16.3. XML stylesheet conversion crosswalks

16.3.1. *FGDCI to DIF stylesheet converter*

[A copy of this stylesheet (fgdcxml-difxml_NOAA_NODC.xsl) can be found at the PICES web site.]

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE DIF SYSTEM
"http://gcmd.gsfc.nasa.gov/Aboutus/xml/dif/dif_v9.4.dtd">

<!-- since an XSL stylesheet is an XML document itself, it always begins with
the XML declaration -->
<!-- <?xml version="1.0" encoding="ISO-8859-1"?> -->
<!-- DTD declared with a Document Type Declaration.
To create an external reference to a DTD, use the SYSTEM command -->

<!-- start a XSL stylesheet -->

<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
   xmlns:java="http://xml.apache.org/xalan/java" exclude-result-
   prefixes="java"
   version="1.0">

<!-- xmlns:java="http://xml.apache.org/xslt/java" exclude-result-
prefixes="java" -->
<!-- declare that this document is an XSLT style sheet and points to the
official W3C XSLT namespace; points to Java to support different browsers and
different user needs (Establishes the java namespace for use with extensions.
Xalan is written in Java).  JavaScript that uses an XML parser to do the
transformation, making XML data available to all skind of browsers -->

<xsl:output method="xml" />

<!--The root element that declares the document to be an XSL style  sheet is
<xsl:stylesheet> or <xsl:transform>, either can be used.  This is following
W3C XSLT recommendation.  To get access to the XSLT  elements, attributes,
and features, XSLT namespace must be declared at the top of the document
(xmlns:xsl=" ").

DIF xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://gcmd.gsfc.nasa.gov/Aboutus/xml/dif/dif.
xsd">

Additional information:
http://www.w3schools.com/xsl/xsl_transformation.asp
http://www.fgdc.gov/metadata/csdgm/
http://biology.usgs.gov/fgdc.metadata/version2/

Command line:

java org.apache.xalan.xslt.Process -IN FGDC_*.xml -XSL fgdcxml-
difxml_NOAA_NODC.xsl -OUT DIF_*.xml
/home/mmeaux/bin/tidy -config /home/mmeaux/bin/config.txt DIF_*.xml >
DIF_*_clean.xml
-->
```

```
<!-- This is an XSLT style sheet that converts metadata in FGDC XML format to
IDN DIF XML format.-->
<!-- Warning: This XSLT is a work in progress -->

<!-- Author: Scott Ritz, ritz@gcmd.nasa.gov -->
<!-- Modified by Melanie Meaux in 2005s -->
<!-- Contact: mmeaux@gcmd.nasa.gov -->

<!-- apply templates; the <xsl:template> element is used to build templates,
one or more set of rules. The match attribute is used to associate a template
with an XML element. The match attribute can also be used to define a
template for the entire XML document. The value of the match attribute is an
XPath expression (i.e. match="/" defines the whole document). The content
inside the element defines some HTML to write to the output.

The <xsl:apply-templates> element applies a template to the current element
or to the current element's child nodes.If we add a select attribute to the
<xsl:apply-templates> element it will process only the child element that
matches the value of the attribute. We can use the "select" attribute to
specify the order in which the child nodes are processed.
-->

<!--Main template for the DIF document; -->
<!-- for repeated fields (i.e personnel) need to include DIF field attributes
in xsl:template match = ""> and not in template below -->

<xsl:template match="metadata">
   <DIF>
   <!--
   <Entry_ID>Place_Holder</Entry_ID>-->
   <!-- Inserts the Resource description short name into the Entry_ID -->
   <Entry_ID><xsl:apply-templates select="/metadata/distinfo/resdesc"
   /></Entry_ID>
   <Entry_Title><xsl:apply-templates
   select="/metadata/idinfo/citation/citeinfo/title" /></Entry_Title>

   <xsl:apply-templates select="idinfo" /> <!-- Identification Information --
   >
   <xsl:apply-templates select="metainfo" /> <!-- Metadata Reference
   Information -->
   <xsl:apply-templates select="distinfo" /> <!-- Distribution Informaiton --
   >
   <xsl:apply-templates select="dataqual" /> <!-- Data Quality Information --
   >
   <IDN_Node>
   <Short_Name>USA/NOAA</Short_Name>
   </IDN_Node>
   <Data_Set_Language>ENGLISH</Data_Set_Language>
   <Originating_Metadata_Node>GCMD</Originating_Metadata_Node>
   <Metadata_Name>CEOS IDN DIF</Metadata_Name> <!-- Metadata Name -->
   <Metadata_Version>VERSION 9.0</Metadata_Version> <!-- Metadata Version -->
   </DIF>
</xsl:template>

<!-- calling above templates -->
<!--                              1. IDENTIFICATION INFORMATION
-->
```

```
<xsl:template match="idinfo">
   <xsl:apply-templates select="citation" /> <!-- citation -->
   <Summary>
   <xsl:apply-templates select="descript" /> <!-- Description -->
   </Summary>
   <xsl:apply-templates select="timeperd" /> <!--Time Period of Content -->
   <!-- place-holder for Paleo Temporal field -->
   <xsl:apply-templates select="status" /> <!-- Status-->
   <xsl:apply-templates select="spdom" />  <!-- Spatial Domain -->
   <xsl:apply-templates select="keywords" />
   <xsl:apply-templates select="accconst" /> <!-- access constraints -->
   <Use_Constraints> <!-- use constraints -->
   <xsl:apply-templates select="useconst" />
   <xsl:apply-templates select="/metadata/distinfo/distliab" />
   </Use_Constraints>
   <xsl:apply-templates select="ptcontac" /> <!-- point of contact -->
   <xsl:apply-templates select="browse" />
   <Reference>
     <xsl:apply-templates select="/metadata/idinfo/crossref/citeinfo/origin"
     />
     <xsl:apply-templates select="/metadata/idinfo/crossref/citeinfo/title" />
     <xsl:apply-templates
     select="/metadata/idinfo/crossref/citeinfo/serinfo/sername" />
     <xsl:apply-templates select="/metadata/idinfo/crossref/citeinfo/pubdate"
     />
     <xsl:apply-templates
     select="/metadata/idinfo/crossref/citeinfo/pubinfo/pubplace" />
     <xsl:apply-templates
     select="/metadata/idinfo/crossref/citeinfo/pubinfo/publish" />
     <xsl:apply-templates select="/metadata/idinfo/crossref/citeinfo/edition"
     />
     <xsl:apply-templates
     select="/metadata/idinfo/crossref/citeinfo/serinfo/issue" />
     <xsl:apply-templates select="/metadata/idinfo/crossref/citeinfo/geoform"
     />
     <xsl:apply-templates select="/metadata/idinfo/crossref/citeinfo/othercit"
     />
     <xsl:apply-templates select="/metadata/idinfo/crossref/citeinfo/onlink"
     />
   </Reference>
</xsl:template>

<!-- begin citation -->
<xsl:template match="citation">
   <Data_Set_Citation><xsl:apply-templates select="citeinfo"
   /></Data_Set_Citation>
</xsl:template>

<xsl:template match="citeinfo"> <!-- Citation Information -->
   <Dataset_Creator><xsl:apply-templates select="origin" /></Dataset_Creator>
   <!-- Originator: the name of an organization or individual that developed
   the data set. If the name of editors or compilers are provided, the name
   must be followed by "(ed.)" or "(comp.)" respectively.  This list can be
   quite long and exceed the character length specificed by the DIF in
   Dataset_Creator. In some cases, the Pulisher can replace the Originator in
```

```
   DIF Dataset_Creator. Publisher: the name of the individual or organization
   that published the data set.-->
   <Dataset_Title><xsl:apply-templates select="title" /></Dataset_Title>
   <xsl:apply-templates select="serinfo" />
   <Dataset_Release_Date><xsl:apply-templates select="pubdate" />
   </Dataset_Release_Date>

   <xsl:apply-templates select="pubinfo" />

   <Version><xsl:apply-templates select="edition" /></Version>

   <Data_Presentation_Form><xsl:apply-templates select="geoform"
   /></Data_Presentation_Form>

<Other_Citation_Details><xsl:apply-templates select="othercit"
/></Other_Citation_Details>
   <Online_Resource><xsl:apply-templates select="onlink" /></Online_Resource>
</xsl:template>

<xsl:template match="serinfo"> <!-- Series Information -->
   <Dataset_Series_Name><xsl:apply-templates select="sername"
   /></Dataset_Series_Name>
   <Issue_Identification><xsl:apply-templates select="issue"
   /></Issue_Identification>
</xsl:template>

<xsl:template match="pubinfo"> <!-- Publication Information -->
   <Dataset_Release_Place><xsl:apply-templates
   select="pubplace"/></Dataset_Release_Place>
   <Dataset_Publisher><xsl:apply-templates select="publish"
   /></Dataset_Publisher>
</xsl:template>

<!-- DIF Data Set Citation -->
<xsl:template match="origin"><xsl:value-of select="normalize-space(.)"
/></xsl:template>
<xsl:template match="title"><xsl:value-of select="normalize-space(.)"
/></xsl:template>
<xsl:template match="sername"><xsl:value-of select="." /></xsl:template>
<xsl:template match="pubdate">
<xsl:variable name="datetime" select="normalize-space(.)"/>
<xsl:variable name="year" select="substring($datetime, 1, 4)"/>
<xsl:variable name="month" select="substring($datetime, 5, 2)"/>
<xsl:variable name="day" select="substring($datetime, 7, 2)"/>
<xsl:value-of select="concat($year, '-', $month, '-', $day)" />
</xsl:template>
<xsl:template match="pubplace"><xsl:value-of select="." /></xsl:template>
<xsl:template match="publish"><xsl:value-of select="." /></xsl:template>
<xsl:template match="edition"><xsl:value-of select="." /></xsl:template>
<xsl:template match="issue"><xsl:value-of select="." /></xsl:template>
<xsl:template match="geoform"><xsl:value-of select="." /></xsl:template>
<xsl:template match="othercit"><xsl:value-of select="." /></xsl:template>
<xsl:template match="onlink"><xsl:value-of select="." /></xsl:template>
<!-- end citation -->

<!-- begin summary -->
```

```
       <!-- DIF Summary -->
<xsl:template match="descript">
   <xsl:apply-templates select="abstract" />
   <xsl:apply-templates select="purpose" />
   <xsl:apply-templates select="supplinf" />
   <xsl:apply-templates select="/metadata/idinfo/native" />
<!--  native: a description of the data set in the producer's processing
environment, including items such as the name of the software (including
version), the computer operating system, file name (including host-, path-,
and filenames), and the data set size.-->
   <!-- <xsl:apply-templates select="/metadata/spref" /> spatial information
   not in DIF; different from Spatial Domain in Identification Information
   section-->
</xsl:template>
<xsl:template match="abstract"><xsl:text> ABSTRACT: </xsl:text><xsl:value-of
select="normalize-space(.)" /></xsl:template>
<xsl:template match="purpose"><xsl:text> PURPOSE: </xsl:text><xsl:value-of
select="normalize-space(.)" /></xsl:template>
<xsl:template match="supplinf"><xsl:text> SUPPLEMENTAL INFORMATION:
</xsl:text><xsl:value-of select="normalize-space(.)" /></xsl:template>
<xsl:template match="native"><xsl:text> NATIVE: </xsl:text><xsl:value-of
select="normalize-space(.)" /></xsl:template>
<!-- end summary -->

<!-- begin Temporal Coverage -->
           <!-- DIF Temportal_Coverage -->
<xsl:template match="timeperd"><xsl:apply-templates select="timeinfo"
/></xsl:template>
           <xsl:template match="timeinfo">
<Temporal_Coverage><xsl:apply-templates select="rngdates"
/></Temporal_Coverage>
</xsl:template>

<xsl:template match="rngdates">
     <Start_Date><xsl:apply-templates select="begdate" /></Start_Date>
     <Stop_Date><xsl:apply-templates select="enddate" /></Stop_Date>
</xsl:template>

<xsl:template match="begdate">
<xsl:variable name="datetime" select="normalize-space(.)"/>
<xsl:variable name="year" select="substring($datetime, 1, 4)"/>
<xsl:variable name="month" select="substring($datetime, 5, 2)"/>
<xsl:variable name="day" select="substring($datetime, 7, 2)"/>
<xsl:value-of select="concat($year, '-', $month, '-', $day)" />
</xsl:template>


<xsl:template match="enddate">
<xsl:variable name="datetime" select="normalize-space(.)"/>
<xsl:variable name="year" select="substring($datetime, 1, 4)"/>
<xsl:variable name="month" select="substring($datetime, 5, 2)"/>
<xsl:variable name="day" select="substring($datetime, 7, 2)"/>
<xsl:value-of select="concat($year, '-', $month, '-', $day)" />
</xsl:template>
<!-- end temporal coverage -->

<!-- begin status -->
```

```
          <!-- DIF Data_Set_Progress -->
<xsl:template match="status">
   <Data_Set_Progress><xsl:apply-templates select="progress"
   /></Data_Set_Progress>
   <!-- <xsl:apply-templates select="update" /> -->
</xsl:template>

<xsl:template match="progress">
<xsl:value-of
select="translate(.,'abcdefghijklmnopqrstuvwxyz','ABCDEFGHIJKLMNOPQRSTUVWXYZ'
)" />
</xsl:template>

<!-- <xsl:template match="update"><xsl:value-of select="." /></xsl:template>
-->
<!-- end status -->

<!-- begin spatial coverage -->
          <!-- DIF Spatial_Coverage -->
<xsl:template match="spdom">
   <Spatial_Coverage><xsl:apply-templates select="bounding"
   /></Spatial_Coverage>
</xsl:template>

<xsl:template match="bounding">
   <Southernmost_Latitude><xsl:apply-templates
   select="southbc"/></Southernmost_Latitude>
   <Northernmost_Latitude><xsl:apply-templates select="northbc"
   /></Northernmost_Latitude>
   <Westernmost_Longitude><xsl:apply-templates select="westbc"
   /></Westernmost_Longitude>
   <Easternmost_Longitude><xsl:apply-templates select="eastbc"
   /></Easternmost_Longitude>
</xsl:template>

<xsl:template match="westbc"><xsl:value-of select="." /></xsl:template>
<xsl:template match="eastbc"><xsl:value-of select="." /></xsl:template>
<xsl:template match="northbc"><xsl:value-of select="." /></xsl:template>
<xsl:template match="southbc"><xsl:value-of select="." /></xsl:template>
<!-- end Spatial Coverage -->

<!-- begin keywords/additional mapping needed to Parameter and keyword DIF
fields -->
<xsl:template match="keywords">
   <xsl:apply-templates select="theme" />
   <xsl:apply-templates select="place" />
</xsl:template>
<xsl:template match="theme">
   <xsl:apply-templates select="themekt" /> <!-- Theme_Keyword_Thesaurus -->
   <xsl:apply-templates select="themekey" /> <!-- Theme Keyword -->
</xsl:template>

<xsl:template match="place">
   <xsl:apply-templates select="placekt" />
   <xsl:apply-templates select="placekey" />
</xsl:template>
```

```
<!-- DIF Keyword and Science Parameter Field -->
<xsl:template match="themekt">
  <xsl:choose>
    <xsl:when test="'Global Change Master Directory'"> <!-- specific to NOAA
    NODC -->
      <Keyword/>
    </xsl:when>
    <xsl:otherwise>
      <Keyword><xsl:value-of select="." /></Keyword>
    </xsl:otherwise>
  </xsl:choose>
</xsl:template>

<xsl:template match="themekey">
  <xsl:choose>
    <xsl:when test="'Global Change Master Directory'">
      <Parameters>
      <Category>
      <xsl:value-of select="." />
      </Category>
      </Parameters>
    </xsl:when>
    <xsl:otherwise>
      <Keyword><xsl:value-of select="." /></Keyword>
    </xsl:otherwise>
  </xsl:choose>
</xsl:template>

<xsl:template match="placekt">
  <xsl:choose>
    <xsl:when test="'Global Change Master Directory'"> <!-- specific to NOAA
    NODC -->
      <Keyword/>
    </xsl:when>
    <xsl:otherwise>
      <Keyword><xsl:value-of select="." /></Keyword>
    </xsl:otherwise>
    </xsl:choose>
</xsl:template>

<xsl:template match="placekey">
  <xsl:choose>
    <xsl:when test="'Global Change Master Directory'">
      <Location>
      <Location_Name>
        <xsl:value-of select="." />
      </Location_Name>
      </Location>
    </xsl:when>
    <xsl:otherwise>
      <Keyword><xsl:value-of select="." /></Keyword>
    </xsl:otherwise>
  </xsl:choose>
</xsl:template>
<!-- end keywords -->

<!-- begin access constraints -->
```

```
          <!-- DIF Access_Constraints -->
<xsl:template match="accconst"><Access_Constraints><xsl:value-of select="."
/></Access_Constraints></xsl:template>
<!-- end access constraints -->

<!-- begin use constraints -->
          <!-- DIF Use_Constraints -->
<xsl:template match="useconst"><xsl:value-of select="." /></xsl:template>
<xsl:template match="distliab"><xsl:value-of select="." /></xsl:template>
<!-- Distribution Liability-statement of the liability assumed by the
distributor -->
<!-- end use constraints -->

<!-- begin point of contact -->
          <!-- DIF Technical Contact -->
<xsl:template match="ptcontac">
 <Personnel>
 <Role>TECHNICAL CONTACT</Role>
 <xsl:apply-templates select="/metadata/idinfo/ptcontac/cntinfo" />
 </Personnel>
</xsl:template>

<xsl:template match="/metadata/idinfo/ptcontac/cntinfo">
   <xsl:apply-templates select="cntperp" /> <!-- contact person primary -->
   <xsl:apply-templates select="cntorgp" /> <!-- contact Organization primary
   -->
   <Contact_Address><xsl:apply-templates select="cntaddr"
   /></Contact_Address>
   <xsl:apply-templates select="cntemail" />
   <xsl:apply-templates select="cntvoice" />
   <xsl:apply-templates select="cntfax" />
</xsl:template>
<xsl:template match="/metadata/idinfo/ptcontac/cntinfo/cntperp">
<xsl:apply-templates
select="/metadata/idinfo/ptcontac/cntinfo/cntperp/cntper" /> <!-- contact
person -->
<xsl:apply-templates
select="/metadata/idinfo/ptcontac/cntinfo/cntperp/cntorg" /> <!-- contact
organization -->
</xsl:template>

<xsl:template
match="/metadata/idinfo/ptcontac/cntinfo/cntperp/cntper"><First_Name><xsl:val
ue-of select="." /></First_Name></xsl:template>
<xsl:template
match="/metadata/idinfo/ptcontac/cntinfo/cntperp/cntorg"><Last_Name><xsl:valu
e-of select="." /></Last_Name></xsl:template>

<xsl:template match="cntaddr">
   <xsl:apply-templates select="address" />
   <xsl:apply-templates select="city" />
   <xsl:apply-templates select="state" />
   <xsl:apply-templates select="postal" />
   <xsl:apply-templates select="country" />
</xsl:template>
```

```
<xsl:template match="address"><Address><xsl:value-of select="."
/></Address></xsl:template>
<xsl:template match="city"><City><xsl:value-of select="."
/></City></xsl:template>
<xsl:template match="state"><Province_or_State><xsl:value-of select="."
/></Province_or_State></xsl:template>
<xsl:template match="postal"><Postal_Code><xsl:value-of select="."
/></Postal_Code></xsl:template>
<xsl:template match="country"><Country><xsl:value-of select="."
/></Country></xsl:template>

<xsl:template match="cntemail"><Email><xsl:value-of select="."
/></Email></xsl:template>
<xsl:template match="cntvoice"><Phone><xsl:value-of select="."
/></Phone></xsl:template>
<xsl:template match="cntfax"><Fax><xsl:value-of select="."
/></Fax></xsl:template>
<!-- end point of contact -->

<!-- begin browse -->
          <!-- DIF Multimedia_Sample -->
<xsl:template match="browse">
  <Multimedia_Sample>
    <File>
      <xsl:apply-templates select="browsen" /> <!-- Browse Graphic File Name --
      >
    </File>
  <Format>
    <xsl:apply-templates select="browset" /> <!-- Browse Graphic File Type -->
  </Format>
  <Description>
    <xsl:apply-templates select="browsed" /> <!-- Browse Graphic File
    Description -->
  </Description>
  </Multimedia_Sample>
</xsl:template>

<xsl:template match="browsen"><xsl:value-of select="." /></xsl:template>
<xsl:template match="browset"><xsl:value-of select="." /></xsl:template>
<xsl:template match="browsed"><xsl:value-of select="." /></xsl:template>
<!-- end browse -->

<!-- begin cross reference -->
<!-- Cross Reference: information about other, related data sets that are
likely to be of interest.-->
<xsl:template match="/metadata/idinfo/crossref/citeinfo/origin"><xsl:value-of
select="normalize-space(.)" /></xsl:template>
<xsl:template match="/metadata/idinfo/crossref/citeinfo/title"><xsl:value-of
select="normalize-space(.)" /></xsl:template>
<xsl:template
match="/metadata/idinfo/crossref/citeinfo/serinfo/sername"><xsl:value-of
select="." /></xsl:template>
<xsl:template match="/metadata/idinfo/crossref/citeinfo/pubdate">
<xsl:variable name="datetime" select="normalize-space(.)"/>
<xsl:variable name="year" select="substring($datetime, 1, 4)"/>
<xsl:variable name="month" select="substring($datetime, 5, 2)"/>
<xsl:variable name="day" select="substring($datetime, 7, 2)"/>
```

```
<xsl:value-of select="concat($year, '-', $month, '-', $day)" />
</xsl:template>
<xsl:template
match="/metadata/idinfo/crossref/citeinfo/pubinfo/pubplace"><xsl:value-of
select="." /></xsl:template>
<xsl:template
match="/metadata/idinfo/crossref/citeinfo/pubinfo/publish"><xsl:value-of
select="." /></xsl:template>
<xsl:template match="/metadata/idinfo/crossref/citeinfo/edition"><xsl:value-
of select="." /></xsl:template>
<xsl:template match="/metadata/idinfo/crossref/citeinfo/issue"><xsl:value-of
select="." /></xsl:template>
<xsl:template match="/metadata/idinfo/crossref/citeinfo/geoform"><xsl:value-
of select="." /></xsl:template>
<xsl:template match="/metadata/idinfo/crossref/citeinfo/othercit"><xsl:value-
of select="." /></xsl:template>
<xsl:template match="/metadata/idinfo/crossref/citeinfo/onlink"><xsl:value-of
select="." /></xsl:template>
<!-- end cross reference -->

<!--                          2. METADATA REFERENCE INFORMATION              -
->
<!-- DIF Author -->
<xsl:template match="metainfo">
   <Personnel>
   <Role>DIF AUTHOR</Role>
   <xsl:apply-templates select="/metadata/metainfo/metc/cntinfo" /> <!--
   Metadata Contact -->
   </Personnel>
   <!-- <xsl:apply-templates select="metstdn" /> --> <!-- Metadata Name -->
   <!-- <xsl:apply-templates select="metstdv" /> --> <!-- Metadata Version --
   >
   <xsl:apply-templates select="metd" /> <!-- Metadata date -->
   <xsl:apply-templates select="metrd" /> <!-- Metadata Review date -->
   <xsl:apply-templates select="metfrd" /> <!-- Metadata Future Review date -
   ->
</xsl:template>

<xsl:template match="/metadata/metainfo/metc/cntinfo">
   <xsl:apply-templates select="cntperp" /> <!-- contact person primary -->
   <xsl:apply-templates select="cntorgp" /> <!-- contact Organization primary
   -->
   <Contact_Address><xsl:apply-templates select="cntaddr"
   /></Contact_Address>
   <xsl:apply-templates select="cntemail" />
   <xsl:apply-templates select="cntvoice" />
   <xsl:apply-templates select="cntfax" />
</xsl:template>

<xsl:template match="/metadata/metainfo/metc/cntinfo/cntperp">
<xsl:apply-templates select="/metadata/metainfo/metc/cntinfo/cntperp/cntper"
/> <!-- contact person -->
<xsl:apply-templates select="/metadata/metainfo/metc/cntinfo/cntperp/cntorg"
/> <!-- contact organization -->
</xsl:template>
```

```
<xsl:template
match="/metadata/metainfo/metc/cntinfo/cntperp/cntper"><First_Name><xsl:value
-of select="." /></First_Name></xsl:template>
<xsl:template
match="/metadata/metainfo/metc/cntinfo/cntperp/cntorg"><Last_Name><xsl:value-
of select="." /></Last_Name></xsl:template>

<xsl:template match="cntaddr">
   <xsl:apply-templates select="address" />
   <xsl:apply-templates select="city" />
   <xsl:apply-templates select="state" />
   <xsl:apply-templates select="postal" />
   <xsl:apply-templates select="country" />
</xsl:template>

<xsl:template match="address"><Address><xsl:value-of select="."
/></Address></xsl:template>
<xsl:template match="city"><City><xsl:value-of select="."
/></City></xsl:template>
<xsl:template match="state"><Province_or_State><xsl:value-of select="."
/></Province_or_State></xsl:template>
<xsl:template match="postal"><Postal_Code><xsl:value-of select="."
/></Postal_Code></xsl:template>
<xsl:template match="country"><Country><xsl:value-of select="."
/></Country></xsl:template>

<xsl:template match="cntemail"><Email><xsl:value-of select="."
/></Email></xsl:template>
<xsl:template match="cntvoice"><Phone><xsl:value-of select="."
/></Phone></xsl:template>
<xsl:template match="cntfax"><Fax><xsl:value-of select="."
/></Fax></xsl:template>

<!-- DIF Metadata Creation Date -->
<xsl:template match="metd">
  <DIF_Creation_Date>
    <xsl:variable name="datetime" select="normalize-space(.)"/>
    <xsl:variable name="year" select="substring($datetime, 1, 4)"/>
    <xsl:variable name="month" select="substring($datetime, 5, 2)"/>
    <xsl:variable name="day" select="substring($datetime, 7, 2)"/>
    <xsl:value-of select="concat($year, '-', $month, '-', $day)" />
  </DIF_Creation_Date>
</xsl:template>

<!-- DIF Last DIF Revision Date -->
<xsl:template match="metrd">
  <Last_DIF_Revision_Date>
    <xsl:variable name="datetime" select="normalize-space(.)"/>
    <xsl:variable name="year" select="substring($datetime, 1, 4)"/>
    <xsl:variable name="month" select="substring($datetime, 5, 2)"/>
    <xsl:variable name="day" select="substring($datetime, 7, 2)"/>
    <xsl:value-of select="concat($year, '-', $month, '-', $day)" />
  </Last_DIF_Revision_Date>
</xsl:template>

<!-- DIF Future DIF Revision Date -->
<xsl:template match="metfrd">
```

```
  <Future_DIF_Review_Date>
    <xsl:variable name="datetime" select="normalize-space(.)"/>
    <xsl:variable name="year" select="substring($datetime, 1, 4)"/>
    <xsl:variable name="month" select="substring($datetime, 5, 2)"/>
    <xsl:variable name="day" select="substring($datetime, 7, 2)"/>
    <xsl:value-of select="concat($year, '-', $month, '-', $day)" />
  </Future_DIF_Review_Date>
</xsl:template>

<!--                          3. DISTRIBUTION INFORMATION
-->
<xsl:template match="distinfo">
   <xsl:apply-templates select="distrib" /> <!-- Distributor -->
   <xsl:apply-templates select="stdorder" /> <!-- Standard Order Process -->
<!-- <xsl:apply-templates select="distliab" /> DIF Use_Constraints not
Distribution field -->
</xsl:template>

<!-- DIF Data Center & Data Center Personnel Information -->
<xsl:template match="distrib">
   <Data_Center>
   <Data_Center_Name>
   <Short_Name>DOC/NOAA/NESDIS/NODC</Short_Name>
   <Long_Name>National Oceanographic Data Center, NESDIS, NOAA, U.S.
   Department of Commerce</Long_Name>
   </Data_Center_Name>
   <Data_Center_URL>
   <xsl:apply-templates select="/metadata/idinfo/citation/citeinfo/onlink" />
   </Data_Center_URL>
   <Data_Set_ID><xsl:apply-templates select="/metadata/distinfo/resdesc"
   /></Data_Set_ID>
   <Personnel>
   <Role>DATA CENTER CONTACT</Role>
   <xsl:apply-templates select="/metadata/distinfo/distrib/cntinfo" />
   </Personnel>
   </Data_Center>
</xsl:template>

<xsl:template match="/metadata/distinfo/distrib/cntinfo">
   <!-- <xsl:apply-templates select="cntperp" /> --> <!-- contact person
   primary -->
   <xsl:apply-templates select="cntorgp" /> <!-- contact Organization primary
   -->
   <Contact_Address><xsl:apply-templates select="cntaddr"
   /></Contact_Address>
   <xsl:apply-templates select="cntemail" />
   <xsl:apply-templates select="cntvoice" />
   <xsl:apply-templates select="cntfax" />
</xsl:template>

<xsl:template match="/metadata/distinfo/distrib/cntinfo/cntorgp">
<!-- <xsl:apply-templates
select="/metadata/distinfo/distrib/cntinfo/cntperp/cntper" /> --> <!--
contact person -->
<xsl:apply-templates
select="/metadata/distinfo/distrib/cntinfo/cntorgp/cntorg" /> <!-- contact
organization -->
```

```
</xsl:template>

<!-- <xsl:template
match="/metadata/distinfo/distrib/cntinfo/cntperp/cntper"><First_Name><xsl:va
lue-of select="." /></First_Name></xsl:template> -->
<xsl:template
match="/metadata/distinfo/distrib/cntinfo/cntorgp/cntorg"><Last_Name><xsl:val
ue-of select="." /></Last_Name></xsl:template>

<xsl:template match="cntaddr">
   <xsl:apply-templates select="address" />
   <xsl:apply-templates select="city" />
   <xsl:apply-templates select="state" />
   <xsl:apply-templates select="postal" />
   <xsl:apply-templates select="country" />
</xsl:template>

<xsl:template match="address"><Address><xsl:value-of select="."
/></Address></xsl:template>
<xsl:template match="city"><City><xsl:value-of select="."
/></City></xsl:template>
<xsl:template match="state"><Province_or_State><xsl:value-of select="."
/></Province_or_State></xsl:template>
<xsl:template match="postal"><Postal_Code><xsl:value-of select="."
/></Postal_Code></xsl:template>
<xsl:template match="country"><Country><xsl:value-of select="."
/></Country></xsl:template>

<xsl:template match="cntemail"><Email><xsl:value-of select="."
/></Email></xsl:template>
<xsl:template match="cntvoice"><Phone><xsl:value-of select="."
/></Phone></xsl:template>
<xsl:template match="cntfax"><Fax><xsl:value-of select="."
/></Fax></xsl:template>

<xsl:template match="resdesc">
<!-- Resource description - identifier by the which
the distributor knows the data set. // DIF Entry_ID -->
   <!-- <xsl:value-of select="." /> -->
   <xsl:value-of select="translate(.,' ','_')" />
</xsl:template>

<!-- DIF Distribution Field -->
<xsl:template match="stdorder">
   <xsl:apply-templates select="digform" />
   <xsl:apply-templates select="digform/digtopt" />
   <!--<xsl:apply-templates select="fees" /> -->
</xsl:template>

<xsl:template match="digform">
<xsl:apply-templates select="digtinfo" />
<!-- <xsl:apply-templates select="digtopt" /> -->
</xsl:template>

<xsl:template match="digtinfo">
  <Distribution>
  <Distribution_Media>ONLINE</Distribution_Media>
```

```
  <Distribution_Format>
    <xsl:apply-templates select="formname" />
    <xsl:apply-templates select="formvern" />
  </Distribution_Format>
  <Distribution_Size>
    <xsl:apply-templates select="transize" />
  </Distribution_Size>
  <Fees>
    <xsl:apply-templates select="/metadata/distinfo/stdorder/fees"/>
  </Fees>
  </Distribution>
</xsl:template>

<xsl:template match="formname"><xsl:value-of select="." /></xsl:template>
<xsl:template match="formvern"><xsl:value-of select="." /></xsl:template>
<xsl:template match="transize"><xsl:value-of select="." /></xsl:template>
<xsl:template match="fees"><xsl:value-of select="." /></xsl:template>

<!-- RELATED URLs -->
<xsl:template match="/digform/digtopt">
  <xsl:apply-templates select="/onlinopt" />
</xsl:template>

<xsl:template match="onlinopt">
<Related_URL>
  <URL_Content_Type>GET DATA</URL_Content_Type>
  <xsl:apply-templates select="computer" />
  <Description>
    <xsl:apply-templates select="accinstr" />
  </Description>
  </Related_URL>
</xsl:template>

<xsl:template match="computer">
    <xsl:apply-templates select="networka" />
</xsl:template>

<xsl:template match="networka">
  <URL><xsl:apply-templates select="networkr" /></URL>
</xsl:template>

<xsl:template match="networkr"><xsl:value-of select="." /></xsl:template>
<xsl:template match="accinstr"><xsl:value-of select="." /></xsl:template>


<!--
<Distribution>
<Distribution_Media>
<xsl:apply-templates select="offmedia" />
</Distribution_Media>
</Distribution>
-->

<!-- DIF Originating Center -->
<xsl:template
match="/metadata/distinfo/distinfo/distrib/cntinfo/cntorgp/cntorg">
  <Originating_Center><xsl:value-of select="." /></Originating_Center>
```

```
</xsl:template>

<!--                        4. DATA QUALITY INFORMATION  -->
<!-- Begin DIF Quality Field -->
<xsl:template match="dataqual">
  <Quality>
    <xsl:apply-templates select="attracc" /> <!-- Attribute Accuracy Report --
    >
    <xsl:apply-templates select="logic" /> <!-- Logical Consistency Report -->
    <xsl:apply-templates select="complete" /> <!-- Completeness Report -->
    <xsl:apply-templates select="lineage/procstep/procdesc" /> <!-- Process
    Description -->
  </Quality>
</xsl:template>

<xsl:template match="attracc">
   <xsl:apply-templates select="attraccr" />
</xsl:template>

<xsl:template match="attraccr"><xsl:value-of select="." /></xsl:template>
<xsl:template match="logic"><xsl:value-of select="." /></xsl:template>
<xsl:template match="complete"><xsl:value-of select="." /></xsl:template>
<xsl:template match="procdesc"><xsl:value-of select="." /></xsl:template>
<!-- End DIF Quality Field -->

<!-- end of stylesheet -->
</xsl:stylesheet>
```

16.3.2. *DIF to FGDCI stylesheet converter*

[A copy of this stylesheet (dif_to_fgdc.xls) can be found at the PICES web site.]

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="yes"?>

<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:str="http://xsltsl.org/string"
xmlns:xalan="http://xml.apache.org/xalan">

<!-- Stylesheet created by Fedele Stella (GeoConnections Discovery Portal) --
>

<!-- Stylesheet modified by Scott Ritz (SSAI,Inc.) in April 2005
     Validates against the FGDC DTD.
     Contact: ritz@gcmd.nasa.gov  -->

<!-- Stylesheet modified by Melanie F. Meaux (SSAI,Inc.) in July 2005
     Validates against the FGDC DTD.
     Contact: mmeaux@gcmd.nasa.gov  -->

<!-- Additional information:
 http://www.w3schools.com/xsl/xsl_transformation.asp
 http://www.fgdc.gov/metadata/csdgm/
 http://biology.usgs.gov/fgdc.metadata/version2/
```

```
 Command line:

 java org.apache.xalan.xslt.Process -IN DIF_*.xml -XSL dif_to_fgdc3.xsl -OUT
FGDC_*.xml

 /home/mmeaux/bin/tidy -config /home/mmeaux/bin/config.txt FGDC_*.xml >
FGDC_*_clean.xml

 mp.lnx FGDC_*_clean.xml -e FGDC_*_clean.err

 for multiple files, run
 cd in directory where files are (US_GLOBEC_NEP_0001 ...etc)
 (source .bashrc)
 csh
 dif_fgdc_mload
 dif_fgdc_clean

 DIF xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://gcmd.gsfc.nasa.gov/Aboutus/xml/dif/dif.
xsd">
 -->

<xsl:import href="string-modified.xsl"/>

<xsl:output method="xml" encoding="ISO-8859-1" omit-xml-declaration="no"
standalone="yes" indent="yes" xalan:indent-amount="3"/>

<xsl:preserve-space elements="*"/>

<xsl:template match="/DIF" xml:space="preserve">
<metadata>
 <idinfo>

<!-- begin Citation -->
      <xsl:choose>
      <xsl:when test="Data_Set_Citation">
        <xsl:apply-templates select="Data_Set_Citation"/>
      </xsl:when>
      <xsl:otherwise>
        <xsl:apply-templates select="Originating_Metadata_Node" />
      </xsl:otherwise>
      </xsl:choose>
 <!-- end Citation -->

 <!-- begin Description -->
      <descript>
      <abstract>
      <xsl:apply-templates select="Summary"/>
      </abstract>
      <purpose>Not Available</purpose>
      <xsl:choose>
       <xsl:when test="Reference">
       <xsl:apply-templates select="Reference"/>
       </xsl:when>
       <xsl:otherwise>
        <supplinf>Not Available</supplinf>
       </xsl:otherwise>
```

```
      </xsl:choose>
      </descript>
<!-- end Description -->

 <!-- begin Time Period of Content -->
       <xsl:choose>
       <xsl:when test="Temporal_Coverage">
       <xsl:apply-templates select="Temporal_Coverage"/>
       </xsl:when>
       <xsl:otherwise>
       <begdate>Unknown</begdate>
       <enddate>Unknown</enddate>
       </xsl:otherwise>
       </xsl:choose>
 <!-- end Time Period of Content -->

 <!-- begin Status -->
      <xsl:choose>
       <xsl:when test="Data_Set_Progress">
        <xsl:apply-templates select="Data_Set_Progress"/>
       </xsl:when>
       <xsl:otherwise>
        <status>
       <progress>Complete</progress> <!-- status and Update are mandatory
fields in FGDC -->
    <update>As needed</update>
     </status>
       </xsl:otherwise>
      </xsl:choose>
 <!-- end Status -->

 <!-- begin Spatial Domain -->
      <xsl:choose>
       <xsl:when test="Spatial_Coverage">
        <xsl:apply-templates select="Spatial_Coverage"/>
       </xsl:when>
       <xsl:otherwise>
       <spdom>
       <bounding>
        <westbc>Not Available</westbc>
        <eastbc>Not Available</eastbc>
        <northbc>Not Available</northbc>
        <southbc>Not Available</southbc>
       </bounding>
       </spdom>
       </xsl:otherwise>
      </xsl:choose>
 <!-- end Spatial Domain -->

 <!-- begin Keywords -->
      <keywords>
         <theme>
       <themekt>GCMD ENTRY ID</themekt>
       <xsl:apply-templates select="Entry_ID"/>
            <themekt>GCMD SCIENCE PARAMETERS</themekt>
            <xsl:apply-templates select="Parameters"/>
       <themekt>GCMD PLATFORM</themekt>
```

```
        <xsl:apply-templates select="Source_Name"/>
        <themekt>GCMD INSTRUMENT</themekt>
            <xsl:apply-templates select="Sensor_Name"/>
        <themekt>PROJECT</themekt>
            <xsl:apply-templates select="Project"/>
        <themekt>AUXILIARY KEYWORDS</themekt>
            <xsl:apply-templates select="Keyword"/>
        <themekt>ISO TOPIC CATEGORY</themekt>
            <xsl:apply-templates select="ISO_Topic_Category"/>
        <themekt>DATA SET LANGUAGE</themekt>
            <xsl:apply-templates select="Data_Set_Language"/>
          </theme>
    <xsl:choose>
    <xsl:when test="Location">
          <place>
            <placekt>GCMD</placekt>
            <xsl:apply-templates select="Location"/>
          </place>
    </xsl:when>
        </xsl:choose>
    <xsl:choose>
    <xsl:when test="/DIF/Data_Resolution/Temporal_Resolution">
     <temporal>
            <tempkt>GCMD</tempkt>
            <xsl:apply-templates
select="/DIF/Data_Resolution/Temporal_Resolution"/>
         </temporal>
         </xsl:when>
         </xsl:choose>
      </keywords>
 <!-- end Keywords -->

 <!-- begin Access & Use Constraints -->
      <xsl:choose>
      <xsl:when test="Access_Constraints">
        <xsl:apply-templates select="Access_Constraints"/>
      </xsl:when>
      <xsl:otherwise>
        <accconst>Not Available</accconst>
      </xsl:otherwise>
      </xsl:choose>

      <xsl:choose>
      <xsl:when test="Use_Constraints">
        <xsl:apply-templates select="Use_Constraints"/>
      </xsl:when>
      <xsl:otherwise>
        <useconst>Not Available</useconst>
      </xsl:otherwise>
      </xsl:choose>
 <!-- end Access & Use Constraints -->

 <!-- begin Point of Contact -->
      <xsl:if test="Personnel/Role/text() = 'TECHNICAL CONTACT'">
      <ptcontac>
       <cntinfo>
          <xsl:apply-templates select="Personnel[Role='TECHNICAL CONTACT']"/>
```

```
        </cntinfo>
       </ptcontac>
       </xsl:if>
       <!--
       <xsl:if test="Personnel/Role/text() = 'INVESTIGATOR'">
       <ptcontac>
       <cntinfo>
           <xsl:apply-templates select="Personnel[Role='INVESTIGATOR']"/>
       </cntinfo>
       </ptcontac>
       </xsl:if>
       -->
 <!-- end Point of Contact -->

 <!-- begin Browse Graphic-->
      <xsl:apply-templates select="Multimedia_Sample"/>
 <!-- end Browse Graphic -->


 <!-- begin Cross Reference -->
 <!--      <xsl:apply-templates select="Reference"/> -->
 <!-- end Cross Reference -->


 </idinfo>

<!-- begin Data Quality information -->
 <dataqual>
       <xsl:choose>
        <xsl:when test="Quality">
          <xsl:apply-templates select="Quality"/>
    </xsl:when>
        <xsl:otherwise>
      <attracc>
             <attraccr>Not Available</attraccr>
          </attracc>
          <logic>Not Available</logic> <!-- Mandatory in data quality -->
          <complete>Not Available</complete> <!-- Mandatory in data quality -
->
          <lineage>
            <procstep>
       <procdesc>Not Available</procdesc>
            <procdate>Unknown</procdate>
             </procstep>
         </lineage> <!-- Mandatory in data quality -->
        </xsl:otherwise>
       </xsl:choose>
 </dataqual>
 <!-- end Data Quality information -->

 <!-- begin Spatial Reference information -->
 <spref>
     <xsl:choose>
      <xsl:when test="Data_Resolution">
        <xsl:apply-templates select="Data_Resolution"/>
      </xsl:when>
      <!--
```

```
      <xsl:otherwise>
        <horizsys>
    <geograph>
    <latres>Not Available</latres>
    <longres>Not Available</longres>
    <geogunit>Decimal degrees</geogunit>
    </geograph>
    </horizsys>
    <vertdef>
      <altsys>
            <altdatum>Not Available</altdatum>
            <altres>Not Available</altres>
            <altunits>Not Available</altunits>
            <altenc>Implicit coordinate</altenc>
          </altsys>
          <depthsys>
            <depthdn>Not Available</depthdn>
            <depthres>Not Available</depthres>
            <depthdu>Not Available</depthdu>
            <depthem>Implicit coordinate</depthem>
          </depthsys>
    </vertdef>
      </xsl:otherwise>
      -->
      </xsl:choose>
 </spref>
 <!-- end Spatial Reference information -->

 <!-- begin Distribution Info -->
 <distinfo>
       <xsl:apply-templates select="Data_Center"/> <!-- distributor -->

       <xsl:choose>
       <xsl:when test="Data_Center/Data_Set_ID">
        <xsl:apply-templates select="Data_Center/Data_Set_ID"/> <!-- resource
description -->
       </xsl:when>
       <xsl:otherwise>
        <resdesc>Not Available</resdesc>
       </xsl:otherwise>
       </xsl:choose>

       <distliab>Not Available</distliab>

       <stdorder>
       <digform>
       <xsl:choose>
       <xsl:when test="Distribution">
          <digtinfo>
            <xsl:apply-templates select="Distribution"/>
      </digtinfo>
       </xsl:when>
       <xsl:otherwise>
          <digtinfo>
      <formname>Not Available</formname> <!-- Mandatory field -->
      </digtinfo>
       </xsl:otherwise>
```

```
      </xsl:choose>

      <xsl:choose>
      <xsl:when test="/DIF/Data_Center/Data_Center_URL">
         <digtopt>
           <xsl:apply-templates select="/DIF/Data_Center/Data_Center_URL"/>
         </digtopt>
      </xsl:when>
      <xsl:otherwise>
         <digtopt/>
      </xsl:otherwise>
      </xsl:choose>

      <xsl:choose>
      <xsl:when test="Related_URL">
           <xsl:apply-templates select="Related_URL"/>
      </xsl:when>
      <xsl:otherwise>
         <digtopt/>
      </xsl:otherwise>
      </xsl:choose>

      </digform>

      <xsl:choose>
      <xsl:when test="/DIF/Distribution/Fees">
           <xsl:apply-templates select="/DIF/Distribution/Fees"/>
      </xsl:when>
      <xsl:otherwise>
         <fees>Not Available</fees>
      </xsl:otherwise>
      </xsl:choose>

      </stdorder>
</distinfo>
<!-- end Distribution Info -->

<!-- begin Metadata Info -->
<metainfo>
      <xsl:choose>
      <xsl:when test="DIF_Creation_Date">
       <xsl:apply-templates select="DIF_Creation_Date"/>
      </xsl:when>
      <xsl:otherwise>
        <metd/>
      </xsl:otherwise>
      </xsl:choose>

      <xsl:apply-templates select="Last_DIF_Revision_Date"/>
      <xsl:apply-templates select="Future_DIF_Review_Date"/>

      <metc>
        <xsl:choose>
           <xsl:when test="Personnel[Role='DIF AUTHOR']">
       <cntinfo>
               <xsl:apply-templates select="Personnel[Role='DIF AUTHOR']"/>
       </cntinfo>
```

```
            </xsl:when>
            <xsl:otherwise>
            <cntinfo>
            <cntperp>
            <cntper>GCMD User Support Office</cntper>
            <cntorg>NASA Global Change Master Directory</cntorg>
            </cntperp>
            <cntaddr>
            <addrtype>Mailing and Physical Address</addrtype>
            <address>Not Available</address>
            <city>Lanham</city>
            <state>MD</state>
            <postal>20706</postal>
            <country>USA</country>
            </cntaddr>
            <cntvoice>Not Available</cntvoice>
            <cntemail>gcmduso@gcmd.gsfc.nasa.gov</cntemail>
            </cntinfo>
            </xsl:otherwise>
          </xsl:choose>
       </metc>
       <metstdn>FGDC Content Standards for Digital Geospatial
Metadata</metstdn>
       <metstdv>FGDC-STD-001-1998</metstdv>
       <mettc>local time</mettc>
 </metainfo>
<!-- end Metadata Info -->

</metadata>
</xsl:template>

<!-- <xsl:apply-templates select="Data_Set_Citation"/> -->
  <xsl:template match="Data_Set_Citation">
    <citation>
     <citeinfo>
       <xsl:choose>
         <xsl:when test="Dataset_Creator">
          <xsl:apply-templates select="Dataset_Creator"/>
         </xsl:when>
         <xsl:otherwise>
         <origin>Unknown</origin>
         </xsl:otherwise>
       </xsl:choose>
       <xsl:choose>
         <xsl:when test="Dataset_Release_Date">
          <xsl:apply-templates select="Dataset_Release_Date"/>
         </xsl:when>
         <xsl:otherwise>
          <pubdate>Unknown</pubdate>
         </xsl:otherwise>
       </xsl:choose>
       <xsl:choose>
         <xsl:when test="Dataset_Title">
          <xsl:apply-templates select="Dataset_Title"/>
         </xsl:when>
         <xsl:otherwise>
          <xsl:apply-templates select="Entry_Title"/>
```

```
          </xsl:otherwise>
        </xsl:choose>
        <edition><xsl:apply-templates select="Version"/></edition>
        <geoform><xsl:apply-templates
select="Data_Presentation_Form"/></geoform>
        <serinfo>
        <sername><xsl:apply-templates select="Dataset_Series_Name"/></sername>
        <issue><xsl:apply-templates select="Issue_Identification"/></issue>
        </serinfo>
        <pubinfo>
        <pubplace><xsl:apply-templates
select="Dataset_Release_Place"/></pubplace>
        <publish><xsl:apply-templates select="Dataset_Publisher"/></publish>
        </pubinfo>
        <othercit><xsl:apply-templates
select="Other_Citation_Details"/></othercit>
        <onlink><xsl:apply-templates select="Online_Resource"/></onlink>
      </citeinfo>
     </citation>
   </xsl:template>

<!-- <xsl:apply-templates select="Originating_Metadata_Node" /> -->
<xsl:template match="Originating_Metadata_Node">
      <citation>
      <citeinfo>
      <origin>Unknown</origin>
      <pubdate>Unknown</pubdate>
      <xsl:apply-templates select="/DIF/Entry_Title"/>
      </citeinfo>
      </citation>
</xsl:template>

<!-- <xsl:apply-templates select="Dataset_Creator"/> etc-->
<xsl:template match="Dataset_Creator">
      <origin><xsl:value-of select="."/></origin>
</xsl:template>
<xsl:template match="Dataset_Release_Date">
      <pubdate><xsl:value-of select="translate(.,'-','')"/></pubdate>
</xsl:template>
<xsl:template match="Dataset_Title">
      <title><xsl:value-of select="."/></title>
</xsl:template>
<xsl:template match="Entry_Title">
      <title><xsl:value-of select="."/></title>
</xsl:template>
<xsl:template match="Version">
      <xsl:value-of select="."/>
</xsl:template>
<xsl:template match="Data_Presentation_Form">
      <xsl:value-of select="."/>
</xsl:template>
<xsl:template match="Dataset_Series_Name">
      <xsl:value-of select="."/>
</xsl:template>
<xsl:template match="Issue_Identification">
      <xsl:value-of select="."/>
</xsl:template>
```

```
<xsl:template match="Dataset_Release_Place">
     <xsl:value-of select="."/>
</xsl:template>
<xsl:template match="Dataset_Publisher">
     <xsl:value-of select="."/>
</xsl:template>
<xsl:template match="Dataset_Series_Name">
     <xsl:value-of select="."/>
</xsl:template>
<xsl:template match="Other_Citation_Details">
     <xsl:value-of select="."/>
</xsl:template>
<xsl:template match="Online_Resource">
     <xsl:value-of select="."/>
</xsl:template>

<!-- <xsl:apply-templates select="Summary"/>  -->
<xsl:template match="Summary">
     <xsl:value-of select="."/>
</xsl:template>

<!-- <xsl:apply-templates select="Reference"/>  -->
<xsl:template match="Reference">
     <supplinf><xsl:text> REFERENCE: </xsl:text><xsl:value-of
select="."/></supplinf>
</xsl:template>

<!-- <xsl:apply-templates select="Temporal_Coverage"/> -->
<xsl:template match="Temporal_Coverage">
<timeperd>
<timeinfo>
<rngdates>
   <xsl:choose>
       <xsl:when test="Start_Date">
        <xsl:apply-templates select="Start_Date"/>
       </xsl:when>
       <xsl:otherwise>
        <begdate>Unknown</begdate>
       </xsl:otherwise>
   </xsl:choose>
   <xsl:choose>
       <xsl:when test="Stop_Date">
        <xsl:apply-templates select="Stop_Date"/>
       </xsl:when>
       <xsl:otherwise>
         <enddate>Unknown</enddate>
       </xsl:otherwise>
   </xsl:choose>
</rngdates>
</timeinfo>
<current>Unknown</current>
</timeperd>
</xsl:template>

<xsl:template match="Start_Date">
     <begdate><xsl:value-of select="translate(.,'-','')"/></begdate>
</xsl:template>
```

```
<xsl:template match="Stop_Date">
     <enddate><xsl:value-of select="translate(.,'-','')"/></enddate>
</xsl:template>

<!-- <xsl:apply-templates select="Data_Set_Progress"/> -->
<xsl:template match="Data_Set_Progress">
    <status>
    <progress>
    <xsl:call-template name="str:capitalise">
        <xsl:with-param name="text" select="."/>
        <xsl:with-param name="all" select="true()"/>
        </xsl:call-template></progress>
     <update>As needed</update>
     </status>
  </xsl:template>

<!-- <xsl:apply-templates select="Spatial_Coverage"/> -->
<xsl:template match="Spatial_Coverage">
   <spdom>
   <bounding>
    <xsl:apply-templates select="Westernmost_Longitude"/>
    <xsl:apply-templates select="Easternmost_Longitude"/>
    <xsl:apply-templates select="Northernmost_Latitude"/>
    <xsl:apply-templates select="Southernmost_Latitude"/>
    </bounding>
   </spdom>
</xsl:template>
<xsl:template match="Westernmost_Longitude">
     <westbc><xsl:value-of select="."/></westbc>
</xsl:template>
<xsl:template match="Easternmost_Longitude">
     <eastbc><xsl:value-of select="."/></eastbc>
</xsl:template>
<xsl:template match="Southernmost_Latitude">
     <southbc><xsl:value-of select="."/></southbc>
</xsl:template>
<xsl:template match="Northernmost_Latitude">
     <northbc><xsl:value-of select="."/></northbc>
</xsl:template>

<!-- <xsl:apply-templates select="Parameters"/> -->
<xsl:template match="Entry_ID">
     <themekey><xsl:value-of select="."/></themekey>
</xsl:template>

<xsl:template match="Parameters">
   <themekey>
    <xsl:apply-templates select="Category"/>
    <xsl:apply-templates select="Topic"/>
    <xsl:apply-templates select="Term"/>
    <xsl:apply-templates select="Variable"/>
    <xsl:apply-templates select="Detailed_Variable"/>
   </themekey>
</xsl:template>
<xsl:template match="Category">
        <xsl:call-template name="str:to-upper">
        <xsl:with-param name="text" select="."/>
```

```
        <xsl:with-param name="all" select="true()"/>
        </xsl:call-template>
</xsl:template>
<xsl:template match="Topic"> &gt; <xsl:call-template name="str:to-upper">
        <xsl:with-param name="text" select="."/>
        <xsl:with-param name="all" select="true()"/>
        </xsl:call-template>
</xsl:template>
<xsl:template match="Term"> &gt; <xsl:call-template name="str:to-upper">
        <xsl:with-param name="text" select="."/>
        <xsl:with-param name="all" select="true()"/>
        </xsl:call-template>
</xsl:template>
<xsl:template match="Variable"> &gt; <xsl:call-template name="str:to-upper">
        <xsl:with-param name="text" select="."/>
        <xsl:with-param name="all" select="true()"/>
        </xsl:call-template>
</xsl:template>
<xsl:template match="Detailed_Variable"> &gt; <xsl:call-template
name="str:to-upper">
        <xsl:with-param name="text" select="."/>
        <xsl:with-param name="all" select="true()"/>
     </xsl:call-template>
</xsl:template>

<xsl:template match="Source_Name">
   <themekey>
    <xsl:apply-templates select="Short_Name"/>
    <xsl:apply-templates select="Long_Name"/>
   </themekey>
</xsl:template>
<xsl:template match="Sensor_Name">
   <themekey>
    <xsl:apply-templates select="Short_Name"/>
    <xsl:apply-templates select="Long_Name"/>
   </themekey>
</xsl:template>
<xsl:template match="Project">
   <themekey>
    <xsl:apply-templates select="Short_Name"/>
    <xsl:apply-templates select="Long_Name"/>
   </themekey>
</xsl:template>
<xsl:template match="Short_Name">
    <xsl:call-template name="str:to-upper">
    <xsl:with-param name="text" select="."/>
    <xsl:with-param name="all" select="true()"/>
    </xsl:call-template>
</xsl:template>
<xsl:template match="Long_Name"> &gt; <xsl:call-template name="str:to-upper">
     <xsl:with-param name="text" select="."/>
     <xsl:with-param name="all" select="true()"/>
     </xsl:call-template>
</xsl:template>

<xsl:template match="Keyword">
   <themekey>
```

```
    <xsl:call-template name="str:capitalise">
    <xsl:with-param name="text" select="."/>
    <xsl:with-param name="all" select="true()"/>
    </xsl:call-template>
  </themekey>
</xsl:template>

<xsl:template match="Location">
  <placekey>
   <xsl:apply-templates select="Location_Name"/>
   <xsl:apply-templates select="Detailed_Location"/>
  </placekey>
</xsl:template>
<xsl:template match="Location_Name">
   <xsl:call-template name="str:to-upper">
   <xsl:with-param name="text" select="normalize-space(.)"/>
   <xsl:with-param name="all" select="true()"/>
   </xsl:call-template>
</xsl:template>
<xsl:template match="Detailed_Location"> &gt; <xsl:call-template
name="str:to-upper">
   <xsl:with-param name="text" select="normalize-space(.)"/>
   <xsl:with-param name="all" select="true()"/>
   </xsl:call-template>
</xsl:template>

<xsl:template match="/DIF/Data_Resolution/Temporal_Resolution">
  <tempkey><xsl:call-template name="str:capitalise">
    <xsl:with-param name="text" select="."/>
    <xsl:with-param name="all" select="true()"/>
    </xsl:call-template>
  </tempkey>
</xsl:template>

<xsl:template match="ISO_Topic_Category">
  <themekey><xsl:call-template name="str:to-upper">
    <xsl:with-param name="text" select="."/>
    <xsl:with-param name="all" select="true()"/>
    </xsl:call-template>
  </themekey>
</xsl:template>

<xsl:template match="Data_Set_Language">
  <themekey><xsl:call-template name="str:capitalise">
    <xsl:with-param name="text" select="."/>
    <xsl:with-param name="all" select="true()"/>
    </xsl:call-template>
  </themekey>
</xsl:template>

<!-- <xsl:apply-templates select="Access and Use Constraints"/> -->
<xsl:template match="Access_Constraints">
  <accconst><xsl:value-of select="."/></accconst>
</xsl:template>
<xsl:template match="Use_Constraints">
  <useconst><xsl:value-of select="."/></useconst>
</xsl:template>
```

```
<!-- <xsl:apply-templates select="Multimedia_Sample"/> -->
<xsl:template match="Multimedia_Sample">
   <browse>
   <xsl:choose>
    <xsl:when test="URL">
     <browsen><xsl:apply-templates select="URL"/></browsen>
    </xsl:when>
    <xsl:otherwise>
         <browsen><xsl:apply-templates select="File"/></browsen>
    </xsl:otherwise>
   </xsl:choose>
   <xsl:choose>
    <xsl:when test="Description">
         <browsed><xsl:apply-templates select="Description"/></browsed>
    </xsl:when>
    <xsl:otherwise>
         <browsed><xsl:apply-templates select="Caption"/></browsed>
    </xsl:otherwise>
   </xsl:choose>
         <browset><xsl:apply-templates select="Format"/></browset>
   </browse>
  </xsl:template>

<xsl:template match="URL">
     <xsl:value-of select="."/>
</xsl:template>
<xsl:template match="File">
     <xsl:value-of select="."/>
</xsl:template>
<xsl:template match="Description">
     <xsl:value-of select="."/>
</xsl:template>
<xsl:template match="Caption">
     <xsl:value-of select="."/>
</xsl:template>
<xsl:template match="Format">
     <xsl:value-of select="."/>
</xsl:template>

<!-- <xsl:apply-templates select="Reference"/> -->
<!-- NO MATCH BECAUSE FGDC uses citation fields, GCMD one "Reference" fields
-->

<!-- <xsl:apply-templates select="Data_Resolution"/>  -->
<xsl:template match="Data_Resolution">
    <xsl:if test="Latitude_Resolution/text() != '' or
                  Longitude_Resolution/text() != ''">
      <horizsys>
       <geograph>
        <xsl:apply-templates select="Latitude_Resolution"/>
        <xsl:apply-templates select="Longitude_Resolution"/>
    <geogunit>Decimal degrees</geogunit>
       </geograph>
      </horizsys>
    </xsl:if>
```

```
    <xsl:if test="Vertical_Resolution/text() != '' ">
     <vertdef>
       <xsl:apply-templates select="Vertical_Resolution"/>
     </vertdef>
    </xsl:if>
</xsl:template>

<xsl:template match="Latitude_Resolution">
     <latres><xsl:value-of select="."/></latres>
</xsl:template>
<xsl:template match="Longitude_Resolution">
     <longres><xsl:value-of select="."/></longres>
</xsl:template>
<!-- Vertical Resolution is match to both altitude and depth resolution in
FGDC -->
<xsl:template match="Vertical_Resolution">
     <altsys>
      <altdatum>Not Available</altdatum> <!-- Mandatory field -->
      <altres><xsl:value-of select="."/></altres> <!-- Mandatory field -->
      <altunits>Not Available</altunits> <!-- Mandatory field -->
      <altenc>Implicit coordinate</altenc> <!-- Mandatory field -->
     </altsys>
     <depthsys>
      <depthdn>Not Available</depthdn> <!-- Mandatory field -->
      <depthres><xsl:value-of select="."/></depthres> <!-- Mandatory field --
>
      <depthdu>Not Available</depthdu> <!-- Mandatory field -->
      <depthem>Implicit coordinate</depthem> <!-- Mandatory field -->
     </depthsys>
</xsl:template>

<!-- <xsl:apply-templates select="Quality"/> -->
<xsl:template match="Quality">
    <attracc>
    <attraccr><xsl:value-of select="."/></attraccr>
    </attracc>
    <logic>Not Available</logic> <!-- Mandatory in data quality -->
    <complete>Not Available</complete> <!-- Mandatory in data quality -->
    <lineage>
       <procstep>
       <procdesc>Not Available</procdesc>
           <procdate>Unknown</procdate>
        </procstep>
    </lineage> <!-- Mandatory in data quality -->
</xsl:template>

<!-- <xsl:apply-templates select="Data_Center and Distribution information"/>
-->
<xsl:template match="Data_Center">
 <distrib>
   <cntinfo>
     <cntorgp>
      <cntorg>
        <xsl:apply-templates select="Data_Center_Name/Short_Name"/>
        <xsl:apply-templates select="Data_Center_Name/Long_Name"/>
      </cntorg>
      <cntper>
```

```
       <xsl:apply-templates select="/DIF/Data_Center/Personnel/First_Name"/>
       <xsl:apply-templates select="/DIF/Data_Center/Personnel/Middle_Name"/>
       <xsl:apply-templates select="/DIF/Data_Center/Personnel/Last_Name"/>
      </cntper>
     </cntorgp>
     <!-- <xsl:apply-templates select="Personnel"/>
     Contact_Information permits only one of Contact_Person_Primary or
     Contact_Organization_Primary  -->
     <!-- <xsl:apply-templates select="/DIF/Data_Center/Personnel/Role"/> -->
     <cntpos>DATA CENTER CONTACT</cntpos>
     <xsl:apply-templates select="/DIF/Data_Center/Personnel/Contact_Address"
/>
     <xsl:apply-templates select="/DIF/Data_Center/Personnel/Phone" />
     <xsl:apply-templates select="/DIF/Data_Center/Personnel/Fax" />
     <xsl:apply-templates select="/DIF/Data_Center/Personnel/Email" />
   </cntinfo>
  </distrib>
</xsl:template>

<xsl:template match="Data_Center_Name/Short_Name"><xsl:value-of
select="."/></xsl:template>
<xsl:template match="Data_Center_Name/Long_Name"> &gt; <xsl:value-of
select="."/></xsl:template>

<xsl:template match="Data_Center/Data_Set_ID"><resdesc><xsl:value-of
select="."/></resdesc></xsl:template>

<!-- Distribution information -->
<xsl:template match="Distribution">
      <xsl:choose>
       <xsl:when test="Distribution_Format">
        <xsl:apply-templates select="Distribution_Format"/>
       </xsl:when>
       <xsl:otherwise>
         <formname>Not Available</formname>
       </xsl:otherwise>
      </xsl:choose>
      <xsl:choose>
       <xsl:when test="Distribution_Size">
        <xsl:apply-templates select="Distribution_Size"/>
       </xsl:when>
       <xsl:otherwise>
       </xsl:otherwise>
      </xsl:choose>
</xsl:template>


<xsl:template match="Distribution_Format"><formname><xsl:value-of
select="."/></formname></xsl:template>
<xsl:template match="Distribution_Size"><transize><xsl:value-of
select="."/></transize></xsl:template>

<xsl:template match="/DIF/Data_Center/Data_Center_URL">
     <onlinopt>
     <computer>
     <networka>
     <networkr><xsl:value-of select="."/></networkr>
```

```
     </networka>
     </computer>
     <accinstr>DATA CENTER URL</accinstr>
     </onlinopt>
</xsl:template>

<xsl:template match="Related_URL">
      <xsl:choose>
       <xsl:when test="URL">
       <digtopt>
        <onlinopt>
         <computer>
          <networka>
           <networkr><xsl:apply-templates select="URL"/></networkr>
          </networka>
         </computer>
     <accinstr><xsl:apply-templates select="Description"/></accinstr>
        </onlinopt>
       </digtopt>
       </xsl:when>
       <xsl:otherwise>
           <digtopt><onlinopt><computer><networka><networkr>Not
Available</networkr></networka></computer></onlinopt></digtopt>
       </xsl:otherwise>
      </xsl:choose>
</xsl:template>
<xsl:template match="URL"><xsl:value-of select="."/></xsl:template>
<xsl:template match="Description"><xsl:value-of select="."/></xsl:template>

<xsl:template match="/DIF/Distribution/Fees">
     <xsl:choose>
       <xsl:when test="/DIF/Distribution/Fees">
        <xsl:apply-templates select="/DIF/Distribution/Fees"/>
       </xsl:when>
       <xsl:otherwise>
         <fees>Not Available</fees>
       </xsl:otherwise>
      </xsl:choose>
</xsl:template>

<xsl:template match="/DIF/Distribution/Fees">
    <fees><xsl:call-template name="str:capitalise">
    <xsl:with-param name="text" select="."/>
    <xsl:with-param name="all" select="true()"/>
    </xsl:call-template></fees>
</xsl:template>

<!--
<xsl:template match="/DIF/Distribution/Fees"><fees><xsl:value-of
select="."/></fees></xsl:template>
-->

<!-- Personnel/Contact Information -->
<xsl:template match="Personnel">
     <cntperp>
      <cntper>
       <xsl:apply-templates select="First_Name"/>
```

```
        <xsl:apply-templates select="Middle_Name"/>
        <xsl:apply-templates select="Last_Name"/>
       </cntper>
      </cntperp>
     <xsl:apply-templates select="Role"/>
      <xsl:choose>
         <xsl:when test="Contact_Address">
         <xsl:apply-templates select="Contact_Address"/>
         </xsl:when>
         <xsl:otherwise>
           <cntaddr>
           <addrtype>Mailing and Physical Address</addrtype>
           <address>Not Available</address>
           <city>Not Available</city>
           <state>Not Available</state>
           <postal>Not Available</postal>
           </cntaddr>
         </xsl:otherwise>
      </xsl:choose>
     <xsl:apply-templates select="Phone" />
     <xsl:apply-templates select="Fax" />
     <xsl:apply-templates select="Email" />
</xsl:template>

<xsl:template match="Contact_Address">
   <cntaddr>
     <addrtype>Mailing and Physical Address</addrtype>
      <xsl:choose>
        <xsl:when test="Address">
         <xsl:apply-templates select="Address"/>
        </xsl:when>
        <xsl:otherwise>
         <address>Not Available</address>
        </xsl:otherwise>
      </xsl:choose>
      <xsl:choose>
        <xsl:when test="City">
         <xsl:apply-templates select="City"/>
        </xsl:when>
        <xsl:otherwise>
         <city>Not Available</city>
        </xsl:otherwise>
      </xsl:choose>
      <xsl:choose>
        <xsl:when test="Province_or_State">
         <xsl:apply-templates select="Province_or_State"/>
        </xsl:when>
        <xsl:otherwise>
         <state>Not Available</state>
       </xsl:otherwise>
      </xsl:choose>
      <xsl:choose>
        <xsl:when test="Postal_Code">
         <xsl:apply-templates select="Postal_Code"/>
        </xsl:when>
        <xsl:otherwise>
         <postal>Not Available</postal>
```

```
        </xsl:otherwise>
      </xsl:choose>
      <xsl:choose>
        <xsl:when test="Country">
         <xsl:apply-templates select="Country"/>
        </xsl:when>
        <xsl:otherwise>
         <postal>Not Available</postal>
        </xsl:otherwise>
      </xsl:choose>
   </cntaddr>
  </xsl:template>

<xsl:template match="First_Name"><xsl:value-of select="."/><xsl:text>
</xsl:text></xsl:template>
<xsl:template match="Middle_Name"><xsl:value-of select="."/><xsl:text>
</xsl:text></xsl:template>
<xsl:template match="Last_Name"><xsl:value-of select="."/><xsl:text>
</xsl:text></xsl:template>
<xsl:template match="Role"><cntpos><xsl:value-of
select="."/></cntpos></xsl:template>
<xsl:template match="Phone"><cntvoice><xsl:value-of
select="."/></cntvoice></xsl:template>
<xsl:template match="Fax"><cntfax><xsl:value-of
select="."/></cntfax></xsl:template>
<xsl:template match="Email"><cntemail><xsl:value-of
select="."/></cntemail></xsl:template>
<xsl:template match="Address"><address><xsl:value-of select="."/><xsl:text>
</xsl:text></address></xsl:template>
<xsl:template match="City"><city><xsl:value-of
select="."/></city></xsl:template>
<xsl:template match="Province_or_State"><state><xsl:value-of
select="."/></state></xsl:template>
<xsl:template match="Postal_Code"><postal><xsl:value-of
select="."/></postal></xsl:template>
<xsl:template match="Country"><country><xsl:value-of
select="."/></country></xsl:template>

<!-- <xsl:apply-templates select="DIF_Creation_Date"/> -->
<xsl:template match="DIF_Creation_Date"><metd><xsl:value-of
select="translate(.,'-','')"/></metd></xsl:template>
<xsl:template match="Last_DIF_Revision_Date"><metrd><xsl:value-of
select="translate(.,'-','')"/></metrd></xsl:template>
<xsl:template match="Future_DIF_Review_Date"><metfrd><xsl:value-of
select="translate(.,'-','')"/></metfrd></xsl:template>

</xsl:stylesheet>
```

16.3.3. *String-modified stylesheet*

[A copy of this stylesheet (string-modified.xsl) can be found at the PICES web site.]

```
<?xml version="1.0"?>

<xsl:stylesheet version="1.0"
```

```
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    xmlns:doc="http://xsltsl.org/xsl/documentation/1.0"
    xmlns:str="http://xsltsl.org/string"
    extension-element-prefixes="doc str">

  <doc:reference xmlns="">
    <referenceinfo>
      <releaseinfo role="meta">
    $Id: string-modified.xsl,v 1.1 2004/06/15 19:28:11 fstella Exp $
      </releaseinfo>
      <author>
    <surname>Ball</surname>
    <firstname>Steve</firstname>
      </author>
      <copyright>
    <year>2001</year>
    <holder>Steve Ball</holder>
      </copyright>
    </referenceinfo>

    <title>String Processing</title>

    <partintro>
      <section>
    <title>Introduction</title>

    <para>This module provides templates for manipulating strings.</para>

      </section>
    </partintro>

  </doc:reference>

  <!-- Common string constants and datasets as XSL variables -->

  <!-- str:lower and str:upper contain pairs of lower and upper case
       characters. Below insanely long strings should contain the
       official lower/uppercase pairs, making this stylesheet working
       for every language on earth. Hopefully. -->
  <!-- These values are not enough, however. There are some
       exceptions, dealt with below. -->
  <xsl:variable name="xsltsl-str-lower"
select="'&#x0061;&#x0062;&#x0063;&#x0064;&#x0065;&#x0066;&#x0067;&#x0068;&#x0
069;&#x006A;&#x006B;&#x006C;&#x006D;&#x006E;&#x006F;&#x0070;&#x0071;&#x0072;&
#x0073;&#x0074;&#x0075;&#x0076;&#x0077;&#x0078;&#x0079;&#x007A;&#x00B5;&#x00E
0;&#x00E1;&#x00E2;&#x00E3;&#x00E4;&#x00E5;&#x00E6;&#x00E7;&#x00E8;&#x00E9;&#x
00EA;&#x00EB;&#x00EC;&#x00ED;&#x00EE;&#x00EF;&#x00F0;&#x00F1;&#x00F2;&#x00F3;
&#x00F4;&#x00F5;&#x00F6;&#x00F8;&#x00F9;&#x00FA;&#x00FB;&#x00FC;&#x00FD;&#x00
FE;&#x00FF;&#x0101;&#x0103;&#x0105;&#x0107;&#x0109;&#x010B;&#x010D;&#x010F;&#
x0111;&#x0113;&#x0115;&#x0117;&#x0119;&#x011B;&#x011D;&#x011F;&#x0121;&#x0123
;&#x0125;&#x0127;&#x0129;&#x012B;&#x012D;&#x012F;&#x0131;&#x0133;&#x0135;&#x0
137;&#x013A;&#x013C;&#x013E;&#x0140;&#x0142;&#x0144;&#x0146;&#x0148;&#x014B;&
#x014D;&#x014F;&#x0151;&#x0153;&#x0155;&#x0157;&#x0159;&#x015B;&#x015D;&#x015
F;&#x0161;&#x0163;&#x0165;&#x0167;&#x0169;&#x016B;&#x016D;&#x016F;&#x0171;&#x
0173;&#x0175;&#x0177;&#x017A;&#x017C;&#x017E;&#x017F;&#x0183;&#x0185;&#x0188;
&#x018C;&#x0192;&#x0195;&#x0199;&#x01A1;&#x01A3;&#x01A5;&#x01A8;&#x01AD;&#x01
B0;&#x01B4;&#x01B6;&#x01B9;&#x01BD;&#x01BF;&#x01C5;&#x01C6;&#x01C8;&#x01C9;&#
```

x01CB;ǌǎǐǒǔǖǘǚ&#x01DC
;ǝǟǡǣǥǧǩǫǭ&#x0
1EF;ǲǳǵǹǻǽǿȁȃ&
#x0205;ȇȉȋȍȏȑȓȕ&#x021
7;șțȝȟȣȥȧȩȫ&#x
022D;ȯȱȳɓɔɖɗəɛ
ɠɣɨɩɯɲɵʀʃ&#x02
88;ʊʋʒͅάέήία&#
x03B2;γδεζηθικ&#x03BB
;μνξοπρςστ&#x0
3C5;φχψωϊϋόύώ&
#x03D0;ϑϕϖϛϝϟϡϣ&#x03E
5;ϧϩϫϭϯϰϱϲϵ&#x
0430;бвгдежзий
клмнопрст&#x04
43;фхцчшщъыь&#
x044D;юяѐёђѓєѕ&#x0456
;їјљњћќѝўџ&#x0
461;ѣѥѧѩѫѭѯѱѳ&
#x0475;ѷѹѻѽѿҁҍҏ&#x049
1;ғҕҗҙқҝҟҡң&#x
04A5;ҧҩҫҭүұҳҵҷ
ҹһҽҿӂӄӈӌӑ&#x04
D3;ӕӗәӛӝӟӡӣӥ&#
x04E7;өӫӭӯӱӳӵӹ&#x0561
;բգդեզէըթժ&#x0
56B;լխծկհձղճմ&
#x0575;նշոչպջռս&#x057
E;տրցւփքօֆḁ&#x
1E03;ḅḇḉḋḍḏḑḓḕ
ḗḙḛḝḟḡḣḥḧ&#x1E
29;ḫḭḯḱḳḵḷḹḻ&#
x1E3D;ḿṁṃṅṇṉṋṍ&#x1E4F
;ṑṓṕṗṙṛṝṟṡ&#x1
E63;ṥṧṩṫṭṯṱṳṵ&
#x1E77;ṹṻṽṿẁẃẅẇ&#x1E8
9;ẋẍẏẑẓẕẛạả&#x
1EA5;ầẩẫậắằẳẵặ
ẹẻẽếềểễệỉ&#x1E
CB;ọỏốồổỗộớờ&#
x1EDF;ỡợụủứừửữ&#x1EF1
;ỳỵỷỹἀἁἂἃἄ&#x1
F05;ἆἇἐἑἒἓἔἕἠ&
#x1F21;ἢἣἤἥἦἧἰἱ&#x1F3
2;ἳἴἵἶἷὀὁὂὃ&#x
1F44;ὅὑὓὕὗὠὡὢὣ
ὤὥὦὧὰάὲέὴ&#x1F
75;ὶίὸόὺύὼώᾀ&#
x1F81;ᾂᾃᾄᾅᾆᾇᾐᾑ&#x1F92
;ᾓᾔᾕᾖᾗᾠᾡᾢᾣ&#x1
FA4;ᾥᾦᾧᾰᾱᾳιῃῐ&
#x1FD1;ῠῡῥῳⅰⅱⅲⅳ&#x217
4;ⅵⅶⅷⅸⅹⅺⅻⅼⅽ&#x
217E;ⅿⓐⓑⓒⓓⓔⓕⓖⓗ
ⓘⓙⓚⓛⓜⓝⓞⓟⓠ&#x24
E1;ⓢⓣⓤⓥⓦⓧⓨⓩａ&#
xFF42;ｃｄｅｆｇｈｉｊ&#xFF4B

;ｌｍｎｏｐｑｒｓｔ&#xF
F55;ｖｗｘｙｚ𐐨𐐩𐐪&#x104
2B;𐐬𐐭𐐮𐐯𐐰𐐱𐐲𐐳&#
x10434;𐐵𐐶𐐷𐐸𐐹𐐺𐐻&#x1043
C;𐐽𐐾𐐿𐑀𐑁𐑂𐑃𐑄&#x
10445;𐑆𐑇𐑈𐑉𐑊𐑋𐑌&#x1044D
;'"/>
  <xsl:variable name="xsltsl-str-upper"
select="'ABCDEFGH&#x0
049;JKLMNOPQR&
#x0053;TUVWXYZΜ&#x00C
0;ÁÂÃÄÅÆÇÈÉ&#x
00CA;ËÌÍÎÏÐÑÒÓ
ÔÕÖØÙÚÛÜÝ&#x00
DE;ŸĀĂĄĆĈĊČĎ&#
x0110;ĒĔĖĘĚĜĞĠ&#x0122
;ĤĦĨĪĬĮIĲĴ&#x0
136;ĹĻĽĿŁŃŅŇŊ&
#x014C;ŎŐŒŔŖŘŚŜ&#x015
E;ŠŢŤŦŨŪŬŮŰ&#x
0172;ŴŶŹŻŽSƂƄƇ
ƋƑǶƘƠƢƤƧƬ&#x01
AF;ƳƵƸƼǷǄǄǇǇ&#
x01CA;ǊǍǏǑǓǕǗǙ&#x01DB
;ƎǞǠǢǤǦǨǪǬ&#x0
1EE;ǱǱǴǸǺǼǾȀȂ&
#x0204;ȆȈȊȌȎȐȒȔ&#x021
6;ȘȚȜȞȢȤȦȨȪ&#x
022C;ȮȰȲƁƆƉƊƏƐ
ƓƔƗƖƜƝƟƦƩ&#x01
AE;ƱƲƷΙΆΈΉΊΑ&#
x0392;ΓΔΕΖΗΘΙΚ&#x039B
;ΜΝΞΟΠΡΣΣΤ&#x0
3A5;ΦΧΨΩΪΫΌΎΏ&
#x0392;ΘΦΠϚϜϞϠϢ&#x03E
4;ϦϨϪϬϮΚΡΣΕ&#x
0410;БВГДЕЖЗИЙ
КЛМНОПРСТ&#x04
23;ФХЦЧШЩЪЫЬ&#
x042D;ЮЯЀЁЂЃЄЅ&#x0406
;ЇЈЉЊЋЌЍЎЏ&#x0
460;ѢѤѦѨѪѬѮѰѲ&
#x0474;ѶѸѺѼѾҀҌҎ&#x049
0;ҒҔҖҘҚҜҞҠҢ&#x
04A4;ҦҨҪҬҮҰҲҴҶ
ҸҺҼҾӁӃӇӋӐ&#x04
D2;ӔӖӘӚӜӞӠӢӤ&#
x04E6;ӨӪӬӮӰӲӴӸ&#x0531
;ԲԳԴԵԶԷԸԹԺ&#x0
53B;ԼԽԾԿՀՁՂՃՄ&
#x0545;ՆՇՈՉՊՋՌՍ&#x054
E;ՏՐՑՒՓՔՕՖḀ&#x
1E02;ḄḆḈḊḌḎḐḒḔ
ḖḘḚḜḞḠḢḤḦ&#x1E
28;ḪḬḮḰḲḴḶḸḺ&#
x1E3C;ḾṀṂṄṆṈṊṌ&#x1E4E
;ṐṒṔṖṘṚṜṞṠ&#x1

```
E62;&#x1E64;&#x1E66;&#x1E68;&#x1E6A;&#x1E6C;&#x1E6E;&#x1E70;&#x1E72;&#x1E74;&
#x1E76;&#x1E78;&#x1E7A;&#x1E7C;&#x1E7E;&#x1E80;&#x1E82;&#x1E84;&#x1E86;&#x1E8
8;&#x1E8A;&#x1E8C;&#x1E8E;&#x1E90;&#x1E92;&#x1E94;&#x1E60;&#x1EA0;&#x1EA2;&#x
1EA4;&#x1EA6;&#x1EA8;&#x1EAA;&#x1EAC;&#x1EAE;&#x1EB0;&#x1EB2;&#x1EB4;&#x1EB6;
&#x1EB8;&#x1EBA;&#x1EBC;&#x1EBE;&#x1EC0;&#x1EC2;&#x1EC4;&#x1EC6;&#x1EC8;&#x1E
CA;&#x1ECC;&#x1ECE;&#x1ED0;&#x1ED2;&#x1ED4;&#x1ED6;&#x1ED8;&#x1EDA;&#x1EDC;&#
x1EDE;&#x1EE0;&#x1EE2;&#x1EE4;&#x1EE6;&#x1EE8;&#x1EEA;&#x1EEC;&#x1EEE;&#x1EF0
;&#x1EF2;&#x1EF4;&#x1EF6;&#x1EF8;&#x1F08;&#x1F09;&#x1F0A;&#x1F0B;&#x1F0C;&#x1
F0D;&#x1F0E;&#x1F0F;&#x1F18;&#x1F19;&#x1F1A;&#x1F1B;&#x1F1C;&#x1F1D;&#x1F28;&
#x1F29;&#x1F2A;&#x1F2B;&#x1F2C;&#x1F2D;&#x1F2E;&#x1F2F;&#x1F38;&#x1F39;&#x1F3
A;&#x1F3B;&#x1F3C;&#x1F3D;&#x1F3E;&#x1F3F;&#x1F48;&#x1F49;&#x1F4A;&#x1F4B;&#x
1F4C;&#x1F4D;&#x1F59;&#x1F5B;&#x1F5D;&#x1F5F;&#x1F68;&#x1F69;&#x1F6A;&#x1F6B;
&#x1F6C;&#x1F6D;&#x1F6E;&#x1F6F;&#x1FBA;&#x1FBB;&#x1FC8;&#x1FC9;&#x1FCA;&#x1F
CB;&#x1FDA;&#x1FDB;&#x1FF8;&#x1FF9;&#x1FEA;&#x1FEB;&#x1FFA;&#x1FFB;&#x1F88;&#
x1F89;&#x1F8A;&#x1F8B;&#x1F8C;&#x1F8D;&#x1F8E;&#x1F8F;&#x1F98;&#x1F99;&#x1F9A
;&#x1F9B;&#x1F9C;&#x1F9D;&#x1F9E;&#x1F9F;&#x1FA8;&#x1FA9;&#x1FAA;&#x1FAB;&#x1
FAC;&#x1FAD;&#x1FAE;&#x1FAF;&#x1FB8;&#x1FB9;&#x1FBC;&#x0399;&#x1FCC;&#x1FD8;&
#x1FD9;&#x1FE8;&#x1FE9;&#x1FEC;&#x1FFC;&#x2160;&#x2161;&#x2162;&#x2163;&#x216
4;&#x2165;&#x2166;&#x2167;&#x2168;&#x2169;&#x216A;&#x216B;&#x216C;&#x216D;&#x
216E;&#x216F;&#x24B6;&#x24B7;&#x24B8;&#x24B9;&#x24BA;&#x24BB;&#x24BC;&#x24BD;
&#x24BE;&#x24BF;&#x24C0;&#x24C1;&#x24C2;&#x24C3;&#x24C4;&#x24C5;&#x24C6;&#x24
C7;&#x24C8;&#x24C9;&#x24CA;&#x24CB;&#x24CC;&#x24CD;&#x24CE;&#x24CF;&#xFF21;&#
xFF22;&#xFF23;&#xFF24;&#xFF25;&#xFF26;&#xFF27;&#xFF28;&#xFF29;&#xFF2A;&#xFF2B
;&#xFF2C;&#xFF2D;&#xFF2E;&#xFF2F;&#xFF30;&#xFF31;&#xFF32;&#xFF33;&#xFF34;&#xF
F35;&#xFF36;&#xFF37;&#xFF38;&#xFF39;&#xFF3A;&#x10400;&#x10401;&#x10402;&#x104
03;&#x10404;&#x10405;&#x10406;&#x10407;&#x10408;&#x10409;&#x1040A;&#x1040B;&#
x1040C;&#x1040D;&#x1040E;&#x1040F;&#x10410;&#x10411;&#x10412;&#x10413;&#x1041
4;&#x10415;&#x10416;&#x10417;&#x10418;&#x10419;&#x1041A;&#x1041B;&#x1041C;&#x
1041D;&#x1041E;&#x1041F;&#x10420;&#x10421;&#x10422;&#x10423;&#x10424;&#x10425
;'"/>
  <xsl:variable name="xsltsl-str-digits" select="'0123456789'"/>
  <!-- space (#x20) characters, carriage returns, line feeds, or tabs. -->
  <xsl:variable name="xsltsl-str-ws" select="'&#x20;&#x9;&#xD;&#xA;'"/>

  <doc:template name="str:to-upper" xmlns="">
    <refpurpose>Make string uppercase</refpurpose>

    <refdescription>
      <para>Converts all lowercase letters to uppercase.</para>
    </refdescription>

    <refparameter>
      <variablelist>
    <varlistentry>
      <term>text</term>
      <listitem>
        <para>The string to be converted</para>
      </listitem>
    </varlistentry>
      </variablelist>
    </refparameter>

    <refreturn>
      <para>Returns string with all uppercase letters.</para>
    </refreturn>
  </doc:template>
```

```
  <xsl:template name="str:to-upper">
    <xsl:param name="text"/>

    <!-- Below exception is extracted from unicode's SpecialCasing.txt
         file. It's the german lowercase "eszett" (the thing looking
         like a greek beta) that's to become "SS" in uppercase (note:
         that are *two* characters, that's why it doesn't fit in the
         list of upper/lowercase characters). There are more
         characters in that file (103, excluding the locale-specific
         ones), but they seemed to be much less used to me and they
         add up to a hellish long stylesheet.... - Reinout -->
    <xsl:param name="modified-text">
      <xsl:call-template name="str:subst">
        <xsl:with-param name="text">
          <xsl:value-of select="$text"/>
        </xsl:with-param>
        <xsl:with-param name="replace">
          <xsl:text>&#x00DF;</xsl:text>
        </xsl:with-param>
        <xsl:with-param name="with">
          <xsl:text>&#x0053;</xsl:text>
          <xsl:text>&#x0053;</xsl:text>
        </xsl:with-param>
      </xsl:call-template>
    </xsl:param>

    <xsl:value-of select="translate($modified-text, $xsltsl-str-lower,
$xsltsl-str-upper)"/>
  </xsl:template>

  <doc:template name="str:to-lower" xmlns="">
    <refpurpose>Make string lowercase</refpurpose>

    <refdescription>
      <para>Converts all uppercase letters to lowercase.</para>
    </refdescription>

    <refparameter>
      <variablelist>
    <varlistentry>
      <term>text</term>
      <listitem>
        <para>The string to be converted</para>
      </listitem>
    </varlistentry>
      </variablelist>
    </refparameter>

    <refreturn>
      <para>Returns string with all lowercase letters.</para>
    </refreturn>
  </doc:template>

  <xsl:template name="str:to-lower">
    <xsl:param name="text"/>
```

```
    <xsl:value-of select="translate($text, $xsltsl-str-upper, $xsltsl-str-
lower)"/>
  </xsl:template>

  <doc:template name="str:capitalise" xmlns="">
    <refpurpose>Capitalise string</refpurpose>

    <refdescription>
      <para>Converts first character of string to an uppercase letter.  All
remaining characters are converted to lowercase.</para>
    </refdescription>

    <refparameter>
      <variablelist>
    <varlistentry>
      <term>text</term>
      <listitem>
        <para>The string to be capitalised</para>
      </listitem>
    </varlistentry>
    <varlistentry>
      <term>all</term>
      <listitem>
        <para>Boolean controlling whether all words in the string are
capitalised.</para>
        <para>Default is true.</para>
      </listitem>
    </varlistentry>
      </variablelist>
    </refparameter>

    <refreturn>
      <para>Returns string with first character uppcase and all remaining
characters lowercase.</para>
    </refreturn>
  </doc:template>

  <xsl:template name="str:capitalise">
    <xsl:param name="text"/>
    <xsl:param name="all" select="true()"/>
    <xsl:choose>
      <xsl:when test="$all and (contains($text, ' ') or contains($text, '   ')
or contains($text, '&#10;') or contains($text, '/') or contains($text, '-
'))">
    <xsl:variable name="firstword">
      <xsl:call-template name="str:substring-before-first">
        <xsl:with-param name="text" select="$text"/>
        <xsl:with-param name="chars" select="$xsltsl-str-ws"/>
      </xsl:call-template>
    </xsl:variable>
    <xsl:call-template name="str:capitalise">
      <xsl:with-param name="text">
        <xsl:value-of select="$firstword"/>
      </xsl:with-param>
      <xsl:with-param name="all" select="false()"/>
    </xsl:call-template>
```

```
    <xsl:value-of select="substring($text, string-length($firstword) + 1,
1)"/>
    <xsl:call-template name="str:capitalise">
      <xsl:with-param name="text">
        <xsl:value-of select="substring($text, string-length($firstword) +
2)"/>
      </xsl:with-param>
    </xsl:call-template>
      </xsl:when>

      <xsl:otherwise>
    <xsl:call-template name="str:to-upper">
      <xsl:with-param name="text" select="substring($text, 1, 1)"/>
    </xsl:call-template>
    <xsl:call-template name="str:to-lower">
      <xsl:with-param name="text" select="substring($text, 2)"/>
    </xsl:call-template>
      </xsl:otherwise>
    </xsl:choose>
  </xsl:template>

  <doc:template name="str:to-camelcase" xmlns="">
    <refpurpose>Convert a string to one camelcase word</refpurpose>

    <refdescription>
      <para>Converts a string to one lowerCamelCase or UpperCamelCase
      word, depending on the setting of the "upper"
      parameter. UpperCamelCase is also called MixedCase while
      lowerCamelCase is also called just camelCase. The template
      removes any spaces, tabs and slashes, but doesn't deal with
      other punctuation. It's purpose is to convert strings like
      "hollow timber flush door" to a term suitable as identifier or
      XML tag like "HollowTimberFlushDoor".
      </para>
    </refdescription>

    <refparameter>
      <variablelist>
    <varlistentry>
      <term>text</term>
      <listitem>
        <para>The string to be capitalised</para>
      </listitem>
    </varlistentry>
    <varlistentry>
      <term>upper</term>
      <listitem>
        <para>Boolean controlling whether the string becomes an
             UpperCamelCase word or a lowerCamelCase word.</para>
        <para>Default is true.</para>
      </listitem>
    </varlistentry>
      </variablelist>
    </refparameter>

    <refreturn>
```

```
      <para>Returns string with first character uppcase and all remaining
characters lowercase.</para>
    </refreturn>
  </doc:template>

  <xsl:template name="str:to-camelcase">
    <xsl:param name="text"/>
    <xsl:param name="upper" select="true()"/>
    <!-- First change all 'strange' characters to spaces -->
    <xsl:param name="string-with-only-spaces">
      <xsl:value-of select="translate($text,concat($xsltsl-str-ws,'/'),'
')"/>
    </xsl:param>
    <!-- Then process them -->
    <xsl:param name="before-space-removal">
      <xsl:variable name="firstword">
        <xsl:call-template name="str:substring-before-first">
          <xsl:with-param name="text" select="$string-with-only-spaces"/>
          <xsl:with-param name="chars" select="$xsltsl-str-ws"/>
        </xsl:call-template>
      </xsl:variable>
      <xsl:choose>
        <xsl:when test="$upper">
          <xsl:call-template name="str:to-upper">
            <xsl:with-param name="text" select="substring($firstword, 1,
1)"/>
          </xsl:call-template>
          <xsl:call-template name="str:to-lower">
            <xsl:with-param name="text" select="substring($firstword, 2)"/>
          </xsl:call-template>
        </xsl:when>
        <xsl:otherwise>
          <xsl:call-template name="str:to-upper">
            <xsl:with-param name="text" select="$firstword"/>
          </xsl:call-template>
        </xsl:otherwise>
      </xsl:choose>

      <xsl:call-template name="str:capitalise">
        <xsl:with-param name="text">
          <xsl:value-of select="substring($string-with-only-spaces, string-
length($firstword) + 2)"/>
        </xsl:with-param>
        <xsl:with-param name="all" select="true()"/>
      </xsl:call-template>
    </xsl:param>
    <xsl:value-of select="translate($before-space-removal,' ','')"/>
  </xsl:template>

  <doc:template name="str:substring-before-first" xmlns="">
    <refpurpose>String extraction</refpurpose>

    <refdescription>
      <para>Extracts the portion of string 'text' which occurs before any of
the characters in string 'chars'.</para>
    </refdescription>
```

```
    <refparameter>
      <variablelist>
    <varlistentry>
      <term>text</term>
      <listitem>
        <para>The string from which to extract a substring.</para>
      </listitem>
    </varlistentry>
    <varlistentry>
      <term>chars</term>
      <listitem>
        <para>The string containing characters to find.</para>
      </listitem>
    </varlistentry>
      </variablelist>
    </refparameter>

    <refreturn>
      <para>Returns string.</para>
    </refreturn>
  </doc:template>

  <xsl:template name="str:substring-before-first">
    <xsl:param name="text"/>
    <xsl:param name="chars"/>

    <xsl:choose>

      <xsl:when test="string-length($text) = 0"/>

      <xsl:when test="string-length($chars) = 0">
    <xsl:value-of select="$text"/>
      </xsl:when>

      <xsl:when test="contains($text, substring($chars, 1, 1))">
    <xsl:variable name="this" select="substring-before($text,
substring($chars, 1, 1))"/>
    <xsl:variable name="rest">
      <xsl:call-template name="str:substring-before-first">
        <xsl:with-param name="text" select="$text"/>
        <xsl:with-param name="chars" select="substring($chars, 2)"/>
      </xsl:call-template>
    </xsl:variable>

    <xsl:choose>
      <xsl:when test="string-length($this) &lt; string-length($rest)">
        <xsl:value-of select="$this"/>
      </xsl:when>
      <xsl:otherwise>
        <xsl:value-of select="$rest"/>
      </xsl:otherwise>
    </xsl:choose>
      </xsl:when>

      <xsl:otherwise>
    <xsl:call-template name="str:substring-before-first">
      <xsl:with-param name="text" select="$text"/>
```

```
      <xsl:with-param name="chars" select="substring($chars, 2)"/>
    </xsl:call-template>
      </xsl:otherwise>

    </xsl:choose>
  </xsl:template>

  <doc:template name="str:substring-after-last" xmlns="">
    <refpurpose>String extraction</refpurpose>

    <refdescription>
      <para>Extracts the portion of string 'text' which occurs after the last
of the character in string 'chars'.</para>
    </refdescription>

    <refparameter>
      <variablelist>
    <varlistentry>
      <term>text</term>
      <listitem>
        <para>The string from which to extract a substring.</para>
      </listitem>
    </varlistentry>
    <varlistentry>
      <term>chars</term>
      <listitem>
        <para>The string containing characters to find.</para>
      </listitem>
    </varlistentry>
      </variablelist>
    </refparameter>

    <refreturn>
      <para>Returns string.</para>
    </refreturn>
  </doc:template>

  <xsl:template name="str:substring-after-last">
    <xsl:param name="text"/>
    <xsl:param name="char"/>

    <xsl:choose>

      <xsl:when test="contains($text, $char)">
        <xsl:variable name="last" select="substring-after($text, $char)"/>

    <xsl:choose>
      <xsl:when test="contains($last, $char)">
        <xsl:call-template name="str:substring-after-last">
          <xsl:with-param name="text" select="$last"/>
          <xsl:with-param name="char" select="$char"/>
        </xsl:call-template>
      </xsl:when>
      <xsl:otherwise>
        <xsl:value-of select="$last"/>
      </xsl:otherwise>
    </xsl:choose>
```

```
      </xsl:when>

      <xsl:otherwise>
        <xsl:value-of select="$text"/>
      </xsl:otherwise>

    </xsl:choose>
  </xsl:template>

  <doc:template name="str:substring-before-last" xmlns="">
    <refpurpose>String extraction</refpurpose>

    <refdescription>
      <para>Extracts the portion of string 'text' which occurs before the
first character of the last occurance of string 'chars'.</para>
    </refdescription>

    <refparameter>
      <variablelist>
    <varlistentry>
      <term>text</term>
      <listitem>
        <para>The string from which to extract a substring.</para>
      </listitem>
    </varlistentry>
    <varlistentry>
      <term>chars</term>
      <listitem>
        <para>The string containing characters to find.</para>
      </listitem>
    </varlistentry>
      </variablelist>
    </refparameter>

    <refreturn>
      <para>Returns string.</para>
    </refreturn>
  </doc:template>

  <xsl:template name="str:substring-before-last">
    <xsl:param name="text"/>
    <xsl:param name="chars"/>

    <xsl:choose>

      <xsl:when test="string-length($text) = 0"/>

      <xsl:when test="string-length($chars) = 0">
    <xsl:value-of select="$text"/>
      </xsl:when>

      <xsl:when test="contains($text, $chars)">
    <xsl:call-template name="str:substring-before-last-aux">
      <xsl:with-param name="text" select="$text"/>
      <xsl:with-param name="chars" select="$chars"/>
    </xsl:call-template>
      </xsl:when>
```

```
    <xsl:otherwise>
      <xsl:value-of select="$text"/>
    </xsl:otherwise>

  </xsl:choose>
</xsl:template>

<xsl:template name="str:substring-before-last-aux">
  <xsl:param name="text"/>
  <xsl:param name="chars"/>

  <xsl:choose>
    <xsl:when test="string-length($text) = 0"/>

    <xsl:when test="contains($text, $chars)">
  <xsl:variable name="after">
    <xsl:call-template name="str:substring-before-last-aux">
      <xsl:with-param name="text" select="substring-after($text, $chars)"/>
      <xsl:with-param name="chars" select="$chars"/>
    </xsl:call-template>
  </xsl:variable>

  <xsl:value-of select="substring-before($text, $chars)"/>
  <xsl:if test="string-length($after) &gt; 0">
    <xsl:value-of select="$chars"/>
    <xsl:copy-of select="$after"/>
  </xsl:if>
    </xsl:when>

    <xsl:otherwise/>
  </xsl:choose>
</xsl:template>

<doc:template name="str:subst" xmlns="">
  <refpurpose>String substitution</refpurpose>

  <refdescription>
    <para>Substitute 'replace' for 'with' in string 'text'.</para>
  </refdescription>

  <refparameter>
    <variablelist>
  <varlistentry>
    <term>text</term>
    <listitem>
      <para>The string upon which to perform substitution</para>
    </listitem>
  </varlistentry>
  <varlistentry>
    <term>replace</term>
    <listitem>
      <para>The string to substitute</para>
    </listitem>
  </varlistentry>
  <varlistentry>
    <term>with</term>
```

```
      <listitem>
        <para>The string to be substituted</para>
      </listitem>
    </varlistentry>
      </variablelist>
    </refparameter>

    <refreturn>
      <para>Returns string.</para>
    </refreturn>
  </doc:template>

  <xsl:template name="str:subst">
    <xsl:param name="text"/>
    <xsl:param name="replace"/>
    <xsl:param name="with"/>

    <xsl:choose>
      <xsl:when test="string-length($replace) = 0">
        <xsl:value-of select="$text"/>
      </xsl:when>
      <xsl:when test="contains($text, $replace)">

    <xsl:variable name="before" select="substring-before($text, $replace)"/>
    <xsl:variable name="after" select="substring-after($text, $replace)"/>

    <xsl:value-of select="$before"/>
    <xsl:value-of select="$with"/>
        <xsl:call-template name="str:subst">
      <xsl:with-param name="text" select="$after"/>
      <xsl:with-param name="replace" select="$replace"/>
      <xsl:with-param name="with" select="$with"/>
    </xsl:call-template>
      </xsl:when>
      <xsl:otherwise>
        <xsl:value-of select="$text"/>
      </xsl:otherwise>
    </xsl:choose>
  </xsl:template>

  <doc:template name="str:count-substring" xmlns="">
    <refpurpose>Count Substrings</refpurpose>

    <refdescription>
      <para>Counts the number of times a substring occurs in a string.  This
can also counts the number of times a character occurs in a string, since a
character is simply a string of length 1.</para>
    </refdescription>

    <example>
      <title>Counting Lines</title>
      <programlisting><![CDATA[
<xsl:call-template name="str:count-substring">
  <xsl:with-param name="text" select="$mytext"/>
  <xsl:with-param name="chars" select="'&#x0a;'"/>
</xsl:call-template>
]]></programlisting>
```

```
    </example>

    <refparameter>
      <variablelist>
    <varlistentry>
      <term>text</term>
      <listitem>
        <para>The source string.</para>
      </listitem>
    </varlistentry>
    <varlistentry>
      <term>chars</term>
      <listitem>
        <para>The substring to count.</para>
      </listitem>
    </varlistentry>
      </variablelist>
    </refparameter>

    <refreturn>
      <para>Returns a non-negative integer value.</para>
    </refreturn>
  </doc:template>

  <xsl:template name="str:count-substring">
    <xsl:param name="text"/>
    <xsl:param name="chars"/>

    <xsl:choose>
      <xsl:when test="string-length($text) = 0 or string-length($chars) = 0">
    <xsl:text>0</xsl:text>
      </xsl:when>
      <xsl:when test="contains($text, $chars)">
    <xsl:variable name="remaining">
      <xsl:call-template name="str:count-substring">
        <xsl:with-param name="text" select="substring-after($text, $chars)"/>
        <xsl:with-param name="chars" select="$chars"/>
      </xsl:call-template>
    </xsl:variable>
    <xsl:value-of select="$remaining + 1"/>
      </xsl:when>
      <xsl:otherwise>
    <xsl:text>0</xsl:text>
      </xsl:otherwise>
    </xsl:choose>
  </xsl:template>

  <doc:template name="str:substring-after-at" xmlns="">
    <refpurpose>String extraction</refpurpose>
    <refdescription>
      <para>Extracts the portion of a 'char' delimited 'text' string "array"
at a given 'position' </para>
    </refdescription>
    <refparameter>
      <variablelist>
        <varlistentry>
          <term>text</term>
```

```
        <listitem>
          <para>The string from which to extract a substring.</para>
        </listitem>
      </varlistentry>
      <varlistentry>
        <term>chars</term>
        <listitem>
        <para>delimiters</para>
        </listitem>
      </varlistentry>
      <varlistentry>
        <term>position</term>
        <listitem>
          <para>position of the elements</para>
        </listitem>
      </varlistentry>
    </variablelist>
  </refparameter>
  <refreturn>
    <para>Returns string.</para>
  </refreturn>
</doc:template>


<xsl:template name="str:substring-after-at">
  <xsl:param name="text"/>
  <xsl:param name="char"/>
  <xsl:param name="position"/>
  <xsl:choose>
    <xsl:when test="$position = 0 or not(contains($text ,$char))">
      <xsl:choose>
        <xsl:when test="contains($text ,$char)">
          <xsl:value-of select="substring-before($text,$char)"/>
        </xsl:when>
        <xsl:otherwise>
          <xsl:value-of select="$text"/>
        </xsl:otherwise>
      </xsl:choose>
    </xsl:when>
    <xsl:when test="contains($text ,$char) and $position &gt; 0">
      <xsl:variable name="last" select="substring-after($text,$char)"/>

      <xsl:choose>
        <xsl:when test="$position &gt; 0">
          <xsl:call-template name="str:substring-after-at">
            <xsl:with-param name="text" select="$last"/>
            <xsl:with-param name="char" select="$char"/>
            <xsl:with-param name="position" select="$position - 1"/>
          </xsl:call-template>
        </xsl:when>
      </xsl:choose>
    </xsl:when>
  </xsl:choose>
</xsl:template>

<doc:template name="str:insert-at" xmlns="">
  <refpurpose>String insertion</refpurpose>
```

```
    <refdescription>
      <para>Insert 'chars' into "text' at any given "position'</para>
    </refdescription>
    <refparameter>
      <variablelist>
        <varlistentry>
          <term>text</term>
          <listitem>
            <para>The string upon which to perform insertion</para>
          </listitem>
        </varlistentry>
        <varlistentry>
          <term>position</term>
          <listitem>
            <para>the position where insertion will be performed</para>
          </listitem>
        </varlistentry>
        <varlistentry>
          <term>with</term>
          <listitem>
            <para>The string to be inserted</para>
          </listitem>
        </varlistentry>
      </variablelist>
    </refparameter>
    <refreturn>
      <para>Returns string.</para>
    </refreturn>
  </doc:template>

  <xsl:template name="str:insert-at">
    <xsl:param name="text"/>
    <xsl:param name="position"/>
    <xsl:param name="chars"/>

    <xsl:variable name="firstpart" select="substring($text, 0, $position)"/>
    <xsl:variable name="secondpart" select="substring($text, $position,
string-length($text))"/>

    <xsl:value-of select="concat($firstpart, $chars, $secondpart)"/>
  </xsl:template>


  <doc:template name="str:backward" xmlns="">
    <refpurpose>String reversal</refpurpose>

    <refdescription>
      <para>Reverse the content of a given string</para>
    </refdescription>

    <refparameter>
      <variablelist>
        <varlistentry>
          <term>text</term>
          <listitem>
            <para>The string to be reversed</para>
          </listitem>
```

```
        </varlistentry>
      </variablelist>
    </refparameter>

    <refreturn>
      <para>Returns string.</para>
    </refreturn>
  </doc:template>

  <xsl:template name="str:backward">
    <xsl:param name="text"/>
    <xsl:variable name="mirror">
      <xsl:call-template name="str:build-mirror">
        <xsl:with-param name="text" select="$text"/>
        <xsl:with-param name="position" select="string-length($text)"/>
      </xsl:call-template>
    </xsl:variable>
    <xsl:value-of select="substring($mirror, string-length($text) + 1,
string-length($text))"/>
  </xsl:template>

  <xsl:template name="str:build-mirror">
    <xsl:param name="text"/>
    <xsl:param name="position"/>

    <xsl:choose>
      <xsl:when test="$position &gt; 0">
        <xsl:call-template name="str:build-mirror">
          <xsl:with-param name="text" select="concat($text, substring($text,
$position, 1))"/>
          <xsl:with-param name="position" select="$position - 1"/>
        </xsl:call-template>
      </xsl:when>
      <xsl:otherwise>
        <xsl:value-of select="$text"/>
      </xsl:otherwise>
    </xsl:choose>
  </xsl:template>

  <doc:template name="str:character-first" xmlns="">
    <refpurpose>Find first occurring character in a string</refpurpose>

    <refdescription>
      <para>Finds which of the given characters occurs first in a
string.</para>
    </refdescription>

    <refparameter>
      <variablelist>
    <varlistentry>
      <term>text</term>
      <listitem>
        <para>The source string.</para>
      </listitem>
    </varlistentry>
    <varlistentry>
      <term>chars</term>
```

```
      <listitem>
        <para>The characters to search for.</para>
      </listitem>
    </varlistentry>
      </variablelist>
    </refparameter>
  </doc:template>

  <xsl:template name="str:character-first">
    <xsl:param name="text"/>
    <xsl:param name="chars"/>

    <xsl:choose>
      <xsl:when test="string-length($text) = 0 or string-length($chars) =
0"/>

      <xsl:when test="contains($text, substring($chars, 1, 1))">
    <xsl:variable name="next-character">
      <xsl:call-template name="str:character-first">
        <xsl:with-param name="text" select="$text"/>
        <xsl:with-param name="chars" select="substring($chars, 2)"/>
      </xsl:call-template>
    </xsl:variable>

    <xsl:choose>
      <xsl:when test="string-length($next-character)">
        <xsl:variable name="first-character-position" select="string-
length(substring-before($text, substring($chars, 1, 1)))"/>
        <xsl:variable name="next-character-position" select="string-
length(substring-before($text, $next-character))"/>

        <xsl:choose>
          <xsl:when test="$first-character-position &lt; $next-character-
position">
       <xsl:value-of select="substring($chars, 1, 1)"/>
          </xsl:when>
          <xsl:otherwise>
       <xsl:value-of select="$next-character"/>
          </xsl:otherwise>
        </xsl:choose>
      </xsl:when>
      <xsl:otherwise>
        <xsl:value-of select="substring($chars, 1, 1)"/>
      </xsl:otherwise>
    </xsl:choose>
      </xsl:when>
      <xsl:otherwise>
    <xsl:call-template name="str:character-first">
      <xsl:with-param name="text" select="$text"/>
      <xsl:with-param name="chars" select="substring($chars, 2)"/>
    </xsl:call-template>
      </xsl:otherwise>
    </xsl:choose>
  </xsl:template>

  <doc:template name="str:string-match" xmlns="">
    <refpurpose>Match A String To A Pattern</refpurpose>
```

```
    <refdescription>
      <para>Performs globbing-style pattern matching on a string.</para>
    </refdescription>

    <example>
      <title>Match Pattern</title>
      <programlisting><![CDATA[
<xsl:call-template name="str:string-match">
  <xsl:with-param name="text" select="$mytext"/>
  <xsl:with-param name="pattern" select="'abc*def?g'"/>
</xsl:call-template>
]]></programlisting>
    </example>

    <refparameter>
      <variablelist>
    <varlistentry>
      <term>text</term>
      <listitem>
        <para>The source string.</para>
      </listitem>
    </varlistentry>
    <varlistentry>
      <term>pattern</term>
      <listitem>
        <para>The pattern to match against.  Certain characters have special
meaning:</para>
        <variablelist>
          <varlistentry>
        <term>*</term>
        <listitem>
          <para>Matches zero or more characters.</para>
        </listitem>
          </varlistentry>
          <varlistentry>
        <term>?</term>
        <listitem>
          <para>Matches a single character.</para>
        </listitem>
          </varlistentry>
          <varlistentry>
        <term>\</term>
        <listitem>
          <para>Character escape.  The next character is taken as a literal
character.</para>
        </listitem>
          </varlistentry>
        </variablelist>
      </listitem>
    </varlistentry>
      </variablelist>
    </refparameter>

    <refreturn>
      <para>Returns "1" if the string matches the pattern, "0"
otherwise.</para>
```

```
    </refreturn>
  </doc:template>

  <xsl:template name="str:string-match">
    <xsl:param name="text"/>
    <xsl:param name="pattern"/>

    <xsl:choose>
      <xsl:when test="$pattern = '*'">
    <!-- Special case: always matches -->
    <xsl:text>1</xsl:text>
      </xsl:when>
      <xsl:when test="string-length($text) = 0 and string-length($pattern) =
0">
    <xsl:text>1</xsl:text>
      </xsl:when>
      <xsl:when test="string-length($text) = 0 or string-length($pattern) =
0">
    <xsl:text>0</xsl:text>
      </xsl:when>
      <xsl:otherwise>
    <xsl:variable name='before-special'>
      <xsl:call-template name='str:substring-before-first'>
        <xsl:with-param name='text' select='$pattern'/>
        <xsl:with-param name='chars' select='"*?\"'/>
      </xsl:call-template>
    </xsl:variable>
    <xsl:variable name='special'>
      <xsl:call-template name='str:character-first'>
        <xsl:with-param name='text' select='$pattern'/>
        <xsl:with-param name='chars' select='"*?\"'/>
      </xsl:call-template>
    </xsl:variable>

    <xsl:variable name='new-text' select='substring($text, string-
length($before-special) + 1)'/>
    <xsl:variable name='new-pattern' select='substring($pattern, string-
length($before-special) + 1)'/>

    <xsl:choose>
      <xsl:when test="not(starts-with($text, $before-special))">
        <!-- Verbatim characters don't match -->
        <xsl:text>0</xsl:text>
      </xsl:when>

      <xsl:when test="$special = '*' and string-length($new-pattern) = 1">
        <xsl:text>1</xsl:text>
      </xsl:when>
      <xsl:when test="$special = '*'">
        <xsl:call-template name='str:match-postfix'>
          <xsl:with-param name='text' select='$new-text'/>
          <xsl:with-param name='pattern' select='substring($new-pattern, 2)'/>
        </xsl:call-template>
      </xsl:when>

      <xsl:when test="$special = '?'">
        <xsl:call-template name="str:string-match">
```

```
          <xsl:with-param name='text' select='substring($new-text, 2)'/>
          <xsl:with-param name='pattern' select='substring($new-pattern, 2)'/>
        </xsl:call-template>
      </xsl:when>

      <xsl:when test="$special = '\' and substring($new-text, 1, 1) =
substring($new-pattern, 2, 1)">
        <xsl:call-template name="str:string-match">
          <xsl:with-param name='text' select='substring($new-text, 2)'/>
          <xsl:with-param name='pattern' select='substring($new-pattern, 3)'/>
        </xsl:call-template>
      </xsl:when>
      <xsl:when test="$special = '\' and substring($new-text, 1, 1) !=
substring($new-pattern, 2, 1)">
        <xsl:text>0</xsl:text>
      </xsl:when>

      <xsl:otherwise>
        <!-- There were no special characters in the pattern -->
        <xsl:choose>
          <xsl:when test='$text = $pattern'>
        <xsl:text>1</xsl:text>
          </xsl:when>
          <xsl:otherwise>
        <xsl:text>0</xsl:text>
          </xsl:otherwise>
        </xsl:choose>
      </xsl:otherwise>
    </xsl:choose>
      </xsl:otherwise>
    </xsl:choose>
  </xsl:template>

  <xsl:template name="str:match-postfix">
    <xsl:param name="text"/>
    <xsl:param name="pattern"/>

    <xsl:variable name='result'>
      <xsl:call-template name='str:string-match'>
    <xsl:with-param name='text' select='$text'/>
    <xsl:with-param name='pattern' select='$pattern'/>
      </xsl:call-template>
    </xsl:variable>

    <xsl:choose>
      <xsl:when test='$result = "1"'>
    <xsl:value-of select='$result'/>
      </xsl:when>
      <xsl:when test='string-length($text) = 0'>
    <xsl:text>0</xsl:text>
      </xsl:when>
      <xsl:otherwise>
    <xsl:call-template name='str:match-postfix'>
      <xsl:with-param name='text' select='substring($text, 2)'/>
      <xsl:with-param name='pattern' select='$pattern'/>
    </xsl:call-template>
      </xsl:otherwise>
```

```
    </xsl:choose>

  </xsl:template>

  <doc:template name="str:generate-string" xmlns="">
    <refpurpose>Create A Repeating Sequence of Characters</refpurpose>

    <refdescription>
      <para>Repeats a string a given number of times.</para>
    </refdescription>

    <refparameter>
      <variablelist>
   <varlistentry>
     <term>text</term>
     <listitem>
       <para>The string to repeat.</para>
     </listitem>
   </varlistentry>
   <varlistentry>
     <term>count</term>
     <listitem>
       <para>The number of times to repeat the string.</para>
     </listitem>
   </varlistentry>
      </variablelist>
    </refparameter>
  </doc:template>

  <xsl:template name="str:generate-string">
    <xsl:param name="text"/>
    <xsl:param name="count"/>

    <xsl:choose>
      <xsl:when test="string-length($text) = 0 or $count &lt;= 0"/>

      <xsl:otherwise>
   <xsl:value-of select="$text"/>
   <xsl:call-template name="str:generate-string">
     <xsl:with-param name="text" select="$text"/>
     <xsl:with-param name="count" select="$count - 1"/>
   </xsl:call-template>
      </xsl:otherwise>
    </xsl:choose>
  </xsl:template>

</xsl:stylesheet>
```