if>
```

```
                                                                                                </xsl:if>
                                                                                        </xsl:if>
                                                                                </xsl:if>
                                                                        </xsl:if>
                                                                </xsl:if>
                                                        </xsl:if>
                                                </xsl:if>
                                        </xsl:if>
                                </xsl:if>


        </xsl:template>
        <xsl:template name="subjectAnyOrder">
                <xsl:for-each select="marc:subfield[@code='v' or @code='x' or @code='y' or @code='z']
                        <xsl:choose>
                                <xsl:when test="@code='v'">
                                        <xsl:call-template name="subjectGenre"/>
                                </xsl:when>
                                <xsl:when test="@code='x'">
                                        <xsl:call-template name="subjectTopic"/>
                                </xsl:when>
                                <xsl:when test="@code='y'">
                                        <xsl:call-template name="subjectTemporalY"/>
                                </xsl:when>
                                <xsl:when test="@code='z'">
                                        <xsl:call-template name="subjectGeographicZ"/>
                                </xsl:when>
                        </xsl:choose>
                </xsl:for-each>
        </xsl:template>
        <xsl:template name="specialSubfieldSelect">
                <xsl:param name="anyCodes"/>
                <xsl:param name="axis"/>
                <xsl:param name="beforeCodes"/>
                <xsl:param name="afterCodes"/>
                <xsl:variable name="str">
                        <xsl:for-each select="marc:subfield">
                                <xsl:if
                                        test="contains($anyCodes, @code) or (contains($beforeCodes,@c
                                        <xsl:value-of select="text()"/>
                                        <xsl:text> </xsl:text>
                                </xsl:if>
                        </xsl:for-each>
                </xsl:variable>
                <xsl:value-of select="substring($str,1,string-length($str)-1)"/>
        </xsl:template>



        <xsl:template match="marc:datafield[@tag=656]">
                <mods:subject>
                        <xsl:call-template name="xxx880"/>
                        <xsl:if test="marc:subfield[@code=2]">
                                <xsl:attribute name="authority">
                                        <xsl:value-of select="marc:subfield[@code=2]"/>
                                </xsl:attribute>
                        </xsl:if>
                        <mods:occupation>
```

```
                        <xsl:call-template name="chopPunctuation">
                                <xsl:with-param name="chopString">
                                        <xsl:value-of select="marc:subfield[@code='a']"/>
                                </xsl:with-param>
                        </xsl:call-template>
                </mods:occupation>
        </mods:subject>
</xsl:template>
<xsl:template name="termsOfAddress">
        <xsl:if test="marc:subfield[@code='b' or @code='c']">
                <mods:namePart type="termsOfAddress">
                        <xsl:call-template name="chopPunctuation">
                                <xsl:with-param name="chopString">
                                        <xsl:call-template name="subfieldSelect">
                                                <xsl:with-param name="codes">bc</xsl:with-par
                                        </xsl:call-template>
                                </xsl:with-param>
                        </xsl:call-template>
                </mods:namePart>
        </xsl:if>
</xsl:template>
<xsl:template name="displayLabel">
        <xsl:if test="marc:subfield[@code='i']">
                <xsl:attribute name="displayLabel">
                        <xsl:value-of select="marc:subfield[@code='i']"/>
                </xsl:attribute>
        </xsl:if>
        <xsl:if test="marc:subfield[@code='3']">
                <xsl:attribute name="displayLabel">
                        <xsl:value-of select="marc:subfield[@code='3']"/>
                </xsl:attribute>
        </xsl:if>
</xsl:template>

<!-- isInvalid
<xsl:template name="isInvalid">
        <xsl:param name="type"/>
        <xsl:if
                test="marc:subfield[@code='z'] or marc:subfield[@code='y'] or marc:subfield[(
                <mods:identifier>
                        <xsl:attribute name="type">
                                <xsl:value-of select="$type"/>
                        </xsl:attribute>
                        <xsl:attribute name="invalid">
                                <xsl:text>yes</xsl:text>
                        </xsl:attribute>
                        <xsl:if test="marc:subfield[@code='z']">
                                <xsl:value-of select="marc:subfield[@code='z']"/>
                        </xsl:if>
                        <xsl:if test="marc:subfield[@code='y']">
                                <xsl:value-of select="marc:subfield[@code='y']"/>
                        </xsl:if>
                        <xsl:if test="marc:subfield[@code='m']">
                                <xsl:value-of select="marc:subfield[@code='m']"/>
                        </xsl:if>
                </mods:identifier>
        </xsl:if>
</xsl:template>
```

```
	-->
	<xsl:template name="subtitle">
		<xsl:if test="marc:subfield[@code='b']">
			<mods:subTitle>
				<xsl:call-template name="chopPunctuation">
					<xsl:with-param name="chopString">
						<xsl:value-of select="marc:subfield[@code='b']"/>
						<!--<xsl:call-template name="subfieldSelect">
							<xsl:with-param name="codes">b</xsl:with-para
						</xsl:call-template>-->
					</xsl:with-param>
				</xsl:call-template>
			</mods:subTitle>
		</xsl:if>
	</xsl:template>
	<xsl:template name="script">
		<xsl:param name="scriptCode"/>
		<xsl:attribute name="script">
			<xsl:choose>
				<!-- ISO 15924        and CJK is a local code        20101123-->
				<xsl:when test="$scriptCode='(3'">Arab</xsl:when>
				<xsl:when test="$scriptCode='(4'">Arab</xsl:when>
				<xsl:when test="$scriptCode='(B'">Latn</xsl:when>
				<xsl:when test="$scriptCode='!E'">Latn</xsl:when>
				<xsl:when test="$scriptCode='$1'">CJK</xsl:when>
				<xsl:when test="$scriptCode='(N'">Cyrl</xsl:when>
				<xsl:when test="$scriptCode='(Q'">Cyrl</xsl:when>
				<xsl:when test="$scriptCode='(2'">Hebr</xsl:when>
				<xsl:when test="$scriptCode='(S'">Grek</xsl:when>
			</xsl:choose>
		</xsl:attribute>
	</xsl:template>
	<xsl:template name="parsePart">
		<!-- assumes 773$q= 1:2:3<4
		     with up to 3 levels and one optional start page
		-->
		<xsl:variable name="level1">
			<xsl:choose>
				<xsl:when test="contains(text(),':')">
					<!-- 1:2 -->
					<xsl:value-of select="substring-before(text(),':')"/>
				</xsl:when>
				<xsl:when test="not(contains(text(),':'))">
					<!-- 1 or 1<3 -->
					<xsl:if test="contains(text(),'&lt;')">
						<!-- 1<3 -->
						<xsl:value-of select="substring-before(text(),'&lt;')
					</xsl:if>
					<xsl:if test="not(contains(text(),'&lt;'))">
						<!-- 1 -->
						<xsl:value-of select="text()"/>
					</xsl:if>
				</xsl:when>
			</xsl:choose>
		</xsl:variable>
		<xsl:variable name="sici2">
			<xsl:choose>
				<xsl:when test="starts-with(substring-after(text(),$level1),':')">
```

```
                                <xsl:value-of select="substring(substring-after(text(),$level
                        </xsl:when>
                        <xsl:otherwise>
                                <xsl:value-of select="substring-after(text(),$level1)"/>
                        </xsl:otherwise>
                </xsl:choose>
        </xsl:variable>
        <xsl:variable name="level2">
                <xsl:choose>
                        <xsl:when test="contains($sici2,':')">
                                <!--  2:3<4  -->
                                <xsl:value-of select="substring-before($sici2,':')"/>
                        </xsl:when>
                        <xsl:when test="contains($sici2,'&lt;')">
                                <!-- 1: 2<4 -->
                                <xsl:value-of select="substring-before($sici2,'&lt;')"/>
                        </xsl:when>
                        <xsl:otherwise>
                                <xsl:value-of select="$sici2"/>
                                <!-- 1:2 -->
                        </xsl:otherwise>
                </xsl:choose>
        </xsl:variable>
        <xsl:variable name="sici3">
                <xsl:choose>
                        <xsl:when test="starts-with(substring-after($sici2,$level2),':')">
                                <xsl:value-of select="substring(substring-after($sici2,$level
                        </xsl:when>
                        <xsl:otherwise>
                                <xsl:value-of select="substring-after($sici2,$level2)"/>
                        </xsl:otherwise>
                </xsl:choose>
        </xsl:variable>
        <xsl:variable name="level3">
                <xsl:choose>
                        <xsl:when test="contains($sici3,'&lt;')">
                                <!-- 2<4 -->
                                <xsl:value-of select="substring-before($sici3,'&lt;')"/>
                        </xsl:when>
                        <xsl:otherwise>
                                <xsl:value-of select="$sici3"/>
                                <!-- 3 -->
                        </xsl:otherwise>
                </xsl:choose>
        </xsl:variable>
        <xsl:variable name="page">
                <xsl:if test="contains(text(),'&lt;')">
                        <xsl:value-of select="substring-after(text(),'&lt;')"/>
                </xsl:if>
        </xsl:variable>
        <xsl:if test="$level1">
                <mods:detail level="1">
                        <mods:number>
                                <xsl:value-of select="$level1"/>
                        </mods:number>
                </mods:detail>
        </xsl:if>
        <xsl:if test="$level2">
```

```
                        <mods:detail level="2">
                                <mods:number>
                                        <xsl:value-of select="$level2"/>
                                </mods:number>
                        </mods:detail>
                </xsl:if>
                <xsl:if test="$level3">
                        <mods:detail level="3">
                                <mods:number>
                                        <xsl:value-of select="$level3"/>
                                </mods:number>
                        </mods:detail>
                </xsl:if>
                <xsl:if test="$page">
                        <mods:extent unit="page">
                                <mods:start>
                                        <xsl:value-of select="$page"/>
                                </mods:start>
                        </mods:extent>
                </xsl:if>
        </xsl:template>
        <xsl:template name="getLanguage">
                <xsl:param name="langString"/>
                <xsl:param name="controlField008-35-37"/>
                <xsl:variable name="length" select="string-length($langString)"/>
                <xsl:choose>
                        <xsl:when test="$length=0"/>
                        <xsl:when test="$controlField008-35-37=substring($langString,1,3)">
                                <xsl:call-template name="getLanguage">
                                        <xsl:with-param name="langString" select="substring($langStri
                                        <xsl:with-param name="controlField008-35-37" select="$control
                                </xsl:call-template>
                        </xsl:when>
                        <xsl:otherwise>
                                <mods:language>
                                        <mods:languageTerm authority="iso639-2b" type="code">
                                                <xsl:value-of select="substring($langString,1,3)"/>
                                        </mods:languageTerm>
                                </mods:language>
                                <xsl:call-template name="getLanguage">
                                        <xsl:with-param name="langString" select="substring($langStri
                                        <xsl:with-param name="controlField008-35-37" select="$control
                                </xsl:call-template>
                        </xsl:otherwise>
                </xsl:choose>
        </xsl:template>
        <xsl:template name="isoLanguage">
                <xsl:param name="currentLanguage"/>
                <xsl:param name="usedLanguages"/>
                <xsl:param name="remainingLanguages"/>
                <xsl:choose>
                        <xsl:when test="string-length($currentLanguage)=0"/>
                        <xsl:when test="not(contains($usedLanguages, $currentLanguage))">
                                <mods:language>
                                        <xsl:if test="@code!='a'">
                                                <xsl:attribute name="objectPart">
                                                        <xsl:choose>
                                                                <xsl:when test="@code='b'">summary or
```

```
                                                        <xsl:when test="@code='d'">sung or sp
                                                        <xsl:when test="@code='e'">libretto</
                                                        <xsl:when test="@code='f'">table of c
                                                        <xsl:when test="@code='g'">accompanyi
                                                        <xsl:when test="@code='h'">translatio
                                                </xsl:choose>
                                        </xsl:attribute>
                                </xsl:if>
                                <mods:languageTerm authority="iso639-2b" type="code">
                                        <xsl:value-of select="$currentLanguage"/>
                                </mods:languageTerm>
                        </mods:language>
                        <xsl:call-template name="isoLanguage">
                                <xsl:with-param name="currentLanguage">
                                        <xsl:value-of select="substring($remainingLanguages,1
                                </xsl:with-param>
                                <xsl:with-param name="usedLanguages">
                                        <xsl:value-of select="concat($usedLanguages,$currentL
                                </xsl:with-param>
                                <xsl:with-param name="remainingLanguages">
                                        <xsl:value-of
                                                select="substring($remainingLanguages,4,strin
                                        />
                                </xsl:with-param>
                        </xsl:call-template>
                </xsl:when>
                <xsl:otherwise>
                        <xsl:call-template name="isoLanguage">
                                <xsl:with-param name="currentLanguage">
                                        <xsl:value-of select="substring($remainingLanguages,1
                                </xsl:with-param>
                                <xsl:with-param name="usedLanguages">
                                        <xsl:value-of select="concat($usedLanguages,$currentL
                                </xsl:with-param>
                                <xsl:with-param name="remainingLanguages">
                                        <xsl:value-of
                                                select="substring($remainingLanguages,4,strin
                                        />
                                </xsl:with-param>
                        </xsl:call-template>
                </xsl:otherwise>
        </xsl:choose>
</xsl:template>
<xsl:template name="chopBrackets">
        <xsl:param name="chopString"/>
        <xsl:variable name="string">
                <xsl:call-template name="chopPunctuation">
                        <xsl:with-param name="chopString" select="$chopString"/>
                </xsl:call-template>
        </xsl:variable>
        <xsl:if test="substring($string, 1,1)='['">
                <xsl:value-of select="substring($string,2, string-length($string)-2)"/>
        </xsl:if>
        <xsl:if test="substring($string, 1,1)!='['">
                <xsl:value-of select="$string"/>
        </xsl:if>
</xsl:template>
<xsl:template name="rfcLanguages">
```

```
        <xsl:param name="nodeNum"/>
        <xsl:param name="usedLanguages"/>
        <xsl:param name="controlField008-35-37"/>
        <xsl:variable name="currentLanguage" select="."/>
        <xsl:choose>
                <xsl:when test="not($currentLanguage)"/>
                <xsl:when
                        test="$currentLanguage!=$controlField008-35-37 and $currentLanguage!=
                        <xsl:if test="not(contains($usedLanguages,$currentLanguage))">
                                <mods:language>
                                        <xsl:if test="@code!='a'">
                                                <xsl:attribute name="objectPart">
                                                        <xsl:choose>
                                                                <xsl:when test="@code='b'">su
                                                                <xsl:when test="@code='d'">su
                                                                <xsl:when test="@code='e'">li
                                                                <xsl:when test="@code='f'">ta
                                                                <xsl:when test="@code='g'">ac
                                                                <xsl:when test="@code='h'">tr
                                                        </xsl:choose>
                                                </xsl:attribute>
                                        </xsl:if>
                                        <mods:languageTerm authority="rfc3066" type="code">
                                                <xsl:value-of select="$currentLanguage"/>
                                        </mods:languageTerm>
                                </mods:language>
                        </xsl:if>
                </xsl:when>
                <xsl:otherwise> </xsl:otherwise>
        </xsl:choose>
</xsl:template>

<!-- tmee added 20100106 for 045$b BC and CE date range info -->
<xsl:template name="dates045b">
        <xsl:param name="str"/>
        <xsl:variable name="first-char" select="substring($str,1,1)"/>
        <xsl:choose>
                <xsl:when test="$first-char ='c'">
                        <xsl:value-of select="concat ('-', substring($str, 2))"/>
                </xsl:when>
                <xsl:when test="$first-char ='d'">
                        <xsl:value-of select="substring($str, 2)"/>
                </xsl:when>
                <xsl:otherwise>
                        <xsl:value-of select="$str"/>
                </xsl:otherwise>
        </xsl:choose>
</xsl:template>

<xsl:template name="scriptCode">
        <xsl:variable name="sf06" select="normalize-space(child::marc:subfield[@code='6'])"/>
        <xsl:variable name="sf06a" select="substring($sf06, 1, 3)"/>
        <xsl:variable name="sf06b" select="substring($sf06, 5, 2)"/>
        <xsl:variable name="sf06c" select="substring($sf06, 7)"/>
        <xsl:variable name="scriptCode" select="substring($sf06, 8, 2)"/>
        <xsl:if test="//marc:datafield/marc:subfield[@code='6']">
                <xsl:attribute name="script">
                        <xsl:choose>
```

```
                                        <xsl:when test="$scriptCode=''">Latn</xsl:when>
                                        <xsl:when test="$scriptCode='(3'">Arab</xsl:when>
                                        <xsl:when test="$scriptCode='(4'">Arab</xsl:when>
                                        <xsl:when test="$scriptCode='(B'">Latn</xsl:when>
                                        <xsl:when test="$scriptCode='!E'">Latn</xsl:when>
                                        <xsl:when test="$scriptCode='$1'">CJK</xsl:when>
                                        <xsl:when test="$scriptCode='(N'">Cyrl</xsl:when>
                                        <xsl:when test="$scriptCode='(Q'">Cyrl</xsl:when>
                                        <xsl:when test="$scriptCode='(2'">Hebr</xsl:when>
                                        <xsl:when test="$scriptCode='(S'">Grek</xsl:when>
                                </xsl:choose>
                        </xsl:attribute>
                </xsl:if>
        </xsl:template>

        <!-- tmee 20100927 for 880s & corresponding fields  20101123 scriptCode -->

        <xsl:template name="xxx880">
                <xsl:if test="child::marc:subfield[@code='6']">
                        <xsl:variable name="sf06" select="normalize-space(child::marc:subfield[@code=
                        <xsl:variable name="sf06a" select="substring($sf06, 1, 3)"/>
                        <xsl:variable name="sf06b" select="substring($sf06, 5, 2)"/>
                        <xsl:variable name="sf06c" select="substring($sf06, 7)"/>
                        <xsl:variable name="scriptCode" select="substring($sf06, 8, 2)"/>
                        <xsl:if test="//marc:datafield/marc:subfield[@code='6']">
                                <xsl:attribute name="altRepGroup">
                                        <xsl:value-of select="$sf06b"/>
                                </xsl:attribute>
                                <xsl:attribute name="script">
                                        <xsl:choose>
                                                <xsl:when test="$scriptCode=''">Latn</xsl:when>
                                                <xsl:when test="$scriptCode='(3'">Arab</xsl:when>
                                                <xsl:when test="$scriptCode='(4'">Arab</xsl:when>
                                                <xsl:when test="$scriptCode='(B'">Latn</xsl:when>
                                                <xsl:when test="$scriptCode='!E'">Latn</xsl:when>
                                                <xsl:when test="$scriptCode='$1'">CJK</xsl:when>
                                                <xsl:when test="$scriptCode='(N'">Cyrl</xsl:when>
                                                <xsl:when test="$scriptCode='(Q'">Cyrl</xsl:when>
                                                <xsl:when test="$scriptCode='(2'">Hebr</xsl:when>
                                                <xsl:when test="$scriptCode='(S'">Grek</xsl:when>
                                        </xsl:choose>
                                </xsl:attribute>
                        </xsl:if>
                </xsl:if>
        </xsl:template>

        <xsl:template name="yyy880">
                <xsl:if test="preceding-sibling::marc:subfield[@code='6']">
                        <xsl:variable name="sf06"
                                select="normalize-space(preceding-sibling::marc:subfield[@code='6'])"
                        <xsl:variable name="sf06a" select="substring($sf06, 1, 3)"/>
                        <xsl:variable name="sf06b" select="substring($sf06, 5, 2)"/>
                        <xsl:variable name="sf06c" select="substring($sf06, 7)"/>
                        <xsl:if test="//marc:datafield/marc:subfield[@code='6']">
                                <xsl:attribute name="altRepGroup">
                                        <xsl:value-of select="$sf06b"/>
                                </xsl:attribute>
                        </xsl:if>
```

```
                </xsl:if>
        </xsl:template>

        <xsl:template name="z2xx880">
                <!-- Evaluating the 260 field -->
                <xsl:variable name="x260">
                        <xsl:choose>
                                <xsl:when test="@tag='260' and marc:subfield[@code='6']">
                                        <xsl:variable name="sf06260"
                                                select="normalize-space(child::marc:subfield[@code='6
                                        <xsl:variable name="sf06260a" select="substring($sf06260, 1,
                                        <xsl:variable name="sf06260b" select="substring($sf06260, 5,
                                        <xsl:variable name="sf06260c" select="substring($sf06260, 7)"
                                        <xsl:value-of select="$sf06260b"/>
                                </xsl:when>
                                <xsl:when
                                        test="@tag='250' and ../marc:datafield[@tag='260']/marc:subfi
                                        <xsl:variable name="sf06260"
                                                select="normalize-space(../marc:datafield[@tag='260']
                                        <xsl:variable name="sf06260a" select="substring($sf06260, 1,
                                        <xsl:variable name="sf06260b" select="substring($sf06260, 5,
                                        <xsl:variable name="sf06260c" select="substring($sf06260, 7)"
                                        <xsl:value-of select="$sf06260b"/>
                                </xsl:when>
                        </xsl:choose>
                </xsl:variable>

                <xsl:variable name="x250">
                        <xsl:choose>
                                <xsl:when test="@tag='250' and marc:subfield[@code='6']">
                                        <xsl:variable name="sf06250"
                                                select="normalize-space(../marc:datafield[@tag='250']
                                        <xsl:variable name="sf06250a" select="substring($sf06250, 1,
                                        <xsl:variable name="sf06250b" select="substring($sf06250, 5,
                                        <xsl:variable name="sf06250c" select="substring($sf06250, 7)"
                                        <xsl:value-of select="$sf06250b"/>
                                </xsl:when>
                                <xsl:when
                                        test="@tag='260' and ../marc:datafield[@tag='250']/marc:subfi
                                        <xsl:variable name="sf06250"
                                                select="normalize-space(../marc:datafield[@tag='250']
                                        <xsl:variable name="sf06250a" select="substring($sf06250, 1,
                                        <xsl:variable name="sf06250b" select="substring($sf06250, 5,
                                        <xsl:variable name="sf06250c" select="substring($sf06250, 7)"
                                        <xsl:value-of select="$sf06250b"/>
                                </xsl:when>
                        </xsl:choose>
                </xsl:variable>

                <xsl:choose>
                        <xsl:when test="$x250!='' and $x260!=''">
                                <xsl:attribute name="altRepGroup">
                                        <xsl:value-of select="concat($x250, $x260)"/>
                                </xsl:attribute>
                        </xsl:when>
                        <xsl:when test="$x250!=''">
                                <xsl:attribute name="altRepGroup">
                                        <xsl:value-of select="$x250"/>
```

```
                        </xsl:attribute>
                    </xsl:when>
                    <xsl:when test="$x260!=''">
                        <xsl:attribute name="altRepGroup">
                            <xsl:value-of select="$x260"/>
                        </xsl:attribute>
                    </xsl:when>
                </xsl:choose>
                <xsl:if test="//marc:datafield/marc:subfield[@code='6']"> </xsl:if>
        </xsl:template>

        <xsl:template name="z3xx880">
                <!-- Evaluating the 300 field -->
                <xsl:variable name="x300">
                    <xsl:choose>
                        <xsl:when test="@tag='300' and marc:subfield[@code='6']">
                            <xsl:variable name="sf06300"
                                select="normalize-space(child::marc:subfield[@code='6
                            <xsl:variable name="sf06300a" select="substring($sf06300, 1,
                            <xsl:variable name="sf06300b" select="substring($sf06300, 5,
                            <xsl:variable name="sf06300c" select="substring($sf06300, 7)"
                            <xsl:value-of select="$sf06300b"/>
                        </xsl:when>
                        <xsl:when
                            test="@tag='351' and ../marc:datafield[@tag='300']/marc:subfi
                            <xsl:variable name="sf06300"
                                select="normalize-space(../marc:datafield[@tag='300']
                            <xsl:variable name="sf06300a" select="substring($sf06300, 1,
                            <xsl:variable name="sf06300b" select="substring($sf06300, 5,
                            <xsl:variable name="sf06300c" select="substring($sf06300, 7)"
                            <xsl:value-of select="$sf06300b"/>
                        </xsl:when>
                    </xsl:choose>
                </xsl:variable>

                <xsl:variable name="x351">
                    <xsl:choose>
                        <xsl:when test="@tag='351' and marc:subfield[@code='6']">
                            <xsl:variable name="sf06351"
                                select="normalize-space(../marc:datafield[@tag='351']
                            <xsl:variable name="sf06351a" select="substring($sf06351, 1,
                            <xsl:variable name="sf06351b" select="substring($sf06351, 5,
                            <xsl:variable name="sf06351c" select="substring($sf06351, 7)"
                            <xsl:value-of select="$sf06351b"/>
                        </xsl:when>
                        <xsl:when
                            test="@tag='300' and ../marc:datafield[@tag='351']/marc:subfi
                            <xsl:variable name="sf06351"
                                select="normalize-space(../marc:datafield[@tag='351']
                            <xsl:variable name="sf06351a" select="substring($sf06351, 1,
                            <xsl:variable name="sf06351b" select="substring($sf06351, 5,
                            <xsl:variable name="sf06351c" select="substring($sf06351, 7)"
                            <xsl:value-of select="$sf06351b"/>
                        </xsl:when>
                    </xsl:choose>
                </xsl:variable>

                <xsl:choose>
```

```
                        <xsl:when test="$x351!='' and $x300!=''">
                                <xsl:attribute name="altRepGroup">
                                        <xsl:value-of select="concat($x351, $x300)"/>
                                </xsl:attribute>
                        </xsl:when>
                        <xsl:when test="$x351!=''">
                                <xsl:attribute name="altRepGroup">
                                        <xsl:value-of select="$x351"/>
                                </xsl:attribute>
                        </xsl:when>
                        <xsl:when test="$x300!=''">
                                <xsl:attribute name="altRepGroup">
                                        <xsl:value-of select="$x300"/>
                                </xsl:attribute>
                        </xsl:when>
                </xsl:choose>
                <xsl:if test="//marc:datafield/marc:subfield[@code='6']"> </xsl:if>
        </xsl:template>



        <xsl:template name="true880">
                <xsl:variable name="sf06" select="normalize-space(marc:subfield[@code='6'])"/>
                <xsl:variable name="sf06a" select="substring($sf06, 1, 3)"/>
                <xsl:variable name="sf06b" select="substring($sf06, 5, 2)"/>
                <xsl:variable name="sf06c" select="substring($sf06, 7)"/>
                <xsl:if test="//marc:datafield/marc:subfield[@code='6']">
                        <xsl:attribute name="altRepGroup">
                                <xsl:value-of select="$sf06b"/>
                        </xsl:attribute>
                </xsl:if>
        </xsl:template>

        <xsl:template match="marc:datafield" mode="trans880">
                <xsl:variable name="dataField880" select="//marc:datafield"/>
                <xsl:variable name="sf06" select="normalize-space(marc:subfield[@code='6'])"/>
                <xsl:variable name="sf06a" select="substring($sf06, 1, 3)"/>
                <xsl:variable name="sf06b" select="substring($sf06, 4)"/>
                <xsl:choose>

                        <!--tranforms 880 equiv-->

                        <xsl:when test="$sf06a='047'">
                                <xsl:call-template name="createGenreFrom047"/>
                        </xsl:when>
                        <xsl:when test="$sf06a='655'">
                                <xsl:call-template name="createGenreFrom655"/>
                        </xsl:when>

                        <xsl:when test="$sf06a='050'">
                                <xsl:call-template name="createClassificationFrom050"/>
                        </xsl:when>
                        <xsl:when test="$sf06a='060'">
                                <xsl:call-template name="createClassificationFrom060"/>
                        </xsl:when>
                        <xsl:when test="$sf06a='080'">
                                <xsl:call-template name="createClassificationFrom080"/>
                        </xsl:when>
```

```
                        <xsl:when test="$sf06a='082'">
                                <xsl:call-template name="createClassificationFrom082"/>
                        </xsl:when>
                        <xsl:when test="$sf06a='084'">
                                <xsl:call-template name="createClassificationFrom080"/>
                        </xsl:when>
                        <xsl:when test="$sf06a='086'">
                                <xsl:call-template name="createClassificationFrom082"/>
                        </xsl:when>
                        <xsl:when test="$sf06a='100'">
                                <xsl:call-template name="createNameFrom100"/>
                        </xsl:when>
                        <xsl:when test="$sf06a='110'">
                                <xsl:call-template name="createNameFrom110"/>
                        </xsl:when>
                        <xsl:when test="$sf06a='111'">
                                <xsl:call-template name="createNameFrom110"/>
                        </xsl:when>
                        <xsl:when test="$sf06a='700'">
                                <xsl:call-template name="createNameFrom700"/>
                        </xsl:when>
                        <xsl:when test="$sf06a='710'">
                                <xsl:call-template name="createNameFrom710"/>
                        </xsl:when>
                        <xsl:when test="$sf06a='711'">
                                <xsl:call-template name="createNameFrom710"/>
                        </xsl:when>
                        <xsl:when test="$sf06a='210'">
                                <xsl:call-template name="createTitleInfoFrom210"/>
                        </xsl:when>
                        <xsl:when test="$sf06a='245'">
                                <xsl:call-template name="createTitleInfoFrom245"/>
                                <xsl:call-template name="createNoteFrom245c"/>
                        </xsl:when>
                        <xsl:when test="$sf06a='246'">
                                <xsl:call-template name="createTitleInfoFrom246"/>
                        </xsl:when>
                        <xsl:when test="$sf06a='240'">
                                <xsl:call-template name="createTitleInfoFrom240"/>
                        </xsl:when>
                        <xsl:when test="$sf06a='740'">
                                <xsl:call-template name="createTitleInfoFrom740"/>
                        </xsl:when>

                        <xsl:when test="$sf06a='130'">
                                <xsl:call-template name="createTitleInfoFrom130"/>
                        </xsl:when>
                        <xsl:when test="$sf06a='730'">
                                <xsl:call-template name="createTitleInfoFrom730"/>
                        </xsl:when>

                        <xsl:when test="$sf06a='505'">
                                <xsl:call-template name="createTOCFrom505"/>
                        </xsl:when>
                        <xsl:when test="$sf06a='520'">
                                <xsl:call-template name="createAbstractFrom520"/>
                        </xsl:when>
                        <xsl:when test="$sf06a='521'">
```

```
                <xsl:call-template name="createTargetAudienceFrom521"/>
        </xsl:when>
        <xsl:when test="$sf06a='506'">
                <xsl:call-template name="createAccessConditionFrom506"/>
        </xsl:when>
        <xsl:when test="$sf06a='540'">
                <xsl:call-template name="createAccessConditionFrom540"/>
        </xsl:when>

        <!-- note 245 362 etc        -->

        <xsl:when test="$sf06a='245'">
                <xsl:call-template name="createNoteFrom245c"/>
        </xsl:when>
        <xsl:when test="$sf06a='362'">
                <xsl:call-template name="createNoteFrom362"/>
        </xsl:when>
        <xsl:when test="$sf06a='502'">
                <xsl:call-template name="createNoteFrom502"/>
        </xsl:when>
        <xsl:when test="$sf06a='504'">
                <xsl:call-template name="createNoteFrom504"/>
        </xsl:when>
        <xsl:when test="$sf06a='508'">
                <xsl:call-template name="createNoteFrom508"/>
        </xsl:when>
        <xsl:when test="$sf06a='511'">
                <xsl:call-template name="createNoteFrom511"/>
        </xsl:when>
        <xsl:when test="$sf06a='515'">
                <xsl:call-template name="createNoteFrom515"/>
        </xsl:when>
        <xsl:when test="$sf06a='518'">
                <xsl:call-template name="createNoteFrom518"/>
        </xsl:when>
        <xsl:when test="$sf06a='524'">
                <xsl:call-template name="createNoteFrom524"/>
        </xsl:when>
        <xsl:when test="$sf06a='530'">
                <xsl:call-template name="createNoteFrom530"/>
        </xsl:when>
        <xsl:when test="$sf06a='533'">
                <xsl:call-template name="createNoteFrom533"/>
        </xsl:when>
        <xsl:when test="$sf06a='534'">
                <xsl:call-template name="createNoteFrom534"/>
        </xsl:when>
        <xsl:when test="$sf06a='535'">
                <xsl:call-template name="createNoteFrom535"/>
        </xsl:when>
        <xsl:when test="$sf06a='536'">
                <xsl:call-template name="createNoteFrom536"/>
        </xsl:when>
        <xsl:when test="$sf06a='538'">
                <xsl:call-template name="createNoteFrom538"/>
        </xsl:when>
        <xsl:when test="$sf06a='541'">
                <xsl:call-template name="createNoteFrom541"/>
```

```
                    </xsl:when>
                    <xsl:when test="$sf06a='545'">
                            <xsl:call-template name="createNoteFrom545"/>
                    </xsl:when>
                    <xsl:when test="$sf06a='546'">
                            <xsl:call-template name="createNoteFrom546"/>
                    </xsl:when>
                    <xsl:when test="$sf06a='561'">
                            <xsl:call-template name="createNoteFrom561"/>
                    </xsl:when>
                    <xsl:when test="$sf06a='562'">
                            <xsl:call-template name="createNoteFrom562"/>
                    </xsl:when>
                    <xsl:when test="$sf06a='581'">
                            <xsl:call-template name="createNoteFrom581"/>
                    </xsl:when>
                    <xsl:when test="$sf06a='583'">
                            <xsl:call-template name="createNoteFrom583"/>
                    </xsl:when>
                    <xsl:when test="$sf06a='585'">
                            <xsl:call-template name="createNoteFrom585"/>
                    </xsl:when>

                    <!--        note 5XX        -->

                    <xsl:when test="$sf06a='501'">
                            <xsl:call-template name="createNoteFrom5XX"/>
                    </xsl:when>
                    <xsl:when test="$sf06a='507'">
                            <xsl:call-template name="createNoteFrom5XX"/>
                    </xsl:when>
                    <xsl:when test="$sf06a='513'">
                            <xsl:call-template name="createNoteFrom5XX"/>
                    </xsl:when>
                    <xsl:when test="$sf06a='514'">
                            <xsl:call-template name="createNoteFrom5XX"/>
                    </xsl:when>
                    <xsl:when test="$sf06a='516'">
                            <xsl:call-template name="createNoteFrom5XX"/>
                    </xsl:when>
                    <xsl:when test="$sf06a='522'">
                            <xsl:call-template name="createNoteFrom5XX"/>
                    </xsl:when>
                    <xsl:when test="$sf06a='525'">
                            <xsl:call-template name="createNoteFrom5XX"/>
                    </xsl:when>
                    <xsl:when test="$sf06a='526'">
                            <xsl:call-template name="createNoteFrom5XX"/>
                    </xsl:when>
                    <xsl:when test="$sf06a='544'">
                            <xsl:call-template name="createNoteFrom5XX"/>
                    </xsl:when>
                    <xsl:when test="$sf06a='552'">
                            <xsl:call-template name="createNoteFrom5XX"/>
                    </xsl:when>
                    <xsl:when test="$sf06a='555'">
                            <xsl:call-template name="createNoteFrom5XX"/>
                    </xsl:when>
```

```
                    <xsl:when test="$sf06a='556'">
                            <xsl:call-template name="createNoteFrom5XX"/>
                    </xsl:when>
                    <xsl:when test="$sf06a='565'">
                            <xsl:call-template name="createNoteFrom5XX"/>
                    </xsl:when>
                    <xsl:when test="$sf06a='567'">
                            <xsl:call-template name="createNoteFrom5XX"/>
                    </xsl:when>
                    <xsl:when test="$sf06a='580'">
                            <xsl:call-template name="createNoteFrom5XX"/>
                    </xsl:when>
                    <xsl:when test="$sf06a='584'">
                            <xsl:call-template name="createNoteFrom5XX"/>
                    </xsl:when>
                    <xsl:when test="$sf06a='586'">
                            <xsl:call-template name="createNoteFrom5XX"/>
                    </xsl:when>

                    <!--  subject 034 043 045 255 656 662 752         -->

                    <xsl:when test="$sf06a='034'">
                            <xsl:call-template name="createSubGeoFrom034"/>
                    </xsl:when>
                    <xsl:when test="$sf06a='043'">
                            <xsl:call-template name="createSubGeoFrom043"/>
                    </xsl:when>
                    <xsl:when test="$sf06a='045'">
                            <xsl:call-template name="createSubTemFrom045"/>
                    </xsl:when>
                    <xsl:when test="$sf06a='255'">
                            <xsl:call-template name="createSubGeoFrom255"/>
                    </xsl:when>

                    <xsl:when test="$sf06a='600'">
                            <xsl:call-template name="createSubNameFrom600"/>
                    </xsl:when>
                    <xsl:when test="$sf06a='610'">
                            <xsl:call-template name="createSubNameFrom610"/>
                    </xsl:when>
                    <xsl:when test="$sf06a='611'">
                            <xsl:call-template name="createSubNameFrom611"/>
                    </xsl:when>

                    <xsl:when test="$sf06a='630'">
                            <xsl:call-template name="createSubTitleFrom630"/>
                    </xsl:when>

                    <xsl:when test="$sf06a='648'">
                            <xsl:call-template name="createSubChronFrom648"/>
                    </xsl:when>
                    <xsl:when test="$sf06a='650'">
                            <xsl:call-template name="createSubTopFrom650"/>
                    </xsl:when>
                    <xsl:when test="$sf06a='651'">
                            <xsl:call-template name="createSubGeoFrom651"/>
                    </xsl:when>
```

```
                    <xsl:when test="$sf06a='653'">
                            <xsl:call-template name="createSubFrom653"/>
                    </xsl:when>
                    <xsl:when test="$sf06a='656'">
                            <xsl:call-template name="createSubFrom656"/>
                    </xsl:when>
                    <xsl:when test="$sf06a='662'">
                            <xsl:call-template name="createSubGeoFrom662752"/>
                    </xsl:when>
                    <xsl:when test="$sf06a='752'">
                            <xsl:call-template name="createSubGeoFrom662752"/>
                    </xsl:when>

                    <!--  location  852 856 -->

                    <xsl:when test="$sf06a='852'">
                            <xsl:call-template name="createLocationFrom852"/>
                    </xsl:when>
                  <xsl:when test="$sf06a='910'">
                            <xsl:call-template name="createLocationFrom910" />
                    </xsl:when>
                    <xsl:when test="$sf06a='856'">
                            <xsl:call-template name="createLocationFrom856"/>
                    </xsl:when>

                    <xsl:when test="$sf06a='490'">
                            <xsl:call-template name="createRelatedItemFrom490"/>
                    </xsl:when>
            </xsl:choose>
    </xsl:template>

    <!-- titleInfo 130 730 245 246 240 740 210 -->

    <!-- 130 -->
    <xsl:template name="createTitleInfoFrom130">
            <xsl:for-each select="marc:datafield[@tag='130'][@ind2!='2']">
                    <mods:titleInfo type="uniform">
                            <mods:title>
                                    <xsl:variable name="str">
                                            <xsl:for-each select="marc:subfield">
                                                    <xsl:if test="(contains('s',@code))">
                                                            <xsl:value-of select="text()"/>
                                                            <xsl:text> </xsl:text>
                                                    </xsl:if>
                                                    <xsl:if
                                                            test="(contains('adfklmors',@code) ar
                                                            <xsl:value-of select="text()"/>
                                                            <xsl:text> </xsl:text>
                                                    </xsl:if>
                                            </xsl:for-each>
                                    </xsl:variable>
                                    <xsl:call-template name="chopPunctuation">
                                            <xsl:with-param name="chopString">
                                                    <xsl:value-of select="substring($str,1,string
                                            </xsl:with-param>
                                    </xsl:call-template>
                            </mods:title>
                            <xsl:call-template name="part"/>
```

```
                </mods:titleInfo>
        </xsl:for-each>
</xsl:template>
<xsl:template name="createTitleInfoFrom730">
        <mods:titleInfo type="uniform">
                <mods:title>
                        <xsl:variable name="str">
                                <xsl:for-each select="marc:subfield">
                                        <xsl:if test="(contains('s',@code))">
                                                <xsl:value-of select="text()"/>
                                                <xsl:text> </xsl:text>
                                        </xsl:if>
                                        <xsl:if
                                                test="(contains('adfklmors',@code) and (not(
                                                <xsl:value-of select="text()"/>
                                                <xsl:text> </xsl:text>
                                        </xsl:if>
                                </xsl:for-each>
                        </xsl:variable>
                        <xsl:call-template name="chopPunctuation">
                                <xsl:with-param name="chopString">
                                        <xsl:value-of select="substring($str,1,string-length
                                </xsl:with-param>
                        </xsl:call-template>

                </mods:title>
                <xsl:call-template name="part"/>
        </mods:titleInfo>
</xsl:template>

<xsl:template name="createTitleInfoFrom210">
        <mods:titleInfo type="abbreviated">
                <xsl:if test="marc:datafield[@tag='210'][@ind2='2']">
                        <xsl:attribute name="authority">
                                <xsl:value-of select="marc:subfield[@code='2']"/>
                        </xsl:attribute>
                </xsl:if>
                <xsl:call-template name="xxx880"/>
                <mods:title>
                        <xsl:call-template name="chopPunctuation">
                                <xsl:with-param name="chopString">
                                        <xsl:call-template name="subfieldSelect">
                                                <xsl:with-param name="codes">a</xsl:with-para
                                        </xsl:call-template>
                                </xsl:with-param>
                        </xsl:call-template>
                </mods:title>
                <xsl:call-template name="subtitle"/>
        </mods:titleInfo>
</xsl:template>

<xsl:template name="createTitleInfoFrom245">
        <mods:titleInfo>
                <xsl:call-template name="xxx880"/>
                <xsl:variable name="title">
                        <xsl:choose>
                                <xsl:when test="marc:subfield[@code='b']">
                                        <xsl:call-template name="specialSubfieldSelect">
```

```
                                                <xsl:with-param name="axis">b</xsl:with-paran
                                                <xsl:with-param name="beforeCodes">afgk</xsl
                                        </xsl:call-template>
                                </xsl:when>
                                <xsl:otherwise>
                                        <xsl:call-template name="subfieldSelect">
                                                <xsl:with-param name="codes">abfgk</xsl:with-
                                        </xsl:call-template>
                                </xsl:otherwise>
                        </xsl:choose>
                </xsl:variable>
                <xsl:variable name="titleChop">
                        <xsl:call-template name="chopPunctuation">
                                <xsl:with-param name="chopString">
                                        <xsl:value-of select="$title"/>
                                </xsl:with-param>
                        </xsl:call-template>
                </xsl:variable>
                <xsl:choose>
                        <xsl:when test="@ind2&gt;0">
                                <xsl:if test="@tag!='880'">
                                        <mods:nonSort>
                                                <xsl:value-of select="substring($titleChop,1,
                                        </mods:nonSort>
                                </xsl:if>
                                <mods:title>
                                        <xsl:value-of select="substring($titleChop,@ind2+1)"/
                                </mods:title>
                        </xsl:when>
                        <xsl:otherwise>
                                <mods:title>
                                        <xsl:value-of select="$titleChop"/>
                                </mods:title>
                        </xsl:otherwise>
                </xsl:choose>
                <xsl:if test="marc:subfield[@code='b']">
                        <mods:subTitle>
                                <xsl:call-template name="chopPunctuation">
                                        <xsl:with-param name="chopString">
                                                <xsl:call-template name="specialSubfieldSelec
                                                        <xsl:with-param name="axis">b</xsl:wi
                                                        <xsl:with-param name="anyCodes">b</xs
                                                        <xsl:with-param name="afterCodes">afg
                                                </xsl:call-template>
                                        </xsl:with-param>
                                </xsl:call-template>
                        </mods:subTitle>
                </xsl:if>
                <xsl:call-template name="part"/>
        </mods:titleInfo>
</xsl:template>

<xsl:template name="createTitleInfoFrom246">
        <mods:titleInfo type="alternative">
                <xsl:call-template name="xxx880"/>
                <xsl:for-each select="marc:subfield[@code='i']">
                        <xsl:attribute name="displayLabel">
                                <xsl:value-of select="text()"/>
```

```
                        </xsl:attribute>
                    </xsl:for-each>
                    <mods:title>
                        <xsl:call-template name="chopPunctuation">
                            <xsl:with-param name="chopString">
                                <xsl:call-template name="subfieldSelect">
                                    <!-- 1/04 removed $h, $b -->
                                    <xsl:with-param name="codes">af</xsl:with-par
                                </xsl:call-template>
                            </xsl:with-param>
                        </xsl:call-template>
                    </mods:title>
                    <xsl:call-template name="subtitle"/>
                    <xsl:call-template name="part"/>
            </mods:titleInfo>
        </xsl:template>

        <!-- 240 nameTitleGroup-->

        <xsl:template name="createTitleInfoFrom240">
            <mods:titleInfo type="uniform">
                <xsl:if
                    test="//marc:datafield[@tag='100']|//marc:datafield[@tag='110']|//mar
                    <xsl:attribute name="nameTitleGroup">
                        <xsl:text>1</xsl:text>
                    </xsl:attribute>
                </xsl:if>
                <xsl:call-template name="xxx880"/>
                <mods:title>
                    <xsl:variable name="str">
                        <xsl:for-each select="marc:subfield">
                            <xsl:if test="(contains('s',@code))">
                                <xsl:value-of select="text()"/>
                                <xsl:text> </xsl:text>
                            </xsl:if>
                            <xsl:if
                                test="(contains('adfklmors',@code) and (not(
                                <xsl:value-of select="text()"/>
                                <xsl:text> </xsl:text>
                            </xsl:if>
                        </xsl:for-each>
                    </xsl:variable>
                    <xsl:call-template name="chopPunctuation">
                        <xsl:with-param name="chopString">
                            <xsl:value-of select="substring($str,1,string-length
                        </xsl:with-param>
                    </xsl:call-template>
                </mods:title>
                <xsl:call-template name="part"/>
            </mods:titleInfo>
        </xsl:template>

        <xsl:template name="createTitleInfoFrom740">
            <mods:titleInfo type="alternative">
                <xsl:call-template name="xxx880"/>
                <mods:title>
                    <xsl:call-template name="chopPunctuation">
                        <xsl:with-param name="chopString">
```

```
                                        <xsl:call-template name="subfieldSelect">
                                                <xsl:with-param name="codes">ah</xsl:with-par
                                        </xsl:call-template>
                                </xsl:with-param>
                        </xsl:call-template>
                </mods:title>
                <xsl:call-template name="part"/>
        </mods:titleInfo>
</xsl:template>

<!-- name 100 110 111 -->

<xsl:template name="createNameFrom100">
        <mods:name type="personal">
                <xsl:attribute name="usage">
                        <xsl:text>primary</xsl:text>
                </xsl:attribute>
                <xsl:call-template name="xxx880"/>
                <xsl:if test="//marc:datafield[@tag='240']">
                        <xsl:attribute name="nameTitleGroup">
                                <xsl:text>1</xsl:text>
                        </xsl:attribute>
                </xsl:if>
                <xsl:call-template name="nameABCDQ"/>
                <xsl:call-template name="affiliation"/>
                <xsl:call-template name="role"/>
        </mods:name>
</xsl:template>

<xsl:template name="createNameFrom110">
        <mods:name type="corporate">
                <xsl:call-template name="xxx880"/>
                <xsl:if test="//marc:datafield[@tag='240']">
                        <xsl:attribute name="nameTitleGroup">
                                <xsl:text>1</xsl:text>
                        </xsl:attribute>
                </xsl:if>
                <xsl:call-template name="nameABCDN"/>
                <xsl:call-template name="role"/>
        </mods:name>
</xsl:template>

<xsl:template name="createNameFrom111">
        <mods:name type="conference">
                <xsl:call-template name="xxx880"/>
                <xsl:if test="//marc:datafield[@tag='240']">
                        <xsl:attribute name="nameTitleGroup">
                                <xsl:text>1</xsl:text>
                        </xsl:attribute>
                </xsl:if>
                <xsl:call-template name="nameACDEQ"/>
                <xsl:call-template name="role"/>
        </mods:name>
</xsl:template>

<!-- name 700 710 711 720 -->

<xsl:template name="createNameFrom700">
```

```
                <mods:name type="personal">
                        <xsl:call-template name="xxx880"/>
                        <xsl:call-template name="nameABCDQ"/>
                        <xsl:call-template name="affiliation"/>
                        <xsl:call-template name="role"/>
                </mods:name>
        </xsl:template>

        <xsl:template name="createNameFrom710">
                <mods:name type="corporate">
                        <xsl:call-template name="xxx880"/>
                        <xsl:call-template name="nameABCDN"/>
                        <xsl:call-template name="role"/>
                </mods:name>
        </xsl:template>

        <xsl:template name="createNameFrom711">
                <mods:name type="conference">
                        <xsl:call-template name="xxx880"/>
                        <xsl:call-template name="nameACDEQ"/>
                        <xsl:call-template name="role"/>
                </mods:name>
        </xsl:template>

        <xsl:template name="createNameFrom720">
                <xsl:if test="marc:datafield[@tag='720'][not(marc:subfield[@code='t'])]">
                        <mods:name>
                                <xsl:if test="@ind1=1">
                                        <xsl:attribute name="type">
                                                <xsl:text>personal</xsl:text>
                                        </xsl:attribute>
                                </xsl:if>
                                <mods:namePart>
                                        <xsl:value-of select="marc:subfield[@code='a']"/>
                                </mods:namePart>
                                <xsl:call-template name="role"/>
                        </mods:name>
                </xsl:if>
        </xsl:template>

        <!-- genre 047 655        -->

        <xsl:template name="createGenreFrom047">
                <mods:genre authority="marcgt">
                        <xsl:attribute name="authority">
                                <xsl:value-of select="marc:subfield[@code='2']"/>
                        </xsl:attribute>
                        <xsl:call-template name="xxx880"/>
                        <xsl:call-template name="subfieldSelect">
                                <xsl:with-param name="codes">abcdef</xsl:with-param>
                                <xsl:with-param name="delimeter">-</xsl:with-param>
                        </xsl:call-template>
                </mods:genre>
        </xsl:template>
        <xsl:template name="createGenreFrom655">
                <mods:genre authority="marcgt">
                        <xsl:attribute name="authority">
                                <xsl:value-of select="marc:subfield[@code='2']"/>
```

```
                </xsl:attribute>
                <xsl:call-template name="xxx880"/>
                <xsl:call-template name="subfieldSelect">
                        <xsl:with-param name="codes">abvxyz</xsl:with-param>
                        <xsl:with-param name="delimeter">-</xsl:with-param>
                </xsl:call-template>
        </mods:genre>
</xsl:template>

<!-- tOC 505 -->

<xsl:template name="createTOCFrom505">
        <mods:tableOfContents>
                <xsl:call-template name="xxx880"/>
                <xsl:call-template name="uri"/>
                <xsl:call-template name="subfieldSelect">
                        <xsl:with-param name="codes">agrt</xsl:with-param>
                </xsl:call-template>
        </mods:tableOfContents>
</xsl:template>

<!-- abstract 520 -->

<xsl:template name="createAbstractFrom520">
        <mods:abstract>
                <xsl:call-template name="xxx880"/>
                <xsl:call-template name="uri"/>
                <xsl:call-template name="subfieldSelect">
                        <xsl:with-param name="codes">ab</xsl:with-param>
                </xsl:call-template>
        </mods:abstract>
</xsl:template>

<!-- targetAudience 521 -->

<xsl:template name="createTargetAudienceFrom521">
        <mods:targetAudience>
                <xsl:call-template name="xxx880"/>
                <xsl:call-template name="subfieldSelect">
                        <xsl:with-param name="codes">ab</xsl:with-param>
                </xsl:call-template>
        </mods:targetAudience>
</xsl:template>

<!-- note 245c thru 585 -->

<xsl:template name="createNoteFrom245c">
        <xsl:choose>
                <xsl:when
                        test="//marc:datafield[@tag='245'] and //marc:datafield[@tag=880]/mar
                        <mods:note type="statement of responsibility">
                                <xsl:attribute name="altRepGroup">
                                        <xsl:text>00</xsl:text>
                                </xsl:attribute>
                                <xsl:call-template name="scriptCode"/>
                                <xsl:call-template name="subfieldSelect">
                                        <xsl:with-param name="codes">c</xsl:with-param>
                                </xsl:call-template>
```

```
                        </mods:note>
                </xsl:when>
                <xsl:when test="//marc:datafield[@tag='245']/marc:subfield[@code='c']">
                        <mods:note type="statement of responsibility">
                                <xsl:call-template name="scriptCode"/>
                                <xsl:call-template name="subfieldSelect">
                                        <xsl:with-param name="codes">c</xsl:with-param>
                                </xsl:call-template>
                        </mods:note>
                </xsl:when>
        </xsl:choose>
</xsl:template>

<xsl:template name="createNoteFrom362">
        <mods:note type="date/sequential designation">
                <xsl:call-template name="xxx880"/>
                <xsl:call-template name="uri"/>
                <xsl:variable name="str">
                        <xsl:for-each select="marc:subfield[@code!='6' and @code!='8']">
                                <xsl:value-of select="."/>
                                <xsl:text> </xsl:text>
                        </xsl:for-each>
                </xsl:variable>
                <xsl:value-of select="substring($str,1,string-length($str)-1)"/>
        </mods:note>
</xsl:template>

<xsl:template name="createNoteFrom500">
        <mods:note>
                <xsl:call-template name="xxx880"/>
                <xsl:call-template name="uri"/>
                <xsl:value-of select="marc:subfield[@code='a']"/>
        </mods:note>
</xsl:template>

<xsl:template name="createNoteFrom502">
        <mods:note type="thesis">
                <xsl:call-template name="xxx880"/>
                <xsl:call-template name="uri"/>
                <xsl:variable name="str">
                        <xsl:for-each select="marc:subfield[@code!='6' and @code!='8']">
                                <xsl:value-of select="."/>
                                <xsl:text> </xsl:text>
                        </xsl:for-each>
                </xsl:variable>
                <xsl:value-of select="substring($str,1,string-length($str)-1)"/>
        </mods:note>
</xsl:template>

<xsl:template name="createNoteFrom504">
        <mods:note type="bibliography">
                <xsl:call-template name="xxx880"/>
                <xsl:call-template name="uri"/>
                <xsl:variable name="str">
                        <xsl:for-each select="marc:subfield[@code!='6' and @code!='8']">
                                <xsl:value-of select="."/>
                                <xsl:text> </xsl:text>
                        </xsl:for-each>
```

```
                </xsl:variable>
                <xsl:value-of select="substring($str,1,string-length($str)-1)"/>
        </mods:note>
</xsl:template>

<xsl:template name="createNoteFrom508">
        <mods:note type="creation/production credits">
                <xsl:call-template name="xxx880"/>
                <xsl:call-template name="uri"/>
                <xsl:variable name="str">
                        <xsl:for-each
                                select="marc:subfield[@code!='u' and @code!='3' and @code!='6
                                <xsl:value-of select="."/>
                                <xsl:text> </xsl:text>
                        </xsl:for-each>
                </xsl:variable>
                <xsl:value-of select="substring($str,1,string-length($str)-1)"/>
        </mods:note>
</xsl:template>

<xsl:template name="createNoteFrom511">
        <mods:note type="performers">
                <xsl:call-template name="xxx880"/>
                <xsl:call-template name="uri"/>
                <xsl:variable name="str">
                        <xsl:for-each select="marc:subfield[@code!='6' and @code!='8']">
                                <xsl:value-of select="."/>
                                <xsl:text> </xsl:text>
                        </xsl:for-each>
                </xsl:variable>
                <xsl:value-of select="substring($str,1,string-length($str)-1)"/>
        </mods:note>
</xsl:template>

<xsl:template name="createNoteFrom515">
        <mods:note type="numbering">
                <xsl:call-template name="xxx880"/>
                <xsl:call-template name="uri"/>
                <xsl:variable name="str">
                        <xsl:for-each select="marc:subfield[@code!='6' and @code!='8']">
                                <xsl:value-of select="."/>
                                <xsl:text> </xsl:text>
                        </xsl:for-each>
                </xsl:variable>
                <xsl:value-of select="substring($str,1,string-length($str)-1)"/>
        </mods:note>
</xsl:template>

<xsl:template name="createNoteFrom518">
        <mods:note type="venue">
                <xsl:call-template name="xxx880"/>
                <xsl:call-template name="uri"/>
                <xsl:variable name="str">
                        <xsl:for-each select="marc:subfield[@code!='3' and @code!='6' and @co
                                <xsl:value-of select="."/>
                                <xsl:text> </xsl:text>
                        </xsl:for-each>
                </xsl:variable>
```

```
                    <xsl:value-of select="substring($str,1,string-length($str)-1)"/>
            </mods:note>
    </xsl:template>

    <xsl:template name="createNoteFrom524">
            <mods:note type="preferred citation">
                    <xsl:call-template name="xxx880"/>
                    <xsl:call-template name="uri"/>
                    <xsl:variable name="str">
                            <xsl:for-each select="marc:subfield[@code!='6' and @code!='8']">
                                    <xsl:value-of select="."/>
                                    <xsl:text> </xsl:text>
                            </xsl:for-each>
                    </xsl:variable>
                    <xsl:value-of select="substring($str,1,string-length($str)-1)"/>
            </mods:note>
    </xsl:template>

    <xsl:template name="createNoteFrom530">
            <mods:note type="additional physical form">
                    <xsl:call-template name="xxx880"/>
                    <xsl:call-template name="uri"/>
                    <xsl:variable name="str">
                            <xsl:for-each
                                    select="marc:subfield[@code!='u' and @code!='3' and @code!='6
                                    <xsl:value-of select="."/>
                                    <xsl:text> </xsl:text>
                            </xsl:for-each>
                    </xsl:variable>
                    <xsl:value-of select="substring($str,1,string-length($str)-1)"/>
            </mods:note>
    </xsl:template>

    <xsl:template name="createNoteFrom533">
            <mods:note type="reproduction">
                    <xsl:call-template name="xxx880"/>
                    <xsl:call-template name="uri"/>
                    <xsl:variable name="str">
                            <xsl:for-each select="marc:subfield[@code!='6' and @code!='8']">
                                    <xsl:value-of select="."/>
                                    <xsl:text> </xsl:text>
                            </xsl:for-each>
                    </xsl:variable>
                    <xsl:value-of select="substring($str,1,string-length($str)-1)"/>
            </mods:note>
    </xsl:template>

    <xsl:template name="createNoteFrom534">
            <mods:note type="original version">
                    <xsl:call-template name="xxx880"/>
                    <xsl:call-template name="uri"/>
                    <xsl:variable name="str">
                            <xsl:for-each select="marc:subfield[@code!='6' and @code!='8']">
                                    <xsl:value-of select="."/>
                                    <xsl:text> </xsl:text>
                            </xsl:for-each>
                    </xsl:variable>
                    <xsl:value-of select="substring($str,1,string-length($str)-1)"/>
```

```
        </mods:note>
    </xsl:template>

    <xsl:template name="createNoteFrom535">
        <mods:note type="original location">
            <xsl:call-template name="xxx880"/>
            <xsl:call-template name="uri"/>
            <xsl:variable name="str">
                <xsl:for-each select="marc:subfield[@code!='6' and @code!='8']">
                    <xsl:value-of select="."/>
                    <xsl:text> </xsl:text>
                </xsl:for-each>
            </xsl:variable>
            <xsl:value-of select="substring($str,1,string-length($str)-1)"/>
        </mods:note>
    </xsl:template>

    <xsl:template name="createNoteFrom536">
        <mods:note type="funding">
            <xsl:call-template name="xxx880"/>
            <xsl:call-template name="uri"/>
            <xsl:variable name="str">
                <xsl:for-each select="marc:subfield[@code!='6' and @code!='8']">
                    <xsl:value-of select="."/>
                    <xsl:text> </xsl:text>
                </xsl:for-each>
            </xsl:variable>
            <xsl:value-of select="substring($str,1,string-length($str)-1)"/>
        </mods:note>
    </xsl:template>

    <xsl:template name="createNoteFrom538">
        <mods:note type="system details">
            <xsl:call-template name="xxx880"/>
            <xsl:call-template name="uri"/>
            <xsl:variable name="str">
                <xsl:for-each select="marc:subfield[@code!='6' and @code!='8']">
                    <xsl:value-of select="."/>
                    <xsl:text> </xsl:text>
                </xsl:for-each>
            </xsl:variable>
            <xsl:value-of select="substring($str,1,string-length($str)-1)"/>
        </mods:note>
    </xsl:template>

    <xsl:template name="createNoteFrom541">
        <mods:note type="acquisition">
            <xsl:call-template name="xxx880"/>
            <xsl:call-template name="uri"/>
            <xsl:variable name="str">
                <xsl:for-each select="marc:subfield[@code!='6' and @code!='8']">
                    <xsl:value-of select="."/>
                    <xsl:text> </xsl:text>
                </xsl:for-each>
            </xsl:variable>
            <xsl:value-of select="substring($str,1,string-length($str)-1)"/>
        </mods:note>
    </xsl:template>
```

```
	<xsl:template name="createNoteFrom545">
		<mods:note type="biographical/historical">
			<xsl:call-template name="xxx880"/>
			<xsl:call-template name="uri"/>
			<xsl:variable name="str">
				<xsl:for-each select="marc:subfield[@code!='6' and @code!='8']">
					<xsl:value-of select="."/>
					<xsl:text> </xsl:text>
				</xsl:for-each>
			</xsl:variable>
			<xsl:value-of select="substring($str,1,string-length($str)-1)"/>
		</mods:note>
	</xsl:template>

	<xsl:template name="createNoteFrom546">
		<mods:note type="language">
			<xsl:call-template name="xxx880"/>
			<xsl:call-template name="uri"/>
			<xsl:variable name="str">
				<xsl:for-each select="marc:subfield[@code!='6' and @code!='8']">
					<xsl:value-of select="."/>
					<xsl:text> </xsl:text>
				</xsl:for-each>
			</xsl:variable>
			<xsl:value-of select="substring($str,1,string-length($str)-1)"/>
		</mods:note>
	</xsl:template>

	<xsl:template name="createNoteFrom561">
		<mods:note type="ownership">
			<xsl:call-template name="xxx880"/>
			<xsl:call-template name="uri"/>
			<xsl:variable name="str">
				<xsl:for-each select="marc:subfield[@code!='6' and @code!='8']">
					<xsl:value-of select="."/>
					<xsl:text> </xsl:text>
				</xsl:for-each>
			</xsl:variable>
			<xsl:value-of select="substring($str,1,string-length($str)-1)"/>
		</mods:note>
	</xsl:template>

	<xsl:template name="createNoteFrom562">
		<mods:note type="version identification">
			<xsl:call-template name="xxx880"/>
			<xsl:call-template name="uri"/>
			<xsl:variable name="str">
				<xsl:for-each select="marc:subfield[@code!='6' and @code!='8']">
					<xsl:value-of select="."/>
					<xsl:text> </xsl:text>
				</xsl:for-each>
			</xsl:variable>
			<xsl:value-of select="substring($str,1,string-length($str)-1)"/>
		</mods:note>
	</xsl:template>

	<xsl:template name="createNoteFrom581">
		<mods:note type="publications">
```

```
                        <xsl:call-template name="xxx880"/>
                        <xsl:call-template name="uri"/>
                        <xsl:variable name="str">
                                <xsl:for-each select="marc:subfield[@code!='6' and @code!='8']">
                                        <xsl:value-of select="."/>
                                        <xsl:text> </xsl:text>
                                </xsl:for-each>
                        </xsl:variable>
                        <xsl:value-of select="substring($str,1,string-length($str)-1)"/>
                </mods:note>
        </xsl:template>

        <xsl:template name="createNoteFrom583">
                <mods:note type="action">
                        <xsl:call-template name="xxx880"/>
                        <xsl:call-template name="uri"/>
                        <xsl:variable name="str">
                                <xsl:for-each select="marc:subfield[@code!='6' and @code!='8']">
                                        <xsl:value-of select="."/>
                                        <xsl:text> </xsl:text>
                                </xsl:for-each>
                        </xsl:variable>
                        <xsl:value-of select="substring($str,1,string-length($str)-1)"/>
                </mods:note>
        </xsl:template>

        <xsl:template name="createNoteFrom585">
                <mods:note type="exhibitions">
                        <xsl:call-template name="xxx880"/>
                        <xsl:call-template name="uri"/>
                        <xsl:variable name="str">
                                <xsl:for-each select="marc:subfield[@code!='6' and @code!='8']">
                                        <xsl:value-of select="."/>
                                        <xsl:text> </xsl:text>
                                </xsl:for-each>
                        </xsl:variable>
                        <xsl:value-of select="substring($str,1,string-length($str)-1)"/>
                </mods:note>
        </xsl:template>

        <xsl:template name="createNoteFrom5XX">
                <mods:note>
                        <xsl:call-template name="xxx880"/>
                        <xsl:call-template name="uri"/>
                        <xsl:variable name="str">
                                <xsl:for-each select="marc:subfield[@code!='6' and @code!='8']">
                                        <xsl:value-of select="."/>
                                        <xsl:text> </xsl:text>
                                </xsl:for-each>
                        </xsl:variable>
                        <xsl:value-of select="substring($str,1,string-length($str)-1)"/>
                </mods:note>
        </xsl:template>

        <!-- subject Geo 034 043 045 255 656 662 752 -->

        <xsl:template name="createSubGeoFrom034">
                <xsl:if
```

```
                test="marc:datafield[@tag=034][marc:subfield[@code='d' or @code='e' or @code=
                <mods:subject>
                        <xsl:call-template name="xxx880"/>
                        <mods:cartographics>
                                <mods:coordinates>
                                        <xsl:call-template name="subfieldSelect">
                                                <xsl:with-param name="codes">defg</xsl:with-p
                                        </xsl:call-template>
                                </mods:coordinates>
                        </mods:cartographics>
                </mods:subject>
        </xsl:if>
</xsl:template>

<xsl:template name="createSubGeoFrom043">
        <mods:subject>
                <xsl:call-template name="xxx880"/>
                <xsl:for-each select="marc:subfield[@code='a' or @code='b' or @code='c']">
                        <mods:geographicCode>
                                <xsl:attribute name="authority">
                                        <xsl:if test="@code='a'">
                                                <xsl:text>marcgac</xsl:text>
                                        </xsl:if>
                                        <xsl:if test="@code='b'">
                                                <xsl:value-of select="following-sibling::marc
                                        </xsl:if>
                                        <xsl:if test="@code='c'">
                                                <xsl:text>iso3166</xsl:text>
                                        </xsl:if>
                                </xsl:attribute>
                                <xsl:value-of select="self::marc:subfield"/>
                        </mods:geographicCode>
                </xsl:for-each>
        </mods:subject>
</xsl:template>

<xsl:template name="createSubGeoFrom255">
        <mods:subject>
                <xsl:call-template name="xxx880"/>
                <xsl:for-each select="marc:subfield[@code='a' or @code='b' or @code='c']">
                        <mods:cartographics>
                                <xsl:if test="@code='a'">
                                        <mods:scale>
                                                <xsl:value-of select="."/>
                                        </mods:scale>
                                </xsl:if>
                                <xsl:if test="@code='b'">
                                        <mods:projection>
                                                <xsl:value-of select="."/>
                                        </mods:projection>
                                </xsl:if>
                                <xsl:if test="@code='c'">
                                        <mods:coordinates>
                                                <xsl:value-of select="."/>
                                        </mods:coordinates>
                                </xsl:if>
                        </mods:cartographics>
                </xsl:for-each>
```

```
        </mods:subject>
    </xsl:template>

    <xsl:template name="createSubNameFrom600">
        <mods:subject>
            <xsl:call-template name="xxx880"/>
            <xsl:call-template name="subjectAuthority"/>
            <mods:name type="personal">
                <xsl:call-template name="termsOfAddress"/>
                <mods:namePart>
                    <xsl:call-template name="chopPunctuation">
                        <xsl:with-param name="chopString">
                            <xsl:call-template name="subfieldSelect">
                                <xsl:with-param name="codes">aq</xsl
                            </xsl:call-template>
                        </xsl:with-param>
                    </xsl:call-template>
                </mods:namePart>
                <xsl:call-template name="nameDate"/>
                <xsl:call-template name="affiliation"/>
                <xsl:call-template name="role"/>
            </mods:name>
            <xsl:if test="marc:subfield[@code='t']">
                <mods:titleInfo>
                    <mods:title>
                        <xsl:call-template name="chopPunctuation">
                            <xsl:with-param name="chopString">
                                <xsl:call-template name="subfieldSele
                                    <xsl:with-param name="codes">
                                </xsl:call-template>
                            </xsl:with-param>
                        </xsl:call-template>
                    </mods:title>
                    <xsl:call-template name="part"/>
                </mods:titleInfo>
            </xsl:if>
            <xsl:call-template name="subjectAnyOrder"/>
        </mods:subject>
    </xsl:template>

    <xsl:template name="createSubNameFrom610">
        <mods:subject>
            <xsl:call-template name="xxx880"/>
            <xsl:call-template name="subjectAuthority"/>
            <mods:name type="corporate">
                <xsl:for-each select="marc:subfield[@code='a']">
                    <mods:namePart>
                        <xsl:value-of select="."/>
                    </mods:namePart>
                </xsl:for-each>
                <xsl:for-each select="marc:subfield[@code='b']">
                    <mods:namePart>
                        <xsl:value-of select="."/>
                    </mods:namePart>
                </xsl:for-each>
                <xsl:if test="marc:subfield[@code='c' or @code='d' or @code='n' or @c
                    <mods:namePart>
                        <xsl:call-template name="subfieldSelect">
```

```
                                                <xsl:with-param name="codes">cdnp</xsl:with-p
                                        </xsl:call-template>
                                </mods:namePart>
                        </xsl:if>
                        <xsl:call-template name="role"/>
                </mods:name>
                <xsl:if test="marc:subfield[@code='t']">
                        <mods:titleInfo>
                                <mods:title>
                                        <xsl:call-template name="chopPunctuation">
                                                <xsl:with-param name="chopString">
                                                        <xsl:call-template name="subfieldSele
                                                                <xsl:with-param name="codes">
                                                        </xsl:call-template>
                                                </xsl:with-param>
                                        </xsl:call-template>
                                </mods:title>
                                <xsl:call-template name="part"/>
                        </mods:titleInfo>
                </xsl:if>
                <xsl:call-template name="subjectAnyOrder"/>
        </mods:subject>
</xsl:template>

<xsl:template name="createSubNameFrom611">
        <mods:subject>
                <xsl:call-template name="xxx880"/>
                <xsl:call-template name="subjectAuthority"/>
                <mods:name type="conference">
                        <mods:namePart>
                                <xsl:call-template name="subfieldSelect">
                                        <xsl:with-param name="codes">abcdeqnp</xsl:with-paran
                                </xsl:call-template>
                        </mods:namePart>
                        <xsl:for-each select="marc:subfield[@code='4']">
                                <mods:role>
                                        <mods:roleTerm authority="marcrelator" type="code">
                                                <xsl:value-of select="."/>
                                        </mods:roleTerm>
                                </mods:role>
                        </xsl:for-each>
                </mods:name>
                <xsl:if test="marc:subfield[@code='t']">
                        <mods:titleInfo>
                                <mods:title>
                                        <xsl:call-template name="chopPunctuation">
                                                <xsl:with-param name="chopString">
                                                        <xsl:call-template name="subfieldSele
                                                                <xsl:with-param name="codes">
                                                        </xsl:call-template>
                                                </xsl:with-param>
                                        </xsl:call-template>
                                </mods:title>
                                <xsl:call-template name="part"/>
                        </mods:titleInfo>
                </xsl:if>
                <xsl:call-template name="subjectAnyOrder"/>
        </mods:subject>
```

```
        </xsl:template>

        <xsl:template name="createSubTitleFrom630">
                <mods:subject>
                        <xsl:call-template name="xxx880"/>
                        <xsl:call-template name="subjectAuthority"/>
                        <mods:titleInfo>
                                <mods:title>
                                        <xsl:call-template name="chopPunctuation">
                                                <xsl:with-param name="chopString">
                                                        <xsl:call-template name="subfieldSelect">
                                                                <xsl:with-param name="codes">adfhklor
                                                        </xsl:call-template>
                                                </xsl:with-param>
                                        </xsl:call-template>
                                </mods:title>
                                <xsl:call-template name="part"/>
                        </mods:titleInfo>
                        <xsl:call-template name="subjectAnyOrder"/>
                </mods:subject>
        </xsl:template>

        <xsl:template name="createSubChronFrom648">
                <mods:subject>
                        <xsl:call-template name="xxx880"/>
                        <xsl:if test="marc:subfield[@code=2]">
                                <xsl:attribute name="authority">
                                        <xsl:value-of select="marc:subfield[@code=2]"/>
                                </xsl:attribute>
                        </xsl:if>
                        <xsl:call-template name="uri"/>
                        <xsl:call-template name="subjectAuthority"/>
                        <mods:temporal>
                                <xsl:call-template name="chopPunctuation">
                                        <xsl:with-param name="chopString">
                                                <xsl:call-template name="subfieldSelect">
                                                        <xsl:with-param name="codes">abcd</xsl:with-p
                                                </xsl:call-template>
                                        </xsl:with-param>
                                </xsl:call-template>
                        </mods:temporal>
                        <xsl:call-template name="subjectAnyOrder"/>
                </mods:subject>
        </xsl:template>

        <xsl:template name="createSubTopFrom650">
                <mods:subject>
                        <xsl:call-template name="xxx880"/>
                        <xsl:call-template name="subjectAuthority"/>
                        <mods:topic>
                                <xsl:call-template name="chopPunctuation">
                                        <xsl:with-param name="chopString">
                                                <xsl:call-template name="subfieldSelect">
                                                        <xsl:with-param name="codes">abcd</xsl:with-p
                                                </xsl:call-template>
                                        </xsl:with-param>
                                </xsl:call-template>
                        </mods:topic>
```

```
                <xsl:call-template name="subjectAnyOrder"/>
        </mods:subject>
</xsl:template>

<xsl:template name="createSubGeoFrom651">
        <mods:subject>
                <xsl:call-template name="xxx880"/>
                <xsl:call-template name="subjectAuthority"/>
                <xsl:for-each select="marc:subfield[@code='a']">
                        <mods:geographic>
                                <xsl:call-template name="chopPunctuation">
                                        <xsl:with-param name="chopString" select="."/>
                                </xsl:call-template>
                        </mods:geographic>
                </xsl:for-each>
                <xsl:call-template name="subjectAnyOrder"/>
        </mods:subject>
</xsl:template>

<xsl:template name="createSubFrom653">

        <xsl:if test="@ind2=' '">
                <mods:subject>
                        <mods:topic>
                                <xsl:value-of select="."/>
                        </mods:topic>
                </mods:subject>
        </xsl:if>
        <xsl:if test="@ind2='0'">
                <mods:subject>
                        <mods:topic>
                                <xsl:value-of select="."/>
                        </mods:topic>
                </mods:subject>
        </xsl:if>

        <xsl:if test="@ind2='1'">
                <mods:subject>
                        <mods:name type="personal">
                        <mods:namePart>
                                <xsl:value-of select="."/>
                        </mods:namePart>
                        </mods:name>
                </mods:subject>
        </xsl:if>
        <xsl:if test="@ind2='2'">
                <mods:subject>
                        <mods:name type="corporate">
                        <mods:namePart>
                                <xsl:value-of select="."/>
                        </mods:namePart>
                        </mods:name>
                </mods:subject>
        </xsl:if>
        <xsl:if test="@ind2='3'">
                <mods:subject>
                        <mods:name type="conference">
                        <mods:namePart>
```

```
                                <xsl:value-of select="."/>
                        </mods:namePart>
                        </mods:name>
                </mods:subject>
        </xsl:if>
        <xsl:if test="@ind2=4">
                <mods:subject>
                        <mods:temporal>
                                <xsl:value-of select="."/>
                        </mods:temporal>
                </mods:subject>
        </xsl:if>
        <xsl:if test="@ind2=5">
                <mods:subject>
                        <mods:geographic>
                                <xsl:value-of select="."/>
                        </mods:geographic>
                </mods:subject>
        </xsl:if>

        <xsl:if test="@ind2=6">
                <mods:subject>
                        <mods:genre>
                                <xsl:value-of select="."/>
                        </mods:genre>
                </mods:subject>
        </xsl:if>
</xsl:template>

<xsl:template name="createSubFrom656">
        <mods:subject>
                <xsl:call-template name="xxx880"/>
                <xsl:if test="marc:subfield[@code=2]">
                        <xsl:attribute name="authority">
                                <xsl:value-of select="marc:subfield[@code=2]"/>
                        </xsl:attribute>
                </xsl:if>
                <mods:occupation>
                        <xsl:call-template name="chopPunctuation">
                                <xsl:with-param name="chopString">
                                        <xsl:value-of select="marc:subfield[@code='a']"/>
                                </xsl:with-param>
                        </xsl:call-template>
                </mods:occupation>
        </mods:subject>
</xsl:template>

<xsl:template name="createSubGeoFrom662752">
        <mods:subject>
                <xsl:call-template name="xxx880"/>
                <mods:hierarchicalGeographic>
                        <xsl:for-each select="marc:subfield[@code='a']">
                                <mods:country>
                                        <xsl:call-template name="chopPunctuation">
                                                <xsl:with-param name="chopString" select="."/
                                        </xsl:call-template>
                                </mods:country>
                        </xsl:for-each>
```

```
                        <xsl:for-each select="marc:subfield[@code='b']">
                                <mods:state>
                                        <xsl:call-template name="chopPunctuation">
                                                <xsl:with-param name="chopString" select="."/
                                        </xsl:call-template>
                                </mods:state>
                        </xsl:for-each>
                        <xsl:for-each select="marc:subfield[@code='c']">
                                <mods:county>
                                        <xsl:call-template name="chopPunctuation">
                                                <xsl:with-param name="chopString" select="."/
                                        </xsl:call-template>
                                </mods:county>
                        </xsl:for-each>
                        <xsl:for-each select="marc:subfield[@code='d']">
                                <mods:city>
                                        <xsl:call-template name="chopPunctuation">
                                                <xsl:with-param name="chopString" select="."/
                                        </xsl:call-template>
                                </mods:city>
                        </xsl:for-each>
                        <xsl:for-each select="marc:subfield[@code='e']">
                                <mods:citySection>
                                        <xsl:call-template name="chopPunctuation">
                                                <xsl:with-param name="chopString" select="."/
                                        </xsl:call-template>
                                </mods:citySection>
                        </xsl:for-each>
                        <xsl:for-each select="marc:subfield[@code='g']">
                                <mods:area>
                                        <xsl:call-template name="chopPunctuation">
                                                <xsl:with-param name="chopString" select="."/
                                        </xsl:call-template>
                                </mods:area>
                        </xsl:for-each>
                        <xsl:for-each select="marc:subfield[@code='h']">
                                <mods:extraterrestrialArea>
                                        <xsl:call-template name="chopPunctuation">
                                                <xsl:with-param name="chopString" select="."/
                                        </xsl:call-template>
                                </mods:extraterrestrialArea>
                        </xsl:for-each>
                </mods:hierarchicalGeographic>
        </mods:subject>
</xsl:template>

<xsl:template name="createSubTemFrom045">
        <xsl:if
                test="//marc:datafield[@tag=045 and @ind1='2'][marc:subfield[@code='b' or @co
                <mods:subject>
                        <xsl:call-template name="xxx880"/>
                        <mods:temporal encoding="iso8601" point="start">
                                <xsl:call-template name="dates045b">
                                        <xsl:with-param name="str" select="marc:subfield[@coo
                                        />
                                </xsl:call-template>
                        </mods:temporal>
                        <mods:temporal encoding="iso8601" point="end">
```

```
                                <xsl:call-template name="dates045b">
                                        <xsl:with-param name="str" select="marc:subfield[@co
                                        />
                                </xsl:call-template>
                        </mods:temporal>
                </mods:subject>
        </xsl:if>
</xsl:template>

<!-- classification 050 060 080 082 084 086 -->

<xsl:template name="createClassificationFrom050">
        <xsl:for-each select="marc:subfield[@code='b']">
                <mods:classification authority="lcc">
                        <xsl:call-template name="xxx880"/>
                        <xsl:if test="../marc:subfield[@code='3']">
                                <xsl:attribute name="displayLabel">
                                        <xsl:value-of select="../marc:subfield[@code='3']"/>
                                </xsl:attribute>
                        </xsl:if>
                        <xsl:value-of select="preceding-sibling::marc:subfield[@code='a'][1]"
                        <xsl:text> </xsl:text>
                        <xsl:value-of select="text()"/>
                </mods:classification>
        </xsl:for-each>
        <xsl:for-each
                select="marc:subfield[@code='a'][not(following-sibling::marc:subfield[@code=
                <mods:classification authority="lcc">
                        <xsl:call-template name="xxx880"/>
                        <xsl:if test="../marc:subfield[@code='3']">
                                <xsl:attribute name="displayLabel">
                                        <xsl:value-of select="../marc:subfield[@code='3']"/>
                                </xsl:attribute>
                        </xsl:if>
                        <xsl:value-of select="text()"/>
                </mods:classification>
        </xsl:for-each>
</xsl:template>
<xsl:template name="createClassificationFrom060">
        <mods:classification authority="nlm">
                <xsl:call-template name="xxx880"/>
                <xsl:call-template name="subfieldSelect">
                        <xsl:with-param name="codes">ab</xsl:with-param>
                </xsl:call-template>
        </mods:classification>
</xsl:template>
<xsl:template name="createClassificationFrom080">
        <mods:classification authority="udc">
                <xsl:call-template name="xxx880"/>
                <xsl:call-template name="subfieldSelect">
                        <xsl:with-param name="codes">abx</xsl:with-param>
                </xsl:call-template>
        </mods:classification>
</xsl:template>
<xsl:template name="createClassificationFrom082">
        <mods:classification authority="ddc">
                <xsl:call-template name="xxx880"/>
                <xsl:if test="marc:subfield[@code='2']">
```

```
                        <xsl:attribute name="edition">
                                <xsl:value-of select="marc:subfield[@code='2']"/>
                        </xsl:attribute>
                </xsl:if>
                <xsl:call-template name="subfieldSelect">
                        <xsl:with-param name="codes">ab</xsl:with-param>
                </xsl:call-template>
        </mods:classification>
</xsl:template>
<xsl:template name="createClassificationFrom084">
        <mods:classification>
                <xsl:attribute name="authority">
                        <xsl:value-of select="marc:subfield[@code='2']"/>
                </xsl:attribute>
                <xsl:call-template name="xxx880"/>
                <xsl:call-template name="subfieldSelect">
                        <xsl:with-param name="codes">ab</xsl:with-param>
                </xsl:call-template>
        </mods:classification>
</xsl:template>
<xsl:template name="createClassificationFrom086">
        <xsl:for-each select="marc:datafield[@tag=086][@ind1=0]">
                <mods:classification authority="sudocs">
                        <xsl:call-template name="xxx880"/>
                        <xsl:value-of select="marc:subfield[@code='a']"/>
                </mods:classification>
        </xsl:for-each>
        <xsl:for-each select="marc:datafield[@tag=086][@ind1=1]">
                <mods:classification authority="candoc">
                        <xsl:call-template name="xxx880"/>
                        <xsl:value-of select="marc:subfield[@code='a']"/>
                </mods:classification>
        </xsl:for-each>
        <xsl:for-each select="marc:datafield[@tag=086][@ind1!=1 and @ind1!=0]">
                <mods:classification>
                        <xsl:call-template name="xxx880"/>
                        <xsl:attribute name="authority">
                                <xsl:value-of select="marc:subfield[@code='2']"/>
                        </xsl:attribute>
                        <xsl:value-of select="marc:subfield[@code='a']"/>
                </mods:classification>
        </xsl:for-each>
</xsl:template>

<!-- identifier 020 024 022 028 010 037 UNDO Nov 23 2010 RG SM-->

<!-- createRelatedItemFrom490 <xsl:for-each select="marc:datafield[@tag=490][@ind1=0]"> -->

<xsl:template name="createRelatedItemFrom490">
        <mods:relatedItem type="series">
                <xsl:call-template name="xxx880"/>
                <mods:titleInfo>
                        <mods:title>
                                <xsl:call-template name="chopPunctuation">
                                        <xsl:with-param name="chopString">
                                                <xsl:call-template name="subfieldSelect">
                                                        <xsl:with-param name="codes">av</xsl
                                                </xsl:call-template>
```

```
                                        </xsl:with-param>
                                </xsl:call-template>
                        </mods:title>
                        <xsl:call-template name="part"/>
                </mods:titleInfo>
        </mods:relatedItem>
</xsl:template>


<!-- location 852 856 -->

<xsl:template name="createLocationFrom852">
        <xsl:if test="marc:subfield[@code='a' or @code='b' or @code='e']">
                <mods:physicalLocation>
                        <xsl:call-template name="subfieldSelect">
                                <xsl:with-param name="codes">abe</xsl:with-param>
                        </xsl:call-template>
                </mods:physicalLocation>
        </xsl:if>
        <xsl:if test="marc:subfield[@code='u']">
                <mods:physicalLocation>
                        <xsl:call-template name="uri"/>
                        <xsl:call-template name="subfieldSelect">
                                <xsl:with-param name="codes">u</xsl:with-param>
                        </xsl:call-template>
                </mods:physicalLocation>
        </xsl:if>
        <xsl:if
                test="marc:subfield[@code='h' or @code='i' or @code='j' or @code='k'
                <mods:shelfLocator>
                        <xsl:call-template name="subfieldSelect">
                                <xsl:with-param name="codes">hijklmt</xsl:with-param>
                        </xsl:call-template>
                </mods:shelfLocator>
        </xsl:if>
</xsl:template>

<xsl:template name="createLocationFrom910">
 <xsl:if test="marc:subfield[@code='a']">
    <mods:physicalLocation authority="siglaADR">
      <xsl:call-template name="subfieldSelect">
         <xsl:with-param name="codes">a</xsl:with-param>
      </xsl:call-template>
  </mods:physicalLocation>
 </xsl:if>
      <xsl:if test="marc:subfield[@code='b']">
      <xsl:call-template name="createShelfLocatorsFor910">
         <xsl:with-param name="list"><xsl:value-of select="marc:subfield[@code='b']" /></xsl
         <xsl:with-param name="delimiter">,</xsl:with-param>
      </xsl:call-template>
    </xsl:if>
</xsl:template>

<xsl:template name="createLocationFrom856">
        <xsl:if test="//marc:datafield[@tag=856][@ind2!=2][marc:subfield[@code='u']]">
                <mods:url displayLabel="electronic resource">
                        <!-- 1.41 tmee AQ1.9 added choice protocol for @usage="prima
                        <xsl:variable name="primary">
```

```
                                        <xsl:choose>
                                                <xsl:when
                                                        test="@ind2=0 and count(preceding-sib
                                                        >true</xsl:when>

                                                <xsl:when
                                                        test="@ind2=1 and
                                                count(ancestor::marc:record//marc:datafield[@
                                                count(preceding-sibling::marc:datafield[@tag=
                                                        >true</xsl:when>

                                                <xsl:when
                                                        test="@ind2!=1 and @ind2!=0 and
                                                @ind2!=2 and count(ancestor::marc:record//mar
                                                @ind2=0])=0 and count(ancestor::marc:record/
                                                @ind2=1])=0 and
                                                count(preceding-sibling::marc:datafield[@tag=
                                                        >true</xsl:when>
                                                <xsl:otherwise>false</xsl:otherwise>
                                        </xsl:choose>
                                </xsl:variable>
                                <xsl:if test="$primary='true'">
                                        <xsl:attribute name="usage">primary display</xsl:attr
                                </xsl:if>

                                <xsl:if test="marc:subfield[@code='y' or @code='3']">
                                        <xsl:attribute name="displayLabel">
                                                <xsl:call-template name="subfieldSelect">
                                                        <xsl:with-param name="codes">y3</xsl
                                                </xsl:call-template>
                                        </xsl:attribute>
                                </xsl:if>
                                <xsl:if test="marc:subfield[@code='z']">
                                        <xsl:attribute name="note">
                                                <xsl:call-template name="subfieldSelect">
                                                        <xsl:with-param name="codes">z</xsl:w
                                                </xsl:call-template>
                                        </xsl:attribute>
                                </xsl:if>
                                <xsl:value-of select="marc:subfield[@code='u']"/>
                        </mods:url>
        </xsl:if>
</xsl:template>

<!-- accessCondition 506 540 -->

<xsl:template name="createAccessConditionFrom506">
        <mods:accessCondition type="restrictionOnAccess">
                <xsl:call-template name="xxx880"/>
                <xsl:call-template name="subfieldSelect">
                        <xsl:with-param name="codes">abcd35</xsl:with-param>
                </xsl:call-template>
        </mods:accessCondition>
</xsl:template>

<xsl:template name="createAccessConditionFrom540">
        <mods:accessCondition type="useAndReproduction">
                <xsl:call-template name="xxx880"/>
```

```
                <xsl:call-template name="subfieldSelect">
                        <xsl:with-param name="codes">abcde35</xsl:with-param>
                </xsl:call-template>
        </mods:accessCondition>
</xsl:template>

<!-- recordInfo 040 005 001 003 -->

<!-- 880 global copy template -->
<xsl:template match="* | @*" mode="global_copy">
        <xsl:copy>
                <xsl:apply-templates select="* | @* | text()" mode="global_copy"/>
        </xsl:copy>
</xsl:template>


  <xsl:variable name="ascii">
        <xsl:text> !"#$%&amp;'()*+,-./0123456789:;&lt;=&gt;?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`
</xsl:variable>

<xsl:variable name="latin1">
        <xsl:text> ¡¢£¤¥¦§¨©ª«¬®¯°±²³´µ¶·¸¹º»¼½¾¿ÀÁÂÃÄÅÆÇÈÉÊËÌÍÎÏÐÑÒÓÔÕÖ×ØÙÚÛÜÝÞßàáâãäåæçèéê
</xsl:variable>
<!-- Characters that usually don't need to be escaped -->
<xsl:variable name="safe">
        <xsl:text>!'()*-.0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ_abcdefghijklmnopqrstuvwxyz~</xs
</xsl:variable>

<xsl:variable name="hex">0123456789ABCDEF</xsl:variable>


<xsl:template name="datafield">
        <xsl:param name="tag"/>
        <xsl:param name="ind1">
                <xsl:text> </xsl:text>
        </xsl:param>
        <xsl:param name="ind2">
                <xsl:text> </xsl:text>
        </xsl:param>
        <xsl:param name="subfields"/>
        <xsl:element name="marc:datafield">
                <xsl:attribute name="tag">
                        <xsl:value-of select="$tag"/>
                </xsl:attribute>
                <xsl:attribute name="ind1">
                        <xsl:value-of select="$ind1"/>
                </xsl:attribute>
                <xsl:attribute name="ind2">
                        <xsl:value-of select="$ind2"/>
                </xsl:attribute>
                <xsl:copy-of select="$subfields"/>
        </xsl:element>
</xsl:template>

<xsl:template name="subfieldSelect">
        <xsl:param name="codes">abcdefghijklmnopqrstuvwxyz</xsl:param>
        <xsl:param name="delimeter">
                <xsl:text> </xsl:text>
```

```
                </xsl:param>
                <xsl:variable name="str">
                        <xsl:for-each select="marc:subfield">
                                <xsl:if test="contains($codes, @code)">
                                        <xsl:value-of select="text()"/>
                                        <xsl:value-of select="$delimeter"/>
                                </xsl:if>
                        </xsl:for-each>
                </xsl:variable>
                <xsl:value-of select="substring($str,1,string-length($str)-string-length($delimeter))
        </xsl:template>

        <xsl:template name="createShelfLocatorsFor910">
                <xsl:param name="list" />
                <xsl:param name="delimiter" />
                <xsl:variable name="newlist">
                        <xsl:choose>
                                <xsl:when test="contains($list, $delimiter)"><xsl:value-of select="no
                                <xsl:otherwise><xsl:value-of select="concat(normalize-space($list), $
                        </xsl:choose>
    </xsl:variable>
      <xsl:variable name="first" select="substring-before($newlist, $delimiter)" />
      <xsl:variable name="remaining" select="substring-after($newlist, $delimiter)" />
      <mods:shelfLocator>
      <xsl:value-of select="$first" />
      </mods:shelfLocator>
      <xsl:if test="$remaining">
          <xsl:call-template name="createShelfLocatorsFor910">
              <xsl:with-param name="list" select="$remaining" />
                       <xsl:with-param name="delimiter"><xsl:value-of select="$delimiter"/></xsl:wi
          </xsl:call-template>
      </xsl:if>
  </xsl:template>

       <xsl:template name="buildSpaces">
                <xsl:param name="spaces"/>
                <xsl:param name="char">
                        <xsl:text> </xsl:text>
                </xsl:param>
                <xsl:if test="$spaces&gt;0">
                        <xsl:value-of select="$char"/>
                        <xsl:call-template name="buildSpaces">
                                <xsl:with-param name="spaces" select="$spaces - 1"/>
                                <xsl:with-param name="char" select="$char"/>
                        </xsl:call-template>
                </xsl:if>
        </xsl:template>

        <xsl:template name="chopPunctuation">
                <xsl:param name="chopString"/>
                <xsl:param name="punctuation">
                        <xsl:text>.:,;/ </xsl:text>
                </xsl:param>
                <xsl:variable name="length" select="string-length($chopString)"/>
                <xsl:choose>
                        <xsl:when test="$length=0"/>
                        <xsl:when test="contains($punctuation, substring($chopString,$length,1))">
                                <xsl:call-template name="chopPunctuation">
```

```
                                        <xsl:with-param name="chopString" select="substring($chopStri
                                        <xsl:with-param name="punctuation" select="$punctuation"/>
                                </xsl:call-template>
                        </xsl:when>
                        <xsl:when test="not($chopString)"/>
                        <xsl:otherwise>
                                <xsl:value-of select="$chopString"/>
                        </xsl:otherwise>
                </xsl:choose>
        </xsl:template>

        <xsl:template name="chopPunctuationFront">
                <xsl:param name="chopString"/>
                <xsl:variable name="length" select="string-length($chopString)"/>
                <xsl:choose>
                        <xsl:when test="$length=0"/>
                        <xsl:when test="contains('.:,;/[ ', substring($chopString,1,1))">
                                <xsl:call-template name="chopPunctuationFront">
                                        <xsl:with-param name="chopString" select="substring($chopStri
                                </xsl:call-template>
                        </xsl:when>
                        <xsl:when test="not($chopString)"/>
                        <xsl:otherwise>
                                <xsl:value-of select="$chopString"/>
                        </xsl:otherwise>
                </xsl:choose>
        </xsl:template>

        <xsl:template name="chopPunctuationBack">
                <xsl:param name="chopString"/>
                <xsl:param name="punctuation">
                        <xsl:text>.:,;/] </xsl:text>
                </xsl:param>
                <xsl:variable name="length" select="string-length($chopString)"/>
                <xsl:choose>
                        <xsl:when test="$length=0"/>
                        <xsl:when test="contains($punctuation, substring($chopString,$length,1))">
                                <xsl:call-template name="chopPunctuation">
                                        <xsl:with-param name="chopString" select="substring($chopStri
                                        <xsl:with-param name="punctuation" select="$punctuation"/>
                                </xsl:call-template>
                        </xsl:when>
                        <xsl:when test="not($chopString)"/>
                        <xsl:otherwise>
                                <xsl:value-of select="$chopString"/>
                        </xsl:otherwise>
                </xsl:choose>
        </xsl:template>

        <!-- nate added 12/14/2007 for lccn.loc.gov: url encode ampersand, etc. -->
        <xsl:template name="url-encode">

                <xsl:param name="str"/>

                <xsl:if test="$str">
                        <xsl:variable name="first-char" select="substring($str,1,1)"/>
                        <xsl:choose>
                                <xsl:when test="contains($safe,$first-char)">
```

```
                                        <xsl:value-of select="$first-char"/>
                                </xsl:when>
                                <xsl:otherwise>
                                        <xsl:variable name="codepoint">
                                                <xsl:choose>
                                                        <xsl:when test="contains($ascii,$first-char)"
                                                                <xsl:value-of select="string-length(s
                                                        </xsl:when>
                                                        <xsl:when test="contains($latin1,$first-char)
                                                                <xsl:value-of select="string-length(s
                                                                <!-- was 160 -->
                                                        </xsl:when>
                                                        <xsl:otherwise>
                                                                <xsl:message terminate="no">Warning:
                                                                        that is out of range! Substit
                                                                <xsl:text>63</xsl:text>
                                                        </xsl:otherwise>
                                                </xsl:choose>
                                        </xsl:variable>
                                        <xsl:variable name="hex-digit1" select="substring($hex,floor
                                        <xsl:variable name="hex-digit2" select="substring($hex,$codep
                                        <!-- <xsl:value-of select="concat('%',$hex-digit2)"/> -->
                                        <xsl:value-of select="concat('%',$hex-digit1,$hex-digit2)"/>
                                </xsl:otherwise>
                        </xsl:choose>
                        <xsl:if test="string-length($str) &gt; 1">
                                <xsl:call-template name="url-encode">
                                        <xsl:with-param name="str" select="substring($str,2)"/>
                                </xsl:call-template>
                        </xsl:if>
                </xsl:if>
        </xsl:template>

</xsl:stylesheet>
```

1.1.9 MARC21toPeriodicalTitle.xsl

This file is used to transform **periodical** publications.

```
<xsl:stylesheet xmlns:mods="http://www.loc.gov/mods/v3"
        xmlns:marc="http://www.loc.gov/MARC21/slim" xmlns:xlink="http://www.w3.org/1999/xlink"
        xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
        exclude-result-prefixes="xlink marc" version="1.0">
        <xsl:output encoding="UTF-8" indent="yes" method="xml" />
        <xsl:strip-space elements="*" />
        <xsl:namespace-alias stylesheet-prefix="mods"
                result-prefix="mods" />

        <xsl:template match="/">

                <mods:mods ID="MODS_VOLUME_0001">
                        <mods:genre>volume</mods:genre>
                </mods:mods>

        </xsl:template>

</xsl:stylesheet>
```

CHAPTER 2

Installation

Module is hosted at PYPI, and can be installed using PIP:

```
sudo pip install marcxml2mods
```

2.1 Source code

Project is released as opensource (GPL) and source codes can be found at GitHub:

- https://github.com/edeposit/marcxml2mods

2.2 Unittests

Almost every feature of the project is tested by unittests. You can run those tests using provided `run_tests.sh` script, which can be found in the root of the project.

2.2.1 Requirements

This script expects that pytest is installed. In case you don't have it yet, it can be easily installed using following command:

```
pip install --user pytest
```

or for all users:

```
sudo pip install pytest
```

2.2.2 Example

```
$ ./run_tests.sh
============================= test session starts ==============================
platform linux2 -- Python 2.7.6 -- py-1.4.26 -- pytest-2.6.4
plugins: cov
collected 29 items

tests/test_transformators.py ..
tests/test_xslt_transformer.py .........
```

```
tests/mods_postprocessor/test_mods_postprocessor_init.py .
tests/mods_postprocessor/test_monograph.py .......
tests/mods_postprocessor/test_multi_monograph.py F
tests/mods_postprocessor/test_periodical.py F
tests/mods_postprocessor/test_shared_funcs.py ........

================================== FAILURES ===================================
_____________________________ test_postprocess_multi_mono ______________________________

    def test_postprocess_multi_mono():
>       raise NotImplementedError()
E       NotImplementedError

tests/mods_postprocessor/test_multi_monograph.py:27: NotImplementedError
_____________________________ test_postprocess_periodical ______________________________

    def test_postprocess_periodical():
>       raise NotImplementedError()
E       NotImplementedError

tests/mods_postprocessor/test_periodical.py:27: NotImplementedError
==================== 2 failed, 27 passed in 0.77 seconds ======================
```

CHAPTER 3

Indices and tables

- genindex
- modindex
- search

m

Symbols

A

D

F

G

I

M

P

R

T

X