Guide to Schematron Rules and Patterns

IRM Schematron Guide

Version 7

Date of Draft Release: 9 January 2012

27 February 2012

Table of Contents

Chapter 1 - Introduction

1.1 - Purpose

(U) The following documentation is informative. The actual Schematron files are the normative record. This documentation is generated from the Schematron files via XSLT it may be missing some file or pieces of a file but whatever is here other than titles came from the original file.

(U) It is envisioned that this will be a useful resource to search and read but for questions and debates the source files should be consulted.

(U) Rules are all of the format IRM-ID-XXXXX any other heading is a supporting file that may strongly influence a rule but is not actually a numbered rule.

Chapter 2 - Rules

2.1 - .//Rules/dateTimeConstraints/IRM_ID_00015.sch

Rule Description: IRM-ID-00015 [IRM-ID-00015][Error] If element ddms:dates exists without one of the attributes @ddms:created or @ddms:posted Human Readable: Every ddms:dates element must have at least one of @ddms:created or @ddms:posted.

Code Description: This rule checks that for each occurrence of ddms:dates that either @ddms:created or @ddms:posted is specified.

Schematron Code:

```
<?ICEA pattern?><!-- Notices - distEditionBlockReplace-->
<sch:pattern id="IRM-ID-00015">




        <sch:rule context="ddms:dates">
              <sch:assert test="@ddms:created or @ddms:posted" flag="error">
           [IRM-ID-00015][Error] Every ddms:date must have at least one of
@ddms:created or @ddms:posted.
        </sch:assert>
         </sch:rule>
      </sch:pattern>
```

2.2 - .//Rules/dateTimeConstraints/IRM_ID_00016.sch

Rule Description: IRM-ID-00016 [IRM-ID-00016][Error] The permissible values for the year range are 1901 through the current year for attributes @ddms:approvedOn, @ddms:dateProcessed, @ddms:receivedOn, @ddms:infoCutOff, @ddms:posted, and @ddms:created. Human Readable: Dates must be after 1901 and in the past for @ddms:approvedOn, @ddms:dateProcessed, @ddms:receivedOn, @ddms:infoCutOff, @ddms:posted, and @ddms:created.

Code Description: This pattern uses abstract rules to consolidate logic. For attributes, we make sure that each date contained within $dateList has a year value within the range $minYear and $maxYear, inclusive.

Schematron Code:

```
<?ICEA pattern?><!-- Notices - distEditionBlockReplace-->
<sch:pattern id="IRM-ID-00016">




        <!-- Use abstract rule to handle required attributes -->
    <sch:rule context="*[@ddms:approvedOn |
```

```
@ddms:dateProcessed |                          @ddms:receivedOn
|                        @ddms:posted |                          @ddms:created
|                         @ddms:infoCutOff]">
            <sch:let name="minYear" value="1901"/>
            <sch:let name="maxYear" value="$currentYear"/>
            <sch:let name="dateList" value="
(string(@ddms:approvedOn),
string(@ddms:dateProcessed),
string(@ddms:receivedOn),             string(@ddms:posted),
string(@ddms:created),              string(@ddms:infoCutOff))"/>
            <sch:let name="errMsg" value="'             [IRM-ID-00016][Error]
The permissible values for the year range are 1901 through the
current             year for attributes @ddms:approvedOn,
@ddms:dateProcessed, @ddms:receivedOn, @ddms:infoCutOff,
@ddms:posted, and @ddms:created.                          Human Readable:
Dates must be after 1901 and in the past for @ddms:approvedOn,
@ddms:dateProcessed, @ddms:receivedOn, @ddms:infoCutOff, @ddms:posted, and
@ddms:created.              '"/>
            <sch:extends rule="abs.dateListYearRangeRule"/>
         </sch:rule>
      </sch:pattern>
```

2.3 - .//Rules/dateTimeConstraints/IRM_ID_00017.sch

Rule Description: IRM-ID-00017 [IRM-ID-00017][Error] The permissible values for the year range are 1901 through 9999 for attribute @ddms:validTil. Human Readable: @ddms:validTil must be after 1901.

Code Description: This pattern uses abstract rules to consolidate logic. For attributes, we make sure that each date contained within $dateList has a year value within the range $minYear and $maxYear, inclusive.

Schematron Code:

```
<?ICEA pattern?><!-- Notices - distEditionBlockReplace-->
<sch:pattern id="IRM-ID-00017">




        <!-- Use abstract rule to handle required attributes -->
       <sch:rule context="*[@ddms:validTil]">
            <sch:let name="minYear" value="1901"/>
            <sch:let name="maxYear" value="9999"/>
                        <sch:let name="dateList"
value="(string(@ddms:validTil))"/>
            <sch:let name="errMsg" value="'             [IRM-ID-00017][Error]
The permissible values for the year range are             1901 through 9999
for attribute @ddms:validTil.                          Human Readable:
@ddms:validTil must be after 1901.             '"/>
            <sch:extends rule="abs.dateListYearRangeRule"/>
         </sch:rule>
```

```
      </sch:pattern>
```

2.4 - .//Rules/dateTimeConstraints/IRM_ID_00018.sch

Rule Description: IRM-ID-00018 [IRM-ID-00018][Error] Rule removed in V7.

Code Description:

Schematron Code:

```
<?ICEA pattern?><!-- Notices - distEditionBlockReplace-->
<sch:pattern id="IRM-ID-00018">

      </sch:pattern>
```

2.5 - .//Rules/dateTimeConstraints/IRM_ID_00019.sch

Rule Description: IRM-ID-00019 [IRM-ID-00019][Warning] @ddms:approvedOn must not be later than @ddms:created and @ddms:posted.

Code Description: This rule uses an abstract pattern to consolidate logic. It compares the date contained within the param $primaryDate to each date contained within the param $secondaryDateList (using the comparison operator contained in param $operator) and makes sure that each comparison returns true. Implementation details for the abstract pattern can be found in the abstract pattern definition file located in the Lib directory.

Schematron Code:

```
<?ICEA pattern?><!-- Notices - distEditionBlockReplace-->
<sch:pattern id="IRM-ID-00019" is-a="CompareDateTimes">




        <sch:param name="ruleText" value="'         [IRM-ID-00019][Warning]
@ddms:approvedOn must not be later than @ddms:created and
@ddms:posted.         '"/>

        <sch:param name="codeDesc" value="'         This rule uses an
abstract pattern to consolidate logic.         It compares the date contained
within the param $primaryDate to each date          contained within the
param $secondaryDateList (using the comparison operator          contained in
param $operator) and makes sure that each comparison returns          true.
Implementation details for the abstract pattern can be found in the
abstract pattern definition file located in the Lib directory.         '"/>

        <sch:param name="context" value="*[@ddms:approvedOn]"/>
        <sch:param name="primaryDate" value="@ddms:approvedOn"/>
        <sch:param name="operator" value="'&amp;lt;='"/>
```

```
         <sch:param name="secondaryDateList"
value="(@ddms:created,
@ddms:posted)"/>
         <sch:param name="flag" value="'error'"/>
      </sch:pattern>
```

2.6 - .//Rules/dateTimeConstraints/IRM_ID_00020.sch

Rule Description: IRM-ID-00020 [IRM-ID-00020][Error] @ddms:infoCutOff must not be later than @ddms:created, and @ddms:posted.

Code Description: This rule uses an abstract pattern to consolidate logic. It compares the date contained within the param $primaryDate to each date contained within the param $secondaryDateList (using the comparison operator contained in param $operator) and makes sure that each comparison returns true. Implementation details for the abstract pattern can be found in the abstract pattern definition file located in the Lib directory.

Schematron Code:

```
<?ICEA pattern?><!-- Notices - distEditionBlockReplace-->
<sch:pattern id="IRM-ID-00020" is-a="CompareDateTimes">




         <sch:param name="ruleText" value="'         [IRM-ID-00020][Error]
@ddms:infoCutOff must not be later than @ddms:created, and
@ddms:posted.          '"/>

         <sch:param name="codeDesc" value="'         This rule uses an
abstract pattern to consolidate logic.         It compares the date contained
within the param $primaryDate to each date          contained within the
param $secondaryDateList (using the comparison operator          contained in
param $operator) and makes sure that each comparison returns          true.
Implementation details for the abstract pattern can be found in the
abstract pattern definition file located in the Lib directory.         '"/>

         <sch:param name="context" value="*[@ddms:infoCutOff]"/>
         <sch:param name="primaryDate" value="@ddms:infoCutOff"/>
         <sch:param name="operator" value="'&lt;='"/>
         <sch:param name="secondaryDateList"
value="(@ddms:created,
@ddms:posted)"/>
         <sch:param name="flag" value="'error'"/>
      </sch:pattern>
```

2.7 - .//Rules/dateTimeConstraints/IRM_ID_00021.sch

Rule Description: IRM-ID-00021 [IRM-ID-00021][Warning] @ddms:validTil must not be earlier than @ddms:created, @ddms:posted, @ddms:infoCutOff, and @ddms:approvedOn.

Code Description: This rule uses an abstract pattern to consolidate logic. It compares the date contained within the param $primaryDate to each date contained within the param $secondaryDateList (using the comparison operator contained in param $operator) and makes sure that each comparison returns true. Implementation details for the abstract pattern can be found in the abstract pattern definition file located in the Lib directory.

Schematron Code:

```
<?ICEA pattern?><!-- Notices - distEditionBlockReplace-->
<sch:pattern id="IRM-ID-00021" is-a="CompareDateTimes">




        <sch:param name="ruleText" value="'         [IRM-ID-00021][Warning]
@ddms:validTil must not be earlier than         @ddms:created, @ddms:posted,
@ddms:infoCutOff, and @ddms:approvedOn.          '"/>

        <sch:param name="codeDesc" value="'         This rule uses an
abstract pattern to consolidate logic.         It compares the date contained
within the param $primaryDate to each date          contained within the
param $secondaryDateList (using the comparison operator          contained in
param $operator) and makes sure that each comparison returns          true.
Implementation details for the abstract pattern can be found in the
abstract pattern definition file located in the Lib directory.         '"/>

        <sch:param name="context" value="*[@ddms:validTil]"/>
        <sch:param name="primaryDate" value="@ddms:validTil"/>
        <sch:param name="operator" value="'&gt;='"/>
        <sch:param name="secondaryDateList"
value="(@ddms:created,
@ddms:posted,
@ddms:infoCutOff,
@ddms:approvedOn)"/>
        <sch:param name="flag" value="'error'"/>
     </sch:pattern>
```

2.8 - .//Rules/dateTimeConstraints/IRM_ID_00022.sch

Rule Description: IRM-ID-00022 [IRM-ID-00022][Error] For any element ddms:temporalCoverage, child element ddms:start must not be later than child element ddms:end. Human Readable: For date-time ranges, the start of a range must be earlier than, or equivalent to, the end of that range.

Code Description: This rule uses an abstract pattern to consolidate logic. It compares the date contained within the param $primaryDate to each date contained within the param $secondaryDateList (using the comparison operator contained in param $operator) and makes sure that each comparison returns true. Implementation details for the abstract pattern can be found in the abstract pattern definition file located in the Lib directory.

Schematron Code:

```
<?ICEA pattern?><!-- Notices - distEditionBlockReplace-->
<sch:pattern id="IRM-ID-00022" is-a="CompareDateTimes">




        <sch:param name="ruleText" value="'         [IRM-ID-00022][Error]
For any element ddms:temporalCoverage, child element ddms:start must
not be later than child element ddms:end.         '"/>

        <sch:param name="codeDesc" value="'         This rule uses an
abstract pattern to consolidate logic.         It compares the date contained
within the param $primaryDate to each date          contained within the
param $secondaryDateList (using the comparison operator          contained in
param $operator) and makes sure that each comparison returns          true.
Implementation details for the abstract pattern can be found in the
abstract pattern definition file located in the Lib directory.         '"/>

        <sch:param name="context" value="ddms:temporalCoverage"/>
        <sch:param name="primaryDate" value="ddms:start"/>
        <sch:param name="operator" value="'&lt;='"/>
        <sch:param name="secondaryDateList" value="(ddms:end)"/>
        <sch:param name="flag" value="'error'"/>
     </sch:pattern>
```

2.9 - .//Rules/dateTimeConstraints/IRM_ID_00023.sch

Rule Description: IRM-ID-00023 [IRM-ID-00023][Error] The permissible values for the year range are 0001 through 9999 for elements ddms:start and ddms:end. Human Readable: ddms:start and ddms:end must be positive integers less than 10,000.

Code Description: This pattern uses abstract rules to consolidate logic. For attributes, we make sure that each date contained within $dateList has a year value within the range $minYear and $maxYear, inclusive.

Schematron Code:

```
<?ICEA pattern?><!-- Notices - distEditionBlockReplace-->
<sch:pattern id="IRM-ID-00023">




        <!-- Use abstract rule to handle required attributes -->
    <sch:rule context="ddms:start | ddms:end">
            <sch:let name="minYear" value="0001"/>
            <sch:let name="maxYear" value="9999"/>
            <sch:let name="dateList" value="(string(ddms:start),
string(ddms:end))"/>
            <sch:let name="errMsg" value="'         [IRM-ID-00023][Error]
The permissible values for the year range are 0001 through 9999          for
```

```
elements ddms:start and ddms:end.          Human Readable: ddms:start and
ddms:end must be positive integers less than 10,000.             '"/>
            <sch:extends rule="abs.dateListYearRangeRule"/>
         </sch:rule>
      </sch:pattern>
```

2.10 - .//Rules/dateTimeConstraints/IRM_ID_00024.sch

Rule Description: IRM-ID-00024 [IRM-ID-00024][Warning] For elements ddms:start and ddms:end and attributes @ddms:approvedOn, @ddms:dateProcessed, @ddms:receivedOn, @ddms:infoCutOff, @ddms:posted, @ddms:validTil, and @ddms:created, if the time designator (T) is specified, it is recommended that time zone be specified. Human Readable: For elements and attributes of date-time types, if the time designator (T) is specified, it is recommended that time zone be specified.

Code Description: The pattern applies to ddms:start and ddms:end elements, as well as any element that contains one or more attributes @ddms:approvedOn, @ddms:infoCutOff, @ddms:posted, and @ddms:created. It joins each of these attribute values, if present, into a larger space-separated string. It then breaks this string into tokens at each space character. If the value of the token contains the time zone designator (T), then it makes sure that the token value matches the regular expression for a timeZone, which is defined in the main schema file (IRM_XML.sch).

Schematron Code:

```
<?ICEA pattern?><!-- Notices - distEditionBlockReplace-->
<sch:pattern id="IRM-ID-00024">




         <!-- Abstract rule, which asserts that if the date $primaryDate
specifies the
        time designator (T), then the timezone is specified -->
    <sch:rule abstract="true" id="abs.rule00024">
            <sch:assert test="             every $date in $dateList
satisfies                 if($date castable as xs:dateTime and
contains(string($date),'T'))                 then matches(string($date),
$endsWithTimeZoneRegEx)                else true()"
flag="warning">
        </sch:assert>
         </sch:rule>

         <!-- Begin using abstract rule on required elements and attributes --
>
    <sch:rule context="ddms:start">
            <sch:let name="dateList" value="."/>
            <sch:extends rule="abs.rule00024"/>
         </sch:rule>

         <sch:rule context="ddms:end">
```

```
            <sch:let name="dateList" value="."/>
            <sch:extends rule="abs.rule00024"/>
         </sch:rule>

         <sch:rule context="*[@ddms:approvedOn |
@ddms:dateProcessed |                          @ddms:receivedOn
|                         @ddms:posted |
@ddms:created |                           @ddms:infoCutOff
|                          @ddms:validTil]">
            <sch:let name="dateList" value="
(@ddms:approvedOn,              @ddms:dateProcessed,
@ddms:receivedOn,             @ddms:posted,
@ddms:created,              @ddms:infoCutOff,
@ddms:validTil)              "/>
            <sch:extends rule="abs.rule00024"/>
         </sch:rule>
      </sch:pattern>
```

2.11 - .//Rules/ddmsConstraints/IRM_ID_00011.sch

Rule Description: IRM-ID-00011 [IRM-ID-00011][Error] If an element ddms:metacardInfo/ddms:identifier does not exist with the attribute @ddms:qualifier value of [IC-ID]. Human Readable: Every metacard must have an ID identified as an [IC-ID].

Code Description: This rule checks that for each qualifier attribute in path ddms:metacardInfo/ddms:identifier/@ddms:qualifier, that its normalized value with leading- and trailing-spaces removed is equal to 'IC-ID'. If there is at least one attribute with the given qualifier attribute, the rule is satisfied.

Schematron Code:

```
<?ICEA pattern?><!-- Notices - distEditionBlockReplace-->
<sch:pattern id="IRM-ID-00011">




         <sch:rule context="ddms:metacardInfo">
            <sch:assert test="                 some $qualifier in
ddms:identifier/@ddms:qualifier satisfies                    normalize-
space(string($qualifier)) = 'IC-ID'             " flag="error">
            [IRM-ID-00011][Error] Every metacard must have an ID identified
as an [IC-ID].
         </sch:assert>
          </sch:rule>
      </sch:pattern>
```

2.12 - .//Rules/ddmsConstraints/IRM_ID_00012.sch

Rule Description: IRM-ID-00012 [IRM-ID-00012][Error] More than 1 element ddms:metacardInfo/ddms:identifier exists with the attribute @ddms:qualifier having a value of [IC-ID]. Human Readable: Every metacard must have 1 and only 1 ID identified as an [IC-ID].

Code Description: This rule checks that for each qualifier attribute in path ddms:metacardInfo/ddms:identifier/@ddms:qualifier, that its normalized value with leading- and trailing-spaces removed is equal to 'IC-ID'. If there is exactly one attribute with the given qualifier attribute, the rule is satisfied.

Schematron Code:

```
<?ICEA pattern?><!-- Notices - distEditionBlockReplace-->
<sch:pattern id="IRM-ID-00012">




        <sch:rule context="ddms:metacardInfo">
           <sch:assert test="                count(                    for
$qualifier in ddms:identifier/@ddms:qualifier return
if(normalize-space(string($qualifier)) = 'IC-ID')
then 1                          else null                ) = 1             "
flag="error">
            [IRM-ID-00012][Error] Every metacard must have 1 and only 1 ID
identified as an [IC-ID].
        </sch:assert>
         </sch:rule>
      </sch:pattern>
```

2.13 - .//Rules/ddmsConstraints/IRM_ID_00013.sch

Rule Description: IRM-ID-00013 [IRM-ID-00013][Error] If an element ddms:resource/ddms:identifier does not exist with the attribute @ddms:qualifier value of [IC-ID]. Human Readable: Every Resource must have an ID identified as an [IC-ID].

Code Description: This rule checks that for each qualifier attribute in path ddms:resource/ddms:identifier/@ddms:qualifier, that its normalized value with leading- and trailing-spaces removed is equal to 'IC-ID'. If there is at least one attribute with the given qualifier attribute, the rule is satisfied.

Schematron Code:

```
<?ICEA pattern?><!-- Notices - distEditionBlockReplace-->
<sch:pattern id="IRM-ID-00013">
```

```
        <sch:rule context="ddms:resource">
           <sch:assert test="            some $qualifier in ddms:identifier/
@ddms:qualifier satisfies                normalize-space(string($qualifier))
= 'IC-ID'             " flag="error">
           [IRM-ID-00013][Error] Every Resource must have an ID identified
as an [IC-ID].
        </sch:assert>
         </sch:rule>
      </sch:pattern>
```

2.14 - .//Rules/ddmsConstraints/IRM_ID_00014.sch

Rule Description: IRM-ID-00014 [IRM-ID-00014][Error] If more than 1 element ddms:resource/ddms:identifier exists with the attribute @ddms:qualifier value of [IC-ID]. Human Readable: Every Resource must have 1 and only 1 ID identified as an [IC-ID].

Code Description: This rule checks that for each qualifier attribute in path ddms:resource/ddms:identifier/@ddms:qualifier, that its normalized value with leading- and trailing-spaces removed is equal to 'IC-ID'. If there is exactly one attribute with the given qualifier attribute, the rule is satisfied.

Schematron Code:

```
<?ICEA pattern?><!-- Notices - distEditionBlockReplace-->
<sch:pattern id="IRM-ID-00014">




        <sch:rule context="ddms:resource">
           <sch:assert test="    count(        for $qualifier in
ddms:identifier/@ddms:qualifier return           if (normalize-
space(string($qualifier)) = 'IC-ID')           then 1
else null             ) = 1             " flag="error">
           [IRM-ID-00014][Error] Every Resource must have 1 and only 1 ID
identified as an [IC-ID].
        </sch:assert>
         </sch:rule>
      </sch:pattern>
```

2.15 - .//Rules/ddmsConstraints/IRM_ID_00029.sch

Rule Description: IRM-ID-00029 [IRM-ID-00029][Error] If element ddms:geospatialCoverage has attribute @ddms:precedence with a value of [Secondary], there must be at least one sibling element ddms:geospatialCoverage for which attribute @ddms:precedence has a value of [Primary]. Human Readable: If a secondary country code is provided, there must also be a primary country code.

Code Description: If there is an element geospatialCoverage with attribute precedence specified with a value of [Secondary], then we make sure that there is at least one sibling geospatialCoverage element with attribute precedence specified with a value of [Primary].

Schematron Code:

```
<?ICEA pattern?><!-- Notices - distEditionBlockReplace-->
<sch:pattern id="IRM-ID-00029">




        <sch:rule
context="ddms:geospatialCoverage[@ddms:precedence='Secondary']">
            <sch:assert test="../
ddms:geospatialCoverage[@ddms:precedence='Primary']" flag="error">
            [IRM-ID-00029][Error]
            If element ddms:geospatialCoverage has attribute @ddms:precedence
with a value of [Secondary],
            there must be at least one sibling element
ddms:geospatialCoverage for which attribute
            @ddms:precedence has a value of [Primary].
        </sch:assert>
         </sch:rule>
     </sch:pattern>
```

2.16 - .//Rules/ddmsConstraints/IRM_ID_00030.sch

Rule Description: IRM-ID-00030 [IRM-ID-00030][Error] If attribute @ddms:order is specified with integer value N, there must exist other @ddms:order attributes with values 1 to N-1 with no duplicates. Human Readable: The values of attribute @ddms:order must be numbered sequentially with no duplicates, beginning at 1.

Code Description: A list, named $orderList, is created containing the value of each order attribute within the document after normalizing to remove extra white-space. If the total number of items in $orderList does not equal the number of distinct values in $orderList, then a duplicate exists and we return false. Otherwise, we make sure that each number from 1 to N, where N is the number of items in $orderList, is contained within $orderList. If each number is contained, then we return true. Otherwise, false.

Schematron Code:

```
<?ICEA pattern?><!-- Notices - distEditionBlockReplace-->
<sch:pattern id="IRM-ID-00030">




            <sch:rule context="ddms:resource[//@ddms:order]">
           <sch:let name="orderList" value="tokenize(string-join(//
@ddms:order/normalize-space(), ' '), ' ')"/>
```

```
            <sch:assert test="count(distinct-values($orderList)) =
count($orderList)                    and (every $index in 1 to
count($orderList)                              satisfies index-of($orderList,
xs:string($index)))             " flag="error">
            [IRM-ID-00030][Error]
            If attribute @ddms:order is specified with integer value N, there
must exist
            other @ddms:order attributes with values 1 to N-1 with no
duplicates.

            Human Readable: The values of attribute @ddms:order must be
numbered
            sequentially with no duplicates, beginning at 1.
        </sch:assert>
         </sch:rule>
      </sch:pattern>
```

2.17 - .//Rules/ddmsConstraints/IRM_ID_00032.sch

Rule Description: IRM-ID-00032 [IRM-ID-00032][] Removed in V6.

Code Description:

Schematron Code:

```
<?ICEA pattern?><!-- Notices - distEditionBlockReplace-->
<sch:pattern id="IRM-ID-00032">


      </sch:pattern>
```

2.18 - .//Rules/ddmsConstraints/IRM_ID_00035.sch

Rule Description: IRM-ID-00035 [IRM-ID-00035][Error] For element ddms:resource, child element ddms:language must be specified at least once. Human Readable: Every Resource must specify its language.

Code Description: Make sure that element ddms:resource has at least one child element ddms:language.

Schematron Code:

```
<!-- Notices - distEditionBlockReplace-->
<sch:pattern id="IRM-ID-00035">




            <sch:rule context="ddms:resource">
                  <sch:assert id="IRM-00035" test="count(ddms:language) &gt;
```

```
0" flag="error">
            [IRM-ID-00035][Error] For element ddms:resource, child element
ddms:language must be specified at least once.
        </sch:assert>
            </sch:rule>
      </sch:pattern>
```

2.19 - .//Rules/ddmsConstraints/IRM_ID_00037.sch

Rule Description: IRM-ID-00037 [IRM-ID-00037][Error] If the attribute @ddms:POCType is specified with a value of [ORCON], then there must exist a descendant element ddms:phone with a value specified. Human Readable: Elements denoted as POCs for an ORIGINATOR CONTROLLED memo must specify a phone number for that POC.

Code Description: For any node that has the attribute @ddms:POCType with a value of 'ORCON', the code will determine if there exists a descendant element ddms:phone with a non-null, non-whitespace text value. If the value exists, the rule will return true; otherwise, false will be returned.

Schematron Code:

```
<?ICEA pattern?><!-- Notices - distEditionBlockReplace-->
<sch:pattern id="IRM-ID-00037">




               <sch:rule context="*[tokenize(string(@ddms:POCType),'
')='ORCON']">
            <sch:assert test="normalize-space(descendant::ddms:phone/text())"
flag="error">
            [IRM-ID-00037][Error]
            If the attribute @ddms:POCType is specified with a value of
[ORCON], then there
            must exist a descendant element ddms:phone with a value specified.

            Human Readable: Elements denoted as POCs for an ORIGINATOR
CONTROLLED memo
            must specify a phone number for that POC.
        </sch:assert>
         </sch:rule>


      </sch:pattern>
```

2.20 - .//Rules/ddmsConstraints/IRM_ID_00038.sch

Rule Description: IRM-ID-00038 [IRM-ID-00038][Error] At least one ddms:resource/publisher must be specified per ddms card. Human Readable: At least one publishing agency must be specified in each metadata card.

Code Description: Make sure that element ddms:resource has at least one child element ddms:publisher.

Schematron Code:

```
<?ICEA pattern?><!-- Notices - distEditionBlockReplace-->
<sch:pattern id="IRM-ID-00038">




            <sch:rule context="ddms:resource">
                  <sch:assert id="IRM-00038" test="count(ddms:publisher) &gt;
0" flag="error">
            [IRM-ID-00038][Error] At least one publishing agency must be
specified in each metadata card.
        </sch:assert>
            </sch:rule>
      </sch:pattern>
```

2.21 - .//Rules/ddmsConstraints/IRM_ID_00039.sch

Rule Description: IRM-ID-00039 [IRM-ID-00039][Error] At least one ddms:subjectCoverage/ ddms:productionMetric must be specified per ddms card. Human Readable: At least one production metric must be specified in each metadata card.

Code Description: Make sure that element ddms:resource has at least one child element ddms:publisher.

Schematron Code:

```
<?ICEA pattern?><!-- Notices - distEditionBlockReplace-->
<sch:pattern id="IRM-ID-00039">




            <sch:rule context="ddms:subjectCoverage">
                  <sch:assert id="IRM-00039"
test="count(ddms:productionMetric) &gt; 0" flag="error">
            [IRM-ID-00039][Error] At least one production metric must be
specified in each metadata card.
        </sch:assert>
            </sch:rule>
      </sch:pattern>
```

2.22 - .//Rules/ddmsConstraints/IRM_ID_00055.sch

Rule Description: IRM-ID-00055 [IRM-ID-00055][Error] If ddms:geospatialCoverage/@order is specified then there must be one and only one of ddms:geospatialIdentifier/ddms:countryCode

or ddms:geospatialIdentifier/ddms:subDivisionCode. Human Readable: A single order value must be applied to one country code or one subdivision code but not to both.

Code Description: Make sure that there is only one ddms:countryCode or order ddms:subDivisionCode when ddms:geospatialCoverage uses the order attribute.

Schematron Code:

```
<!-- Notices - distEditionBlockReplace-->
<sch:pattern id="IRM-ID-00055">




           <sch:rule context="ddms:geospatialCoverage[@ddms:order]">
                 <sch:assert id="IRM-00055"
test="count(ddms:geographicIdentifier/ddms:countryCode) +
count(ddms:geographicIdentifier/ddms:subDivisionCode) = 1" flag="error">
           [IRM-ID-00055][Error] If ddms:geospatialCoverage/@order is
specified then
           there must be one and only one of ddms:geospatialIdentifier/
ddms:countryCode or
           ddms:geospatialIdentifier/ddms:subDivisionCode. Human Readable: A
single order value
           must be applied to one country code or one subdivision code
       </sch:assert>
           </sch:rule>
     </sch:pattern>
```

2.23 - .//Rules/globalConstraints/IRM_ID_00002.sch

Rule Description: IRM-ID-00002 [IRM-ID-00002][Error] For every attribute in the namespace [urn:us:gov:ic:irm] or [urn:us:mil:ces:metadata:ddms:4], a non-whitespace value must be specified.

Code Description: For each element which specifies an attribute in the IRM or DDMS namespace, we make sure that each attribute in the IRM or DDMS namespace specifies a non-whitespace value.

Schematron Code:

```
<?ICEA pattern?><!-- Notices - distEditionBlockReplace-->
<sch:pattern id="IRM-ID-00002">




        <sch:rule context="*[@*[namespace-uri()=('urn:us:gov:ic:irm',
'urn:us:mil:ces:metadata:ddms:4')]]">
           <sch:assert test="            every $attribute in @*[namespace-
uri()=('urn:us:gov:ic:irm', 'urn:us:mil:ces:metadata:ddms:4')]
```

```
satisfies                 normalize-space(string($attribute))" flag="error">
            [IRM-ID-00002][Error] For every attribute in the namespace
            [urn:us:gov:ic:irm] or [urn:us:mil:ces:metadata:ddms:4], a non-
whitespace
            value must be specified.
        </sch:assert>
         </sch:rule>
      </sch:pattern>
```

2.24 - .//Rules/ismConstraints/IRM_ID_00025.sch

Rule Description: IRM-ID-00025 [IRM-ID-00025][Error] The attribute @ism:excludeFromRollup must not be specified for any element in the namespace [urn:us:mil:ces:metadata:ddms:4] except security. Human Readable: The only DDMS 4.0 element whose ISM attributes are allowed to be excluded from roll-up to the Resource element is ddms:security.

Code Description: For any element in the ddms namespace containing the ism:excludeFromRollup attribute, the rule will be satisfied if the name of the element is security.

Schematron Code:

```
<?ICEA pattern?><!-- Notices - distEditionBlockReplace-->
<sch:pattern id="IRM-ID-00025">




         <sch:rule context="*[@ism:excludeFromRollup and namespace-
uri()='urn:us:mil:ces:metadata:ddms:4']">
            <sch:assert test="local-name() = 'security'" flag="error">
            [IRM-ID-00025][Error] The attribute @ism:excludeFromRollup must
not be specified for any element
            in the namespace [urn:us:mil:ces:metadata:ddms:4] except security.
        </sch:assert>
         </sch:rule>
      </sch:pattern>
```

2.25 - .//Rules/ismConstraints/IRM_ID_00026.sch

Rule Description: IRM-ID-00026 [IRM-ID-00026][] Removed in V4.

Code Description:

Schematron Code:

```
<?ICEA pattern?><!-- Notices - distEditionBlockReplace-->
<sch:pattern id="IRM-ID-00026">
```

```
      </sch:pattern>
```

2.26 - .//Rules/ismConstraints/IRM_ID_00040.sch

Rule Description: IRM-ID-00040 [IRM-ID-00040][Error] If element ddms:type is specified with a qualifier of [urn:us:gov:ic:cvenum:irm:activity], then attribute ism:classification must also be specified.

Code Description: For each ddms:type element which specifies attribute ddms:qualifier with a value of [urn:us:gov:ic:cvenum:irm:activity], we make sure that attribute ism:classification is specified.

Schematron Code:

```
<?ICEA pattern?><!-- Notices - distEditionBlockReplace-->
<sch:pattern is-a="IsmEnforcement" id="IRM-ID-00040">



         <sch:param name="ruleText" value="'[IRM-ID-00040][Error]         If
element ddms:type is specified with a qualifier of
[urn:us:gov:ic:cvenum:irm:activity], then         attribute
ism:classification must also be specified.'"/>
         <sch:param name="codeDesc" value="'         For each ddms:type
element which specifies attribute ddms:qualifier with         a value of
[urn:us:gov:ic:cvenum:irm:activity],         we make sure that attribute
ism:classification is specified.         '"/>

         <sch:param name="context"
value="ddms:type[@ddms:qualifier='urn:us:gov:ic:cvenum:irm:activity']"/>
         <sch:param name="errMsg" value="'         [IRM-ID-00040]
[Error]         If element ddms:type is specified with a qualifier
of          [urn:us:gov:ic:cvenum:irm:activity], then         attribute
ism:classification must also be specified.         '"/>
      </sch:pattern>
```

2.27 - .//Rules/ismConstraints/IRM_ID_00041.sch

Rule Description: IRM-ID-00041 IRM-ID-00041][Error] If element ddms:type is specified with a qualifier of [urn:us:gov:ic:irm:intel:disciplines], then attribute ism:classification must also be specified.

Code Description: For each ddms:type element which specifies attribute ddms:qualifier with a value of [urn:us:gov:ic:irm:intel:disciplines], we make sure that attribtue ism:classification is specified.

Schematron Code:

```
<?ICEA pattern?><!-- Notices - distEditionBlockReplace-->
<sch:pattern is-a="IsmEnforcement" id="IRM-ID-00041">
```

```
        <sch:param name="ruleText" value="'[IRM-ID-00041][Error]        If
element ddms:type is specified with a qualifier of
[urn:us:gov:ic:irm:intel:disciplines], then        attribute
ism:classification must also be specified.'"/>
        <sch:param name="codeDesc" value="'        For each ddms:type
element which specifies attribute ddms:qualifier with        a value of
[urn:us:gov:ic:irm:intel:disciplines],        we make sure that attribtue
ism:classification is specified.'"/>

        <sch:param name="context"
value="ddms:type[@ddms:qualifier='urn:us:gov:ic:irm:intel:disciplines']"/>
        <sch:param name="errMsg" value="'        [IRM-ID-00041]
[Error]        If element ddms:type is specified with a qualifier
of        [urn:us:gov:ic:irm:intel:disciplines], then        attribute
ism:classification must also be specified.        '"/>
      </sch:pattern>
```

2.28 - .//Rules/ismConstraints/IRM_ID_00042.sch

Rule Description: IRM-ID-00042 [IRM-ID-00042][Error] If element ddms:type is specified with a qualifier of [urn:us:gov:ic:cvenum:irm:intel:subdisciplines], then attribute ism:classification must also be specified.

Code Description: For each ddms:type element which specifies attribute ddms:qualifier with a value of [urn:us:gov:ic:cvenum:irm:intel:subdisciplines], we make sure that attribtue ism:classification is specified.

Schematron Code:

```
<?ICEA pattern?><!-- Notices - distEditionBlockReplace-->
<sch:pattern is-a="IsmEnforcement" id="IRM-ID-00042">



        <sch:param name="ruleText" value="'[IRM-ID-00042][Error]        If
element ddms:type is specified with a qualifier of
[urn:us:gov:ic:cvenum:irm:intel:subdisciplines], then        attribute
ism:classification must also be specified.'"/>
        <sch:param name="codeDesc" value="'        For each ddms:type
element which specifies attribute ddms:qualifier with        a value of
[urn:us:gov:ic:cvenum:irm:intel:subdisciplines],        we make sure that
attribtue ism:classification is specified.'"/>


        <sch:param name="context"
value="ddms:type[@ddms:qualifier='urn:us:gov:ic:cvenum:irm:intel:subdiscipline
s']"/>
        <sch:param name="errMsg" value="'        [IRM-ID-00042]
[Error]        If element ddms:type is specified with a qualifier
```

```
of          [urn:us:gov:ic:cvenum:irm:intel:subdisciplines], then
attribute ism:classification must also be specified.          '"/>
      </sch:pattern>
```

2.29 - .//Rules/ismConstraints/IRM_ID_00043.sch

Rule Description: IRM-ID-00043 If element ddms:type is specified with a qualifier of [urn:us:gov:ic:cvenum:irm:intel:subdisciplinetechnique], then attribute ism:classification must also be specified.'

Code Description: For each ddms:type element which specifies attribute ddms:qualifier with a value of [urn:us:gov:ic:cvenum:irm:intel:subdisciplinetechnique], we make sure that attribtue ism:classification is specified.

Schematron Code:

```
<?ICEA pattern?><!-- Notices - distEditionBlockReplace-->
<sch:pattern is-a="IsmEnforcement" id="IRM-ID-00043">




         <sch:param name="ruleText" value="'[IRM-ID-00043][Error]         If
element ddms:type is specified with a qualifier of
[urn:us:gov:ic:cvenum:irm:intel:subdisciplinetechnique], then
attribute ism:classification must also be specified.'"/>

         <sch:param name="codeDesc" value="'         For each ddms:type
element which specifies attribute ddms:qualifier with         a value of
[urn:us:gov:ic:cvenum:irm:intel:subdisciplinetechnique],         we make sure
that attribtue ism:classification is specified.'"/>

         <sch:param name="context"
value="ddms:type[@ddms:qualifier='urn:us:gov:ic:cvenum:irm:intel:subdiscipline
technique']"/>
         <sch:param name="errMsg" value="'         [IRM-ID-00043]
[Error]         If element ddms:type is specified with a qualifier
of          [urn:us:gov:ic:cvenum:irm:intel:subdisciplinetechnique],
then          attribute ism:classification must also be specified.         '"/>
      </sch:pattern>
```

2.30 - .//Rules/ismConstraints/IRM_ID_00044.sch

Rule Description: IRM-ID-00044 [IRM-ID-00044][Error] If element ddms:type is specified with a qualifier of [urn:us:gov:ic:irm:productline] then attribute ism:classification must also be specified.

Code Description: For each ddms:type element which specifies attribute ddms:qualifier with a value of [urn:us:gov:ic:irm:productline], we make sure that attribtue ism:classification is specified.'

Schematron Code:

```
<?ICEA pattern?><!-- Notices - distEditionBlockReplace-->
<sch:pattern is-a="IsmEnforcement" id="IRM-ID-00044">



        <sch:param name="ruleText" value="'[IRM-ID-00044][Error]        If
element ddms:type is specified with a qualifier of
[urn:us:gov:ic:irm:productline] then attribute ism:classification
must         also be specified.'"/>
        <sch:param name="codeDesc" value="'         For each ddms:type
element which specifies attribute ddms:qualifier with         a value of
[urn:us:gov:ic:irm:productline], we make sure that attribtue
ism:classification is specified.'"/>


        <sch:param name="context"
value="ddms:type[@ddms:qualifier='urn:us:gov:ic:irm:productline']"/>
        <sch:param name="errMsg" value="'         [IRM-ID-00044]
[Error]        If element ddms:type is specified with a qualifier of
[urn:us:gov:ic:irm:productline] then attribute ism:classification
must         also be specified.         '"/>
      </sch:pattern>
```

2.31 - .//Rules/ismConstraints/IRM_ID_00045.sch

Rule Description: IRM-ID-00045 [IRM-ID-00045][Error] Element ddms:geospatialCoverage must have ISM classification markings.

Code Description: For each ddms:geospatialCoverage element, we make sure that attribute ism:classification is specified.

Schematron Code:

```
<?ICEA pattern?><!-- Notices - distEditionBlockReplace-->
<sch:pattern id="IRM-ID-00045">



        <sch:rule context="ddms:geospatialCoverage">
           <sch:assert test="@ism:classification" flag="error">
           [IRM-ID-00045][Error] Element ddms:geospatialCoverage must have
ISM
           classification markings.
        </sch:assert>
         </sch:rule>
      </sch:pattern>
```

2.32 - .//Rules/valueEnumerationConstraints/ IRM_ID_00001.sch

Rule Description: IRM-ID-00001 [IRM-ID-00001][Error] If element ddms:countryCode has attribute @ddms:qualifier specified as [urn:us:gov:ic:cvenum:irm:coverage:fips:digraph] then the value of attribute @ddms:value must be in CVEnumIRMCoverageFIPSDigraph.xml. Human Readable: FIPS CountryCodes must appear in the FIPS CVE.

Code Description: This rule uses an abstract pattern to consolidate logic. It checks that the value in parameter $searchTerm is contained in the parameter $list. The parameter $searchTerm is relative in scope to the parameter $context. The value for the parameter $list is a variable defined in the main document (IRM_XML.sch), which reads values from a specific CVE file.

Schematron Code:

```
<?ICEA pattern?><!-- Notices - distEditionBlockReplace-->
<sch:pattern id="IRM-ID-00001" is-a="ValidateValueExistenceInList">




            <sch:param name="ruleText" value="'   [IRM-ID-00001][Error] If
element ddms:countryCode has attribute @ddms:qualifier specified as
[urn:us:gov:ic:cvenum:irm:coverage:fips:digraph] then the value of attribute
@ddms:value must   be in CVEnumIRMCoverageFIPSDigraph.xml.      Human
Readable: FIPS CountryCodes must in the the FIPS CVE.   '"/>

            <sch:param name="codeDesc" value="'   This rule uses an abstract
pattern to consolidate logic. It checks that the   value in parameter
$searchTerm is contained in the parameter $list. The parameter   $searchTerm
is relative in scope to the parameter $context. The value for the
parameter    $list is a variable defined in the main document (IRM_XML.sch),
which reads    values from a specific CVE file.   '"/>

            <sch:param name="context"
value="ddms:countryCode[@ddms:qualifier='urn:us:gov:ic:cvenum:irm:coverage:fip
s:digraph']"/>
            <sch:param name="searchTerm" value="@ddms:value"/>
            <sch:param name="list" value="$coverageFipsDigraphList"/>
            <sch:param name="errMsg" value="'   [IRM-ID-00001][Error] If
element ddms:countryCode has attribute @ddms:qualifier specified as
[urn:us:gov:ic:cvenum:irm:coverage:fips:digraph] then the value of attribute
@ddms:value must   be in CVEnumIRMCoverageFIPSDigraph.xml.      Human
Readable: FIPS CountryCodes must in the the FIPS CVE.   '"/>
      </sch:pattern>
```

2.33 - .//Rules/valueEnumerationConstraints/ IRM_ID_00003.sch

Rule Description: IRM-ID-00003 [IRM-ID-00003][Error] If element ddms:countryCode has attribute @ddms:qualifier specified as [urn:us:gov:ic:cvenum:irm:coverage:iso3166:trigraph] then the value of attribute @ddms:value must be in CVEnumIRMCoverageISO3166trigraph.xml. Human Readable: ISO trigraph CountryCodes must appear in the ISO trigraph CVE

Code Description: This rule uses an abstract pattern to consolidate logic. It checks that the value in parameter $searchTerm is contained in the parameter $list. The parameter $searchTerm is relative in scope to the parameter $context. The value for the parameter $list is a variable defined in the main document (IRM_XML.sch), which reads values from a specific CVE file.

Schematron Code:

```
<?ICEA pattern?><!-- Notices - distEditionBlockReplace-->
<sch:pattern id="IRM-ID-00003" is-a="ValidateValueExistenceInList">




            <sch:param name="ruleText" value="'   [IRM-ID-00003][Error] If
element ddms:countryCode has attribute @ddms:qualifier specified as
[urn:us:gov:ic:cvenum:irm:coverage:iso3166:trigraph] then the value of
attribute @ddms:value must   be in CVEnumIRMCoverageISO3166trigraph.xml.
Human Readable: ISO trigraph CountryCodes must in the the ISO trigraph CVE.
'"/>

            <sch:param name="codeDesc" value="'   This rule uses an abstract
pattern to consolidate logic. It checks that the   value in parameter
$searchTerm is contained in the parameter $list. The parameter   $searchTerm
is relative in scope to the parameter $context. The value for the
parameter    $list is a variable defined in the main document (IRM_XML.sch),
which reads    values from a specific CVE file.   '"/>

            <sch:param name="context"
value="ddms:countryCode[@ddms:qualifier='urn:us:gov:ic:cvenum:irm:coverage:iso
3166:trigraph']"/>
            <sch:param name="searchTerm" value="@ddms:value"/>
            <sch:param name="list" value="$coverageIso3166TrigraphList"/>
            <sch:param name="errMsg" value="'   [IRM-ID-00003][Error] If
element ddms:countryCode has attribute @ddms:qualifier specified as
[urn:us:gov:ic:cvenum:irm:coverage:iso3166:trigraph] then the value of
attribute @ddms:value must   be in CVEnumIRMCoverageISO3166trigraph.xml.
Human Readable: ISO trigraph CountryCodes must in the the ISO trigraph CVE.
'"/>
      </sch:pattern>
```

2.34 - .//Rules/valueEnumerationConstraints/ IRM_ID_00004.sch

Rule Description: IRM-ID-00004 [IRM-ID-00004][] Removed in V4.

Code Description:

Schematron Code:

```
<?ICEA pattern?><!-- Notices - distEditionBlockReplace-->
<sch:pattern id="IRM-ID-00004">


     </sch:pattern>
```

2.35 - .//Rules/valueEnumerationConstraints/ IRM_ID_00005.sch

Rule Description: IRM-ID-00005 [IRM-ID-00005][Error] If element ddms:language has the attribute @ddms:qualifier value of [urn:us:gov:ic:cvenum:irm:iso639:digraph] then the value of attribute @ddms:value must be in CVEnumIRMISO639Digraph.xml and no country code portion may be specified in the @ddms:language element value. Human Readable: ISO 639 digraph language codes must be in the ISO 639 digraph CVE.

Code Description: This rule uses an abstract pattern to consolidate logic. It checks that the value in parameter $searchTerm is contained in the parameter $list. The parameter $searchTerm is relative in scope to the parameter $context. The value for the parameter $list is a variable defined in the main document (IRM_XML.sch), which reads values from a specific CVE file. This rule can directly check if the value of element Language is in the appropriate list because if a country code portion is specified in the element ddms:language's value, then the value of element ddms:language will not be found in the appropriate list and the assertion will fail as expected.

Schematron Code:

```
<?ICEA pattern?><!-- Notices - distEditionBlockReplace-->
<sch:pattern id="IRM-ID-00005" is-a="ValidateValueExistenceInList">




           <sch:param name="ruleText" value="'   [IRM-ID-00005][Error] If
element ddms:language has the attribute @ddms:qualifier    value of
[urn:us:gov:ic:cvenum:irm:iso639:digraph] then the value of attribute
@ddms:value must be in CVEnumIRMISO639Digraph.xml and no country code portion
may    be specified in the ddms:language element value.      Human Readable:
ISO 639 digraph language codes must be in the ISO 639 digraph CVE.    '"/>

           <sch:param name="codeDesc" value="'   This rule uses an abstract
```

```
pattern to consolidate logic. It checks that the   value in parameter
$searchTerm is contained in the parameter $list. The parameter   $searchTerm
is relative in scope to the parameter $context. The value for the
parameter    $list is a variable defined in the main document (IRM_XML.sch),
which reads    values from a specific CVE file.   '"/>

            <sch:param name="context"
value="ddms:language[@ddms:qualifier='urn:us:gov:ic:cvenum:irm:iso639:digraph'
]"/>
            <sch:param name="searchTerm" value="@ddms:value"/>
            <sch:param name="list" value="$iso639DigraphList"/>
            <sch:param name="errMsg" value="'   [IRM-ID-00005][Error] If
element ddms:language has the attribute @ddms:qualifier    value of
[urn:us:gov:ic:cvenum:irm:iso639:digraph] then the value of attribute
@ddms:value must be in CVEnumIRMISO639Digraph.xml and no country code portion
may    be specified in the ddms:language element value.      Human Readable:
ISO 639 digraph language codes must be in the ISO 639 digraph CVE.    '"/>
      </sch:pattern>
```

2.36 - .//Rules/valueEnumerationConstraints/ IRM_ID_00006.sch

Rule Description: IRM-ID-00006 [IRM-ID-00006][Error] If element ddms:language has the attribute @ddms:qualifier value of [urn:us:gov:ic:cvenum:irm:iso639-2:trigraph] then the value of attribute @ddms:value must be in CVEnumIRMISO639-2Trigraph.xml and no country code portion may be specified in the ddms:value attribute value. Human Readable: ISO 639-2 trigraph language codes must be in the ISO 639-2 trigraph CVE.

Code Description: This rule uses an abstract pattern to consolidate logic. It checks that the value in parameter $searchTerm is contained in the parameter $list. The parameter $searchTerm is relative in scope to the parameter $context. The value for the parameter $list is a variable defined in the main document (IRM_XML.sch), which reads values from a specific CVE file. This rule can directly check if the value of element ddms:language is in the appropriate list because if a country code portion is specified in the element ddms:language's value, then the value of element ddms:language will not be found in the appropriate list and the assertion will fail as expected.

Schematron Code:

```
<?ICEA pattern?><!-- Notices - distEditionBlockReplace-->
<sch:pattern id="IRM-ID-00006" is-a="ValidateValueExistenceInList">




            <sch:param name="ruleText" value="'   [IRM-ID-00006][Error] If
element ddms:language has the attribute @ddms:qualifier    value of
[urn:us:gov:ic:cvenum:irm:iso639-2:trigraph] then the value of    attribute
@ddms:value must be in CVEnumIRMISO639-2Trigraph.xml and no country    code
portion may be specified in the ddms:value attribute value.      Human
```

```
Readable: ISO 639-2 trigraph language codes must be in the ISO 639-2
trigraph CVE.   '"/>

            <sch:param name="codeDesc" value="'   This rule uses an abstract
pattern to consolidate logic. It checks that the   value in parameter
$searchTerm is contained in the parameter $list. The parameter   $searchTerm
is relative in scope to the parameter $context. The value for the
parameter    $list is a variable defined in the main document (IRM_XML.sch),
which reads    values from a specific CVE file.   '"/>

            <sch:param name="context"
value="ddms:language[@ddms:qualifier='urn:us:gov:ic:cvenum:irm:iso639-2:trigra
ph']"/>
            <sch:param name="searchTerm" value="@ddms:value"/>
            <sch:param name="list" value="$iso639-2TrigraphList"/>
            <sch:param name="errMsg" value="'   [IRM-ID-00006][Error] If
element ddms:language has the attribute @ddms:qualifier    value of
[urn:us:gov:ic:cvenum:irm:iso639-2:trigraph] then the value of    attribute
@ddms:value must be in CVEnumIRMISO639-2Trigraph.xml and no country    code
portion may be specified in the ddms:value attribute value.      Human
Readable: ISO 639-2 trigraph language codes must be in the ISO 639-2
trigraph CVE.   '"/>
      </sch:pattern>
```

2.37 - .//Rules/valueEnumerationConstraints/ IRM_ID_00007.sch

Rule Description: IRM-ID-00007 [IRM-ID-00007][Error] If element ddms:language has the attribute @ddms:qualifier value of [urn:us:gov:ic:cvenum:irm:iso639-3:trigraph] then the value of attribute @ddms:value must be in CVEnumIRMISO639-3Trigraph.xml and no country code portion may be specified in the ddms:value attribute value. Human Readable: ISO 639-3 trigraph language codes must be in the ISO 639-3 trigraph CVE.

Code Description: This rule uses an abstract pattern to consolidate logic. It checks that the value in parameter $searchTerm is contained in the parameter $list. The parameter $searchTerm is relative in scope to the parameter $context. The value for the parameter $list is a variable defined in the main document (IRM_XML.sch), which reads values from a specific CVE file. This rule can directly check if the value of element Language is in the appropriate list because if a country code portion is specified in the element ddms:language's value, then the value of element ddms:language will not be found in the appropriate list and the assertion will fail as expected.

Schematron Code:

```
<?ICEA pattern?><!-- Notices - distEditionBlockReplace-->
<sch:pattern id="IRM-ID-00007" is-a="ValidateValueExistenceInList">
```

```
            <sch:param name="ruleText" value="'   [IRM-ID-00007][Error] If
element ddms:language has the attribute @ddms:qualifier    value of
[urn:us:gov:ic:cvenum:irm:iso639-3:trigraph] then the value of    attribute
@ddms:value must be in CVEnumIRMISO639-3Trigraph.xml and no country code
portion may be specified in the ddms:value attribute value.      Human
Readable: ISO 639-3 trigraph language codes must in the the ISO 639-3
trigraph CVE.   '"/>

            <sch:param name="codeDesc" value="'   This rule uses an abstract
pattern to consolidate logic. It checks that the   value in parameter
$searchTerm is contained in the parameter $list. The parameter   $searchTerm
is relative in scope to the parameter $context. The value for the
parameter    $list is a variable defined in the main document (IRM_XML.sch),
which reads    values from a specific CVE file.   '"/>

            <sch:param name="context"
value="ddms:language[@ddms:qualifier='urn:us:gov:ic:cvenum:irm:iso639-3:trigra
ph']"/>
            <sch:param name="searchTerm" value="@ddms:value"/>
            <sch:param name="list" value="$iso639-3TrigraphList"/>
            <sch:param name="errMsg" value="'   [IRM-ID-00007][Error] If
element ddms:language has the attribute @ddms:qualifier    value of
[urn:us:gov:ic:cvenum:irm:iso639-3:trigraph] then the value of    attribute
@ddms:value must be in CVEnumIRMISO639-3Trigraph.xml and no country code
portion may be specified in the ddms:value attribute value.      Human
Readable: ISO 639-3 trigraph language codes must in the the ISO 639-3
trigraph CVE.   '"/>
      </sch:pattern>
```

2.38 - .//Rules/valueEnumerationConstraints/ IRM_ID_00008.sch

Rule Description: IRM-ID-00008 [IRM-ID-00008][Error] If element ddms:language has the attribute @ddms:qualifier value of [RFC1766] then the language code portion of the @ddms:value attribute value must be in CVEnumIRMISO639Digraph.xml and the country code portion, if present, must be in CVEnumIRMCoverageISO3166Digraph.xml. Human Readable: RFC1766 language codes must comply with the RFC by using parts from ISO 639 Digraph and ISO 3166 Digraph.

Code Description: Finds ddms:language element and checks its qualifier attribute for a value of [RFC4646]. If this value is found it will ensure that the value of the element's ddms:value attribute exists in the CVEnumIRMISO639Digraph.xml enumeration file represented by the $iso639DigraphList variable and country code portions (denoted by '-' separation) must be in CVEnumIRMCoverageISO3166Digraph.xml enumeration file represented by the $coverageIso3166DigraphList variable.

Schematron Code:

```
<?ICEA pattern?><!-- Notices - distEditionBlockReplace-->
<sch:pattern id="IRM-ID-00008">
```

```
              <sch:rule context="ddms:language[@ddms:qualifier='RFC1766']">
        <!-- Tokenize the element Language value into a list -->
        <sch:let name="tokens" value="tokenize(@ddms:value,'-')"/>

            <!-- For convenience and readability, save the primary and
secondary subtags
             as defined in RFC 4646 -->
        <sch:let name="primarySubtag" value="$tokens[1]"/>
            <sch:let name="secondarySubtag" value="$tokens[2]"/>

            <sch:let name="badPrimaryValues" value="             if(index-
of($iso639DigraphList,$primarySubtag)&gt;0)                 then
null             else $primarySubtag"/>

            <sch:let name="badSecondaryValues" value="
if($secondarySubtag and                 index-of($coverageIso3166DigraphList,
$secondarySubtag)&gt;0)                  then null             else
$secondarySubtag"/>

            <sch:let name="badValues" value="string-
join(($badPrimaryValues,              $badSecondaryValues), ' ')"/>

            <!-- Check if primary subtag is valid -->
        <sch:let name="primarySubtagValid" value="
count($badPrimaryValues) = 0             "/>

            <!-- Check if secondary subtag is valid -->
        <sch:let name="secondarySubtagValid" value="
if(not($secondarySubtag)) then true() else
count($badSecondaryValues) = 0             "/>

            <sch:assert test="$primarySubtagValid and $secondarySubtagValid"
flag="error">
            [IRM-ID-00008][Error] If element ddms:language has the attribute
@ddms:qualifier
            value of [RFC1766] then the language code portion of the
@ddms:value attribute
            value must be in CVEnumIRMISO639Digraph.xml and the country code
portion, if
            present, must be in CVEnumIRMCoverageISO3166Digraph.xml.

            Human Readable: RFC1766 language codes must comply with the RFC
by using parts from ISO 639 Digraph
            and ISO 3166 Digraph. The following values were found but are not
in the CVEs:
            <sch:value-of select="for $each in tokenize($badValues, ' ')
return concat('[',$each,'] ')"/>
            </sch:assert>
         </sch:rule>
```

```
    </sch:pattern>
```

2.39 - .//Rules/valueEnumerationConstraints/ IRM_ID_00009.sch

Rule Description: IRM-ID-00009 [IRM-ID-00009][Error] If element ddms:language has the attribute @ddms:qualifier value of [RFC3066] then the language code portion of the @ddms:value attribute value must be in CVEnumIRMISO639Digraph.xml or CVEnumIRMISO639-2Trigraph.xml and the country code portion, if present, must be in CVEnumIRMCoverageISO3166Digraph.xml. Human Readable: RFC3066 language codes must comply with the RFC by using parts from ISO 639 Digraph, 639-2 Trigraph, and ISO 3166 Digraph.

Code Description: Finds ddms:language elements and checks its qualifier attribute for a value of [RFC3066]. If this value is found it will ensure that the value of the element's ddms:value attribute exists in the CVEnumIRMISO639Digraph.xml or CVEnumIRMISO639-2Trigraph.xml enumeration files represented by the $iso639DigraphList or $iso639-2TrigraphList variables. Country code portions (denoted by '-' separation) must be in the CVEnumIRMCoverageISO3166Digraph.xml enumeration file represented by the $coverageIso3166DigraphList variable.

Schematron Code:

```
<?ICEA pattern?><!-- Notices - distEditionBlockReplace-->
<sch:pattern id="IRM-ID-00009">




              <sch:rule context="ddms:language[@ddms:qualifier='RFC3066']">
       <!-- Tokenize the element Language value into a list -->
       <sch:let name="tokens" value="tokenize(@ddms:value,'-')"/>

           <!-- For convenience and readability, save the primary and
secondary subtags
            as defined in RFC 3066 -->
       <sch:let name="primarySubtag" value="$tokens[1]"/>
           <sch:let name="secondarySubtag" value="$tokens[2]"/>

           <sch:let name="badPrimaryValues" value="             if(index-
of($iso639-2TrigraphList,$primarySubtag)&gt;0 or               index-
of($iso639DigraphList,$primarySubtag)&gt;0)                   then
null             else $primarySubtag"/>

           <sch:let name="badSecondaryValues" value="
if($secondarySubtag and                index-of($coverageIso3166DigraphList,
$secondarySubtag)&gt;0)                 then null             else
$secondarySubtag"/>

           <sch:let name="badValues" value="string-
```

```
join(($badPrimaryValues,             $badSecondaryValues), ' ')"/>

            <!-- Check if primary subtag is valid -->
        <sch:let name="primarySubtagValid" value="
count($badPrimaryValues) = 0             "/>

            <!-- Check if secondary subtag is valid -->
        <sch:let name="secondarySubtagValid" value="
if(not($secondarySubtag)) then true() else
count($badSecondaryValues) = 0             "/>

            <sch:assert test="$primarySubtagValid and $secondarySubtagValid"
flag="error">
            [IRM-ID-00009][Error] If element ddms:language has the attribute
@ddms:qualifier
            value of [RFC3066] then the language code portion of the
@ddms:value attribute
            value must be in CVEnumIRMISO639Digraph.xml or
CVEnumIRMISO639-2Trigraph.xml
            and the country code portion, if present, must be in
CVEnumIRMCoverageISO3166Digraph.xml.

            Human Readable: RFC3066 language codes must comply with the RFC
by using parts from
            ISO 639 Digraph or ISO 639-2 Trigraph and ISO 3166 Digraph. The
following values were found but
            are not in the CVEs:
            <sch:value-of select="for $each in tokenize($badValues, ' ')
return concat('[',$each,'] ')"/>
            </sch:assert>
          </sch:rule>
      </sch:pattern>
```

2.40 - .//Rules/valueEnumerationConstraints/IRM_ID_00010.sch

Rule Description: IRM-ID-00010 [IRM-ID-00010][Error] If element ddms:language has the attribute @ddms:qualifier value of [RFC4646] then the language code portion of the @ddms:value attribute value must be in CVEnumIRMISO639Digraph.xml or CVEnumIRMISO639-2Trigraph.xml and the country code portion, if present, must be in CVEnumIRMCoverageISO3166Digraph.xml. Human Readable: RFC4646 language codes must comply with the RFC by using parts from ISO 639 Digraph or ISO 639-2 Trigraph and ISO 3166 Digraph.

Code Description: Finds ddms:language elements and checks its qualifier attribute for a value of [RFC4646]. If this value is found it will ensure that the value of the element's ddms:value attribute exists in the CVEnumIRMISO639Digraph.xml or CVEnumIRMISO639-2Trigraph.xml enumeration files represented by the $iso639DigraphList or $iso639-2TrigraphList variables. Country code portions (denoted by '-' separation) must be in the CVEnumIRMCoverageISO3166Digraph.xml enumeration file represented by the $coverageIso3166TrigraphList variable.

Schematron Code:

```
<?ICEA pattern?><!-- Notices - distEditionBlockReplace-->
<sch:pattern id="IRM-ID-00010">




              <sch:rule context="ddms:language[@ddms:qualifier='RFC4646']">
       <!-- Tokenize the element Language value into a list -->
       <sch:let name="tokens" value="tokenize(@ddms:value,'-')"/>

           <!-- For convenience and readability, save the primary and
secondary subtags
            as defined in RFC 4646 -->
       <sch:let name="primarySubtag" value="$tokens[1]"/>
           <sch:let name="secondarySubtag" value="$tokens[2]"/>

           <sch:let name="badPrimaryValues" value="             if(index-
of($iso639-2TrigraphList,lower-case($primarySubtag))&gt;0 or
index-of($iso639DigraphList,$primarySubtag)&gt;0)                  then
null             else $primarySubtag"/>

           <sch:let name="badSecondaryValues" value="
if($secondarySubtag and               index-of($coverageIso3166DigraphList,
$secondarySubtag)&gt;0)                then null             else
$secondarySubtag"/>

           <sch:let name="badValues" value="string-
join(($badPrimaryValues,             $badSecondaryValues), ' ')"/>

           <!-- Check if primary subtag is valid -->
       <sch:let name="primarySubtagValid" value="
count($badPrimaryValues) = 0             "/>

           <!-- Check if secondary subtag is valid -->
       <sch:let name="secondarySubtagValid" value="
if(not($secondarySubtag)) then true() else
count($badSecondaryValues) = 0             "/>

           <sch:assert test="$primarySubtagValid and $secondarySubtagValid"
flag="error">
           [IRM-ID-00010][Error] If element ddms:language has the attribute
@ddms:qualifier
           value of [RFC4646] then the language code portion of the
@ddms:value attribute
           value must be in CVEnumIRMISO639Digraph.xml or
CVEnumIRMISO639-2Trigraph.xml
           and the country code portion, if present, must be in
CVEnumIRMCoverageISO3166Digraph.xml.

           Human Readable: RFC4646 language codes must comply with the RFC
by using parts
           from ISO 639 Digraph or ISO 639-2 Trigraph and ISO 3166 Digraph.
```

```
The following
            values were found but are not in the CVEs:
            <sch:value-of select="for $each in tokenize($badValues, ' ')
return concat('[',$each,'] ')"/>
            </sch:assert>
         </sch:rule>
      </sch:pattern>
```

2.41 - .//Rules/valueEnumerationConstraints/ IRM_ID_00027.sch

Rule Description: IRM-ID-00027 [IRM-ID-00027][] Removed in V6.

Code Description:

Schematron Code:

```
<?ICEA pattern?><!-- Notices - distEditionBlockReplace-->
<sch:pattern id="IRM-ID-00027">


      </sch:pattern>
```

2.42 - .//Rules/valueEnumerationConstraints/ IRM_ID_00028.sch

Rule Description: IRM-ID-00028 [IRM-ID-00028][] Removed in V6.

Code Description:

Schematron Code:

```
<?ICEA pattern?><!-- Notices - distEditionBlockReplace-->
<sch:pattern id="IRM-ID-00028">


      </sch:pattern>
```

2.43 - .//Rules/valueEnumerationConstraints/ IRM_ID_00031.sch

Rule Description: IRM-ID-00031 [IRM-ID-00031][Error] The element ddms:countryCode must have the attribute ddms:qualifier specified with a value in CVEnumIRMCompoundCountryCodeQualifierType.xml. Human Readable: If a qualifier is specified for a country code, it must have be defined in the CompoundCountryCodeQualifierType CVE.

Code Description: This rule uses an abstract pattern to consolidate logic. It checks that the value in parameter $searchTerm is contained in the parameter $list. The parameter $searchTerm is relative in scope to the parameter $context. The value for the parameter $list is a variable defined in the main document (IRM_XML.sch), which reads values from a specific CVE file.

Schematron Code:

```
      <sch:pattern id="IRM-ID-00031" is-a="ValidateValueExistenceInList">




            <sch:param name="ruleText" value="'   [IRM-ID-00031][Error]   The
element ddms:countryCode must have the attribute ddms:qualifier specified
with a value in CVEnumIRMCompoundCountryCodeQualifierType.xml.   '"/>

            <sch:param name="codeDesc" value="'   This rule uses an abstract
pattern to consolidate logic. It checks that the   value in parameter
$searchTerm is contained in the parameter $list. The parameter   $searchTerm
is relative in scope to the parameter $context. The value for the
parameter    $list is a variable defined in the main document (IRM_XML.sch),
which reads    values from a specific CVE file.   '"/>

            <sch:param name="context" value="ddms:countryCode"/>
            <sch:param name="searchTerm" value="@ddms:qualifier"/>
            <sch:param name="list"
value="$compoundCountryCodeQualifierTypeList"/>
            <sch:param name="errMsg" value="'   [IRM-ID-00031][Error]   The
element countryCode must have the attribute qualifier specified   with a
value in CVEnumIRMCompoundCountryCodeQualifierType.xml.   '"/>
      </sch:pattern>
```

2.44 - .//Rules/valueEnumerationConstraints/ IRM_ID_00033.sch

Rule Description: IRM-ID-00033 [IRM-ID-00033][Error] If element ddms:mimeType is specified, it must have a value from CVEnumIRMMimeType.xml. Human Readable: Values for ddms:mimeType must be defined in the MimeType CVE.

Code Description: This rule uses an abstract pattern to consolidate logic. It checks that the value in parameter $searchTerm is contained in the parameter $list. The parameter $searchTerm is relative in scope to the parameter $context. The value for the parameter $list is a variable defined in the main document (IRM_XML.sch), which reads values from a specific CVE file.

Schematron Code:

```
      <sch:pattern id="IRM-ID-00033" is-a="ValidateValueExistenceInList">
```

```
            <sch:param name="ruleText" value="'   [IRM-ID-00033][Error] If
element ddms:mimeType is specified, it must have a value from
CVEnumIRMMimeType.xml.   '"/>

            <sch:param name="codeDesc" value="'   This rule uses an abstract
pattern to consolidate logic. It checks that the   value in parameter
$searchTerm is contained in the parameter $list. The parameter   $searchTerm
is relative in scope to the parameter $context. The value for the
parameter    $list is a variable defined in the main document (IRM_XML.sch),
which reads    values from a specific CVE file.   '"/>

            <sch:param name="context" value="ddms:mimeType"/>
            <sch:param name="searchTerm" value="."/>
            <sch:param name="list" value="$mimeTypeList"/>
            <sch:param name="errMsg" value="'   [IRM-ID-00033][Error] If
element ddms:mimeType is specified, it must have a value from
CVEnumIRMMimeType.xml.   '"/>
      </sch:pattern>
```

2.45 - .//Rules/valueEnumerationConstraints/ IRM_ID_00034.sch

Rule Description: IRM-ID-00034 [IRM-ID-00034][Error] For element ddms:language, attribute ddms:qualifier must have a value in CVEnumIRMCompoundLanguageQualifierType.xml. Human Readable: If a qualifier is specified for a language, it must appear in the CompoundLanguageQualifierType CVE.

Code Description: This rule uses an abstract pattern to consolidate logic. It checks that the value in parameter $searchTerm is contained in the parameter $list. The parameter $searchTerm is relative in scope to the parameter $context. The value for the parameter $list is a variable defined in the main document (IRM_XML.sch), which reads values from a specific CVE file.

Schematron Code:

```
      <sch:pattern id="IRM-ID-00034" is-a="ValidateValueExistenceInList">



            <sch:param name="ruleText" value="'   [IRM-ID-00034][Error] For
element ddms:language, attribute ddms:qualifier must have a    value in
CVEnumIRMCompoundLanguageQualifierType.xml.   '"/>

            <sch:param name="codeDesc" value="'   This rule uses an abstract
pattern to consolidate logic. It checks that the   value in parameter
```

```
$searchTerm is contained in the parameter $list. The parameter   $searchTerm
is relative in scope to the parameter $context. The value for the
parameter    $list is a variable defined in the main document (IRM_XML.sch),
which reads    values from a specific CVE file.   '"/>

            <sch:param name="context" value="ddms:language"/>
            <sch:param name="searchTerm" value="@ddms:qualifier"/>
            <sch:param name="list"
value="$compoundLanguageQualifierTypeList"/>
            <sch:param name="errMsg" value="'   [IRM-ID-00034][Error] For
element ddms:language, attribute ddms:qualifier must have a    value in
CVEnumIRMCompoundLanguageQualifierType.xml.   '"/>
      </sch:pattern>
```

2.46 - .//Rules/valueEnumerationConstraints/ IRM_ID_00046.sch

Rule Description: IRM-ID-00046 [IRM-ID-00046][Error] If element ddms:type has attribute @ddms:qualifier specified as [urn:us:gov:ic:cvenum:irm:intel:subdisciplinetechniques] the attribute @ddms:value must be in CVEnumIRMIntelSubDisciplineTechniques.xml. Human Readable: Intel Sub Discipline Techniques must be in the CVEnumIRMIntelSubDisciplineTechniques CVE.

Code Description: This rule uses an abstract pattern to consolidate logic. It checks that the value in parameter $searchTerm is contained in the parameter $list. The parameter $searchTerm is relative in scope to the parameter $context. The value for the parameter $list is a variable defined in the main document (IRM_XML.sch), which reads values from a specific CVE file.

Schematron Code:

```
<?ICEA pattern?><!-- Notices - distEditionBlockReplace-->
<sch:pattern is-a="ValidateValueExistenceInList" id="IRM-ID-00046">




         <sch:param name="ruleText" value="'         [IRM-ID-00046]
[Error]         If element ddms:type has attribute @ddms:qualifier specified
as         [urn:us:gov:ic:cvenum:irm:intel:subdisciplinetechniques]
the attribute @ddms:type must be in
CVEnumIRMIntelSubDisciplineTechniques.xml.                  Human Readable:
Intel Sub Discipline Techniques must be in the
CVEnumIRMIntelSubDisciplineTechniques CVE.         '"/>

         <sch:param name="codeDesc" value="'         This rule uses an
abstract pattern to consolidate logic. It checks that the         value in
parameter $searchTerm is contained in the parameter $list. The
parameter         $searchTerm is relative in scope to the parameter $context.
The value for the parameter         $list is a variable defined in the main
document (IRM_XML.sch), which reads          values from a specific CVE
```

```
file.         '"/>

         <sch:param name="context"
value="ddms:type[@ddms:qualifier='urn:us:gov:ic:cvenum:irm:intel:subdiscipline
techniques']"/>
         <sch:param name="searchTerm" value="@ddms:value"/>
         <sch:param name="list" value="$subDisciplineTechniquesList"/>
         <sch:param name="errMsg" value="'         [IRM-ID-00046]
[Error]         If element ddms:type has attribute @ddms:qualifier specified
as         [urn:us:gov:ic:cvenum:irm:intel:subdisciplinetechniques]
the attribute @ddms:type must be in
CVEnumIRMIntelSubDisciplineTechniques.xml.                  Human Readable:
Intel Sub Discipline Techniques must be in the
CVEnumIRMIntelSubDisciplineTechniques CVE.         '"/>
      </sch:pattern>
```

2.47 - .//Rules/valueEnumerationConstraints/ IRM_ID_00047.sch

Rule Description: IRM-ID-00047 [IRM-ID-00047][Error] If element ddms:type has attribute @ddms:qualifier specified as [urn:us:gov:ic:cvenum:irm:intel:subdisciplines] the attribute @ddms:value must be in CVEnumIRMIntelSubDisciplines.xml. Human Readable: Intel Sub Disciplines must be in the CVEnumIRMIntelSubDisciplines CVE.

Code Description: This rule uses an abstract pattern to consolidate logic. It checks that the value in parameter $searchTerm is contained in the parameter $list. The parameter $searchTerm is relative in scope to the parameter $context. The value for the parameter $list is a variable defined in the main document (IRM_XML.sch), which reads values from a specific CVE file.

Schematron Code:

```
<?ICEA pattern?><!-- Notices - distEditionBlockReplace-->
<sch:pattern is-a="ValidateValueExistenceInList" id="IRM-ID-00047">




         <sch:param name="ruleText" value="'         [IRM-ID-00047]
[Error]         If element ddms:type has attribute @ddms:qualifier specified
as         [urn:us:gov:ic:cvenum:irm:intel:subdisciplines]         the
attribute @ddms:value must be in
CVEnumIRMIntelSubDisciplines.xml.                  Human Readable: Intel Sub
Disciplines must be in the         CVEnumIRMIntelSubDisciplines CVE.
'"/>

         <sch:param name="codeDesc" value="'         This rule uses an
abstract pattern to consolidate logic. It checks that the         value in
parameter $searchTerm is contained in the parameter $list. The
parameter         $searchTerm is relative in scope to the parameter $context.
The value for the parameter         $list is a variable defined in the main
```

```
document (IRM_XML.sch), which reads          values from a specific CVE
file.         '"/>

         <sch:param name="context"
value="ddms:type[@ddms:qualifier='urn:us:gov:ic:cvenum:irm:intel:subdiscipline
s']"/>
         <sch:param name="searchTerm" value="@ddms:value"/>
         <sch:param name="list" value="$intelSubDisciplinesList"/>
         <sch:param name="errMsg" value="'         [IRM-ID-00047]
[Error]         If element ddms:type has attribute @ddms:qualifier specified
as         [urn:us:gov:ic:cvenum:irm:intel:subdisciplines]         the
attribute @ddms:value must be in
CVEnumIRMIntelSubDisciplines.xml.                  Human Readable: Intel Sub
Disciplines must be in the          CVEnumIRMIntelSubDisciplines CVE.
'"/>
      </sch:pattern>
```

2.48 - .//Rules/valueEnumerationConstraints/ IRM_ID_00048.sch

Rule Description: IRM-ID-00048 [IRM-ID-00048][Error] If element ddms:type has attribute @ddms:qualifier specified as [urn:us:gov:ic:cvenum:irm:intel:disciplines] the attribute @ddms:value must be in CVEnumIRMIntelDisciplines.xml. Human Readable: Intel Disciplines must be in the CVEnumIRMIntelDisciplines CVE.

Code Description: This rule uses an abstract pattern to consolidate logic. It checks that the value in parameter $searchTerm is contained in the parameter $list. The parameter $searchTerm is relative in scope to the parameter $context. The value for the parameter $list is a variable defined in the main document (IRM_XML.sch), which reads values from a specific CVE file.

Schematron Code:

```
<?ICEA pattern?><!-- Notices - distEditionBlockReplace-->
<sch:pattern is-a="ValidateValueExistenceInList" id="IRM-ID-00048">




         <sch:param name="ruleText" value="'         [IRM-ID-00048]
[Error]         If element ddms:type has attribute @ddms:qualifier specified
as         [urn:us:gov:ic:cvenum:irm:intel:disciplines]         the attribute
@ddms:value must be in CVEnumIRMIntelDisciplines.xml.                  Human
Readable: Intel Disciplines must be in the          CVEnumIRMIntelDisciplines
CVE.         '"/>

         <sch:param name="codeDesc" value="'         This rule uses an
abstract pattern to consolidate logic. It checks that the         value in
parameter $searchTerm is contained in the parameter $list. The
parameter         $searchTerm is relative in scope to the parameter $context.
The value for the parameter          $list is a variable defined in the main
```

```
document (IRM_XML.sch), which reads          values from a specific CVE
file.          '"/>

         <sch:param name="context"
value="ddms:type[@ddms:qualifier='urn:us:gov:ic:cvenum:irm:intel:disciplines']
"/>
         <sch:param name="searchTerm" value="@ddms:value"/>
         <sch:param name="list" value="$intelDisciplinesList"/>
         <sch:param name="errMsg" value="'         [IRM-ID-00048]
[Error]         If element ddms:type has attribute @ddms:qualifier specified
as          [urn:us:gov:ic:cvenum:irm:intel:disciplines]         the attribute
@ddms:value must be in CVEnumIRMIntelDisciplines.xml.                  Human
Readable: Intel Disciplines must be in the          CVEnumIRMIntelDisciplines
CVE.          '"/>
      </sch:pattern>
```

2.49 - .//Rules/valueEnumerationConstraints/ IRM_ID_00049.sch

Rule Description: IRM-ID-00049 [IRM-ID-00049][Error] If element ddms:subDivisionCode has attribute @ddms:qualifier specified as [urn:us:gov:ic:cvenum:irm:coverage:iso3166-2:subcountry] the attribute @ddms:value must be in CVEnumIRMCoverageISO3166-2SubCountry.xml. Human Readable: ISO 3166-2 Sub Country codes must be in the CVEnumIRMCoverageISO3166-2SubCountry CVE.

Code Description: This rule uses an abstract pattern to consolidate logic. It checks that the value in parameter $searchTerm is contained in the parameter $list. The parameter $searchTerm is relative in scope to the parameter $context. The value for the parameter $list is a variable defined in the main document (IRM_XML.sch), which reads values from a specific CVE file.

Schematron Code:

```
<?ICEA pattern?><!-- Notices - distEditionBlockReplace-->
<sch:pattern is-a="ValidateValueExistenceInList" id="IRM-ID-00049">




         <sch:param name="ruleText" value="'         [IRM-ID-00049]
[Error]         If element ddms:subDivisionCode has attribute @ddms:qualifier
specified as
[urn:us:gov:ic:cvenum:irm:coverage:iso3166-2:subcountry]         the
attribute @ddms:value must be in
CVEnumIRMCoverageISO3166-2SubCountry.xml.                  Human Readable:
ISO 3166-2 Sub Country codes must be in the
CVEnumIRMCoverageISO3166-2SubCountry CVE.         '"/>

         <sch:param name="codeDesc" value="'         This rule uses an
abstract pattern to consolidate logic. It checks that the         value in
parameter $searchTerm is contained in the parameter $list. The
```

```
parameter         $searchTerm is relative in scope to the parameter $context.
The value for the parameter          $list is a variable defined in the main
document (IRM_XML.sch), which reads          values from a specific CVE
file.         '"/>

         <sch:param name="context"
value="ddms:subDivisionCode[@ddms:qualifier='urn:us:gov:ic:cvenum:irm:coverage
:iso3166-2:subcountry']"/>
         <sch:param name="searchTerm" value="@ddms:value"/>
         <sch:param name="list" value="$coverageIso3166-2SubCountryList"/>
         <sch:param name="errMsg" value="'         [IRM-ID-00049]
[Error]         If element ddms:subDivisionCode has attribute @ddms:qualifier
specified as
[urn:us:gov:ic:cvenum:irm:coverage:iso3166-2:subcountry]         the
attribute @ddms:value must be in
CVEnumIRMCoverageISO3166-2SubCountry.xml.                  Human Readable:
ISO 3166-2 Sub Country codes must be in the
CVEnumIRMCoverageISO3166-2SubCountry CVE.         '"/>
      </sch:pattern>
```

2.50 - .//Rules/valueEnumerationConstraints/ IRM_ID_00050.sch

Rule Description: IRM-ID-00050 [IRM-ID-00050][Error] For element ddms:productionsMetrics, attribute @ddms:subject must be in CVEnumIRMProductionMetricsSubject.xml. Human Readable: Production Metric Subjects must be in the CVEnumIRMProductionMetricsSubject CVE.

Code Description: This rule uses an abstract pattern to consolidate logic. It checks that the value in parameter $searchTerm is contained in the parameter $list. The parameter $searchTerm is relative in scope to the parameter $context. The value for the parameter $list is a variable defined in the main document (IRM_XML.sch), which reads values from a specific CVE file.

Schematron Code:

```
<?ICEA pattern?><!-- Notices - distEditionBlockReplace-->
<sch:pattern is-a="ValidateValueExistenceInList" id="IRM-ID-00050">




         <sch:param name="ruleText" value="'         [IRM-ID-00050]
[Error]         For element ddms:productionsMetrics, attribute @ddms:subject
must          be in CVEnumIRMProductionMetricsSubject.xml.
Human Readable: Production Metric Subjects must be in the
CVEnumIRMProductionMetricsSubject CVE.         '"/>

         <sch:param name="codeDesc" value="'         This rule uses an
abstract pattern to consolidate logic. It checks that the         value in
parameter $searchTerm is contained in the parameter $list. The
```

```
parameter         $searchTerm is relative in scope to the parameter $context.
The value for the parameter          $list is a variable defined in the main
document (IRM_XML.sch), which reads          values from a specific CVE
file.         '"/>

         <sch:param name="context"
value="ddms:productionMetric[@ddms:subject]"/>
         <sch:param name="searchTerm" value="@ddms:subject"/>
         <sch:param name="list" value="$productionMetricsSubjectList"/>
         <sch:param name="errMsg" value="'         [IRM-ID-00050]
[Error]         For element ddms:productionsMetrics, attribute @ddms:subject
must          be in CVEnumIRMProductionMetricsSubject.xml.
Human Readable: Production Metric Subjects must be in the
CVEnumIRMProductionMetricsSubject CVE.         '"/>
      </sch:pattern>
```

2.51 - .//Rules/valueEnumerationConstraints/ IRM_ID_00051.sch

Rule Description: IRM-ID-00051 [IRM-ID-00051][Error] For element ddms:productionsMetrics, attribute @ddms:coverage must be in CVEnumIRMProductionMetricsCoverage.xml. Human Readable: Production Metric Coverage values must be in the CVEnumIRMProductionMetricsCoverage CVE.

Code Description: This rule uses an abstract pattern to consolidate logic. It checks that the value in parameter $searchTerm is contained in the parameter $list. The parameter $searchTerm is relative in scope to the parameter $context. The value for the parameter $list is a variable defined in the main document (IRM_XML.sch), which reads values from a specific CVE file.

Schematron Code:

```
<?ICEA pattern?><!-- Notices - distEditionBlockReplace-->
<sch:pattern is-a="ValidateValueExistenceInList" id="IRM-ID-00051">




         <sch:param name="ruleText" value="'         [IRM-ID-00051]
[Error]         For element ddms:productionsMetrics, attribute @ddms:coverage
must          be in CVEnumIRMProductionMetricsCoverage.xml.
Human Readable: Production Metric Coverage values must be in the
CVEnumIRMProductionMetricsCoverage CVE.         '"/>

         <sch:param name="codeDesc" value="'         This rule uses an
abstract pattern to consolidate logic. It checks that the         value in
parameter $searchTerm is contained in the parameter $list. The
parameter         $searchTerm is relative in scope to the parameter $context.
The value for the parameter          $list is a variable defined in the main
document (IRM_XML.sch), which reads          values from a specific CVE
file.         '"/>
```

```
        <sch:param name="context"
value="ddms:productionMetric[@ddms:coverage]"/>
        <sch:param name="searchTerm" value="@ddms:coverage"/>
        <sch:param name="list" value="$productionMetricsCoverageList"/>
        <sch:param name="errMsg" value="'         [IRM-ID-00051]
[Error]         For element ddms:productionsMetrics, attribute @ddms:coverage
must          be in CVEnumIRMProductionMetricsCoverage.xml.
Human Readable: Production Metric Coverage values must be in the
CVEnumIRMProductionMetricsCoverage CVE.         '"/>
      </sch:pattern>
```

2.52 - .//Rules/valueEnumerationConstraints/ IRM_ID_00052.sch

Rule Description: IRM-ID-00052 [IRM-ID-00052][Error] If element ddms:organization has attribute @ddms:acronym specified, then the value must be in CVEnumIRMAgencyAcronym.xml. Human Readable: Agency acronyms must be in the CVEnumIRMAgencyAcronym CVE.

Code Description: This rule uses an abstract pattern to consolidate logic. It checks that the value in parameter $searchTerm is contained in the parameter $list. The parameter $searchTerm is relative in scope to the parameter $context. The value for the parameter $list is a variable defined in the main document (IRM_XML.sch), which reads values from a specific CVE file.

Schematron Code:

```
<?ICEA pattern?><!-- Notices - distEditionBlockReplace-->
<sch:pattern is-a="ValidateValueExistenceInList" id="IRM-ID-00052">




        <sch:param name="ruleText" value="'         [IRM-ID-00052]
[Error]         If element ddms:organization has attribute @ddms:acronym
specified,         then the value must be in
CVEnumIRMAgencyAcronym.xml.                  Human Readable: Agency acronyms
must be in the CVEnumIRMAgencyAcronym CVE.         '"/>

        <sch:param name="codeDesc" value="'         This rule uses an
abstract pattern to consolidate logic. It checks that the         value in
parameter $searchTerm is contained in the parameter $list. The
parameter         $searchTerm is relative in scope to the parameter $context.
The value for the parameter          $list is a variable defined in the main
document (IRM_XML.sch), which reads          values from a specific CVE
file.         '"/>

        <sch:param name="context" value="ddms:organization[@ddms:acronym]"/>
        <sch:param name="searchTerm" value="@ddms:acronym"/>
        <sch:param name="list" value="$agencyAcronymList"/>
```

```
        <sch:param name="errMsg" value="'         [IRM-ID-00052]
[Error]         If element ddms:organization has attribute @ddms:acronym
specified,         then the value must be in
CVEnumIRMAgencyAcronym.xml.                  Human Readable: Agency acronyms
must be in the CVEnumIRMAgencyAcronym CVE.         '"/>
      </sch:pattern>
```

2.53 - .//Rules/valueEnumerationConstraints/ IRM_ID_00053.sch

Rule Description: IRM-ID-00053 [IRM-ID-00053][Error] If element ddms:type has attribute @ddms:qualifier specified as [urn:us:gov:ic:cvenum:irm:activity] the attribute @ddms:value must be in CVEnumIRMActivity.xml. Human Readable: Activity must be in the CVEnumIRMActivity CVE.

Code Description: This rule uses an abstract pattern to consolidate logic. It checks that the value in parameter $searchTerm is contained in the parameter $list. The parameter $searchTerm is relative in scope to the parameter $context. The value for the parameter $list is a variable defined in the main document (IRM_XML.sch), which reads values from a specific CVE file.

Schematron Code:

```
     <sch:pattern is-a="ValidateValueExistenceInList" id="IRM-ID-00053">



        <sch:param name="ruleText" value="'         [IRM-ID-00053]
[Error]         If element ddms:type has attribute @ddms:qualifier specified
as         [urn:us:gov:ic:cvenum:irm:activity]         the attribute
@ddms:type must be in CVEnumIRMActivity.xml.                  Human Readable:
Activity must be in the          CVEnumIRMActivity CVE.         '"/>

        <sch:param name="codeDesc" value="'         This rule uses an
abstract pattern to consolidate logic. It checks that the         value in
parameter $searchTerm is contained in the parameter $list. The
parameter         $searchTerm is relative in scope to the parameter $context.
The value for the parameter          $list is a variable defined in the main
document (IRM_XML.sch), which reads          values from a specific CVE
file.         '"/>

        <sch:param name="context"
value="ddms:type[@ddms:qualifier='urn:us:gov:ic:cvenum:irm:activity']"/>
        <sch:param name="searchTerm" value="@ddms:value"/>
        <sch:param name="list" value="$activityList"/>
        <sch:param name="errMsg" value="'         [IRM-ID-00053]
[Error]         If element ddms:type has attribute @ddms:qualifier specified
as         [urn:us:gov:ic:cvenum:irm:activity]         the attribute
@ddms:type must be in CVEnumIRMActivity.xml.                  Human Readable:
```

```
Activity must be in the          CVEnumIRMActivity CVE.          '"/>
      </sch:pattern>
```

2.54 - .//Rules/valueEnumerationConstraints/ IRM_ID_00054.sch

Rule Description: IRM-ID-00054 [IRM-ID-00054][Error] If element ddms:geospatialCoverage has attribute @ddms:precedence specified, then the value must be in CVEnumIRMCoveragePrecedence.xml. Human Readable: Agency acronyms must be in the CVEnumIRMCoveragePrecedence CVE.

Code Description: This rule uses an abstract pattern to consolidate logic. It checks that the value in parameter $searchTerm is contained in the parameter $list. The parameter $searchTerm is relative in scope to the parameter $context. The value for the parameter $list is a variable defined in the main document (IRM_XML.sch), which reads values from a specific CVE file.

Schematron Code:

```
      <sch:pattern is-a="ValidateValueExistenceInList" id="IRM-ID-00054">



         <sch:param name="ruleText" value="'         [IRM-ID-00054]
[Error]         If element ddms:geospatialCoverage has attribute
@ddms:precedence specified,          then the value must be in
CVEnumIRMCoveragePrecedence.xml.                  Human Readable: Agency
acronyms must be in the CVEnumIRMCoveragePrecedence CVE.          '"/>

         <sch:param name="codeDesc" value="'         This rule uses an
abstract pattern to consolidate logic. It checks that the         value in
parameter $searchTerm is contained in the parameter $list. The
parameter         $searchTerm is relative in scope to the parameter $context.
The value for the parameter          $list is a variable defined in the main
document (IRM_XML.sch), which reads          values from a specific CVE
file.         '"/>

         <sch:param name="context"
value="ddms:geospatialCoverage[@ddms:precedence]"/>
         <sch:param name="searchTerm" value="@ddms:precedence"/>
         <sch:param name="list" value="$coveragePrecedenceList"/>
         <sch:param name="errMsg" value="'          [IRM-ID-00054]
[Error]         If element ddms:geospatialCoverage has attribute
@ddms:precedence specified,          then the value must be in
CVEnumIRMCoveragePrecedence.xml.                  Human Readable: Agency
acronyms must be in the CVEnumIRMCoveragePrecedence CVE.          '"/>
      </sch:pattern>
```

2.55 - .//Rules/xlinkConstraints/IRM_ID_00036.sch

Rule Description: IRM-ID-00036 [IRM-ID-00036][Error] For any element, if any attribute is specified with the xlink namespace [http://www.w3.org/1999/xlink], then attributes @xlink:type and/or @xlink:href must be specified. Human Readable: If any XLink attributes are specified for an element, then the type and/or URL of the link must also be specified.

Code Description: Makes sure that for each element that has any attribute in the xlink namespace has either xlink:type or xlink:href specified.

Schematron Code:

```
<?ICEA pattern?><!-- Notices - distEditionBlockReplace-->
<sch:pattern id="IRM-ID-00036">




            <sch:rule context="*[@xlink:*]">
                  <sch:assert test="normalize-space(string(@xlink:type))
or normalize-space(string(@xlink:href))" flag="error">
            [IRM-ID-00036][Error] For any element, if any attribute is
specified with the
            xlink namespace [http://www.w3.org/1999/xlink], then attributes
@xlink:type and/or
            @xlink:href must be specified.
        </sch:assert>
            </sch:rule>
      </sch:pattern>
```

Chapter 3 - Abstract Patterns

3.1 - .//Lib/CompareDateTimes.sch

Rule Description: CompareDateTimes

Code Description:

Schematron Code:

```
<?ICEA abstractPattern?><!-- Notices - distEditionBlockReplace--><!--
    $context := an xpath to an element
    $primaryDate := an xpath, relative to $context, to a date to compare
against all dates in $secondaryDateList
    $secondaryDateList := a list of xpaths, relative to $context, each to a
dates in which to compare against $primaryDate
    $operator := the equality operator to use for comparing each date in
$secondaryDateList to $primaryDate

    First, we make sure that the primaryDate is an allowable date format. If
the primary date is not a valid
    date format, then we return true because we cannot guarantee the value
provided is not allowed. Then, for
    each date in $secondaryDateList we perform the same check for a valid
date format and compare the
    secondaryDate to the primaryDate. To perform comparisons between dates,
we use the comparison operator
    contained in the param $operator and make sure that all comparisons
between primary and secondary dates
    returns true.
-->
<sch:pattern abstract="true" id="CompareDateTimes">




        <sch:rule context="$context">
           <sch:assert test="             if ($flag = 'warning' and
dtf:isAllowableDateTimeFormat(string($primaryDate)))             then every
$secondaryDate in $secondaryDateList satisfies
if(dtf:isAllowableDateTimeFormat(string($secondaryDate)))
then dtf:compareDateTimeRanges(string($primaryDate), $operator,
string($secondaryDate))                 else true()             else
true()             " flag="warning">
              <sch:value-of select="$ruleText"/>
           </sch:assert>
           <sch:assert test="             if ($flag = 'error' and
dtf:isAllowableDateTimeFormat(string($primaryDate)))             then every
$secondaryDate in $secondaryDateList satisfies
if(dtf:isAllowableDateTimeFormat(string($secondaryDate)))             then
dtf:compareDateTimeRanges(string($primaryDate), $operator,
string($secondaryDate))             else true()             else
true()             " flag="error">
```

```
            <sch:value-of select="$ruleText"/>
         </sch:assert>
      </sch:rule>
   </sch:pattern>
```

3.2 - .//Lib/IsmEnforcement.sch

Rule Description: IsmEnforcement

Code Description:

Schematron Code:

```
<?ICEA abstractPattern?><!-- Notices - distEditionBlockReplace--><!--
    This abstract pattern checks that if a particular qualifier is specified
on a
    ddms:type element that ism:classification is also specified.

    $qualifier   := the qualifier value that requires ism to be present
    $errMsg      := the error message text to display when the assertion fails

    Example usage:
    <sch:pattern is-a="DdmsTypeIsmEnforcement" id="IRM_ID_00039"
xmlns:sch="http://purl.oclc.org/dsdl/schematron">
    <sch:param name="ruleText" value=""/>
    <sch:param name="codeDesc" value=""/>
    <sch:param name="context" value="ddms:type[@ddms:qualifier=$qualifier]"/>
    <sch:param name="errMsg" value="'
    [IRM-ID-00039][Error]
    If ddms:type is specified with a qualifier of
urn:us:gov:ic:irm:productline then
    ism:classification must also be specified.
    '"/>
    </sch:pattern>

    Note: $iso4217TrigraphList is defined in the main document, IRM_XML.xml.
-->
<sch:pattern abstract="true" id="IsmEnforcement">



        <sch:rule context="$context">
           <sch:assert test="@ism:classification" flag="error">
              <sch:value-of select="$errMsg"/>
           </sch:assert>
        </sch:rule>
     </sch:pattern>
```

3.3 - .//Lib/ValidateValueExistenceInList.sch

Rule Description: ValidateValueExistenceInList

Code Description:

Schematron Code:

```
<?ICEA abstractPattern?><!-- Notices - distEditionBlockReplace--><!--
    This abstract pattern checks to see if an attribute of an element exists
in a list.

    $context     := the context in which the searchValue exists
    $searchTerm  := the value which you want to verify is in the list
    $list        := the list in which to search for the searchValue
    $errMsg      := the error message text to display when the assertion fails

    Example usage:
    <sch:pattern is-a="ValidateValueExistenceInList" id="IRM_ID_00027"
xmlns:sch="http://purl.oclc.org/dsdl/schematron">
        <sch:param name="context" value="//
irm:CountryCode[@irm:vocabulary='FIPS']"/>
        <sch:param name="searchTerm" value="."/>
        <sch:param name="list" value="$coverageFipsDigraphList"/>
        <sch:param name="errMsg" value="'
            [IRM-ID-00027][Error]
            If element CountryCode has attribute vocabulary specified as FIPS
            the element value must be in CVEnumIRMCoverageFIPSDigraph.xml.
        '"/>
    </sch:pattern>

    Note: $iso4217TrigraphList is defined in the main document, IRM_XML.xml.
-->
<sch:pattern abstract="true" id="ValidateValueExistenceInList">



         <sch:rule context="$context">
            <sch:assert test="             some $token in $list
satisfies              $token = $searchTerm or
matches($searchTerm, concat('^',$token,'$'))                              "
flag="error">
               <sch:value-of select="$errMsg"/>
            </sch:assert>
         </sch:rule>
      </sch:pattern>
```

Chapter 4 - Schematron Schema

4.1 - .//IRM_XML.sch

Rule Description: This is the root file for the IRM Schematron ruleset. It loads all of the required CVEs, declares some variables, and includes all of the Rule .sch files.

Code Description: This is the root file for the IRM Schematron ruleset. It loads all of the required CVEs, declares some variables, and includes all of the Rule .sch files.

Schematron Code:

```
<?ICEA master?><!-- UNCLASSIFIED --><!-- Notices - distEditionBlockReplace--
><!-- WARNING:
    Once compiled into an XSLT the result will
    be the aggregate classification of all the CVES
    and included .sch files
-->
<sch:schema xmlns:cve="urn:us:gov:ic:cve" xmlns:xsl="http://www.w3.org/
1999/XSL/Transform" queryBinding="xslt2">
         <sch:ns uri="http://www.w3.org/2001/XMLSchema" prefix="xsd"/>
         <sch:ns uri="urn:us:gov:ic:ism" prefix="ism"/>
         <sch:ns uri="urn:us:gov:ic:irm" prefix="irm"/>
         <sch:ns uri="urn:us:mil:ces:metadata:ddms:4" prefix="ddms"/>
         <sch:ns uri="urn:us:gov:ic:cve" prefix="cve"/>
         <sch:ns uri="http://www.w3.org/1999/xlink" prefix="xlink"/>
         <sch:ns uri="http://www.w3.org/1999/XSL/Transform" prefix="xsl"/>
         <sch:ns uri="date:time:function" prefix="dtf"/>



         <!-- (U) Resources  -->
    <sch:let name="agencyAcronymList" value="document('../../CVE/IRM/
CVEnumIRMAgencyAcronym.xml')//cve:CVE/cve:Enumeration/cve:Term/cve:Value"/>
            <sch:let name="coverageFipsDigraphList"
value="document('../../CVE/IRM/CVEnumIRMCoverageFIPSDigraph.xml')//cve:CVE/
cve:Enumeration/cve:Term/cve:Value"/>
         <sch:let name="coverageIso3166DigraphList"
value="document('../../CVE/IRM/CVEnumIRMCoverageISO3166Digraph.xml')//cve:CVE/
cve:Enumeration/cve:Term/cve:Value"/>
         <sch:let name="coverageIso3166TrigraphList"
value="document('../../CVE/IRM/CVEnumIRMCoverageISO3166Trigraph.xml')//
cve:CVE/cve:Enumeration/cve:Term/cve:Value"/>
         <sch:let name="coverageIso3166-2SubCountryList"
value="document('../../CVE/IRM/CVEnumIRMCoverageISO3166-2SubCountry.xml')//
cve:CVE/cve:Enumeration/cve:Term/cve:Value"/>
         <sch:let name="iso639DigraphList" value="document('../../CVE/IRM/
CVEnumIRMISO639Digraph.xml')//cve:CVE/cve:Enumeration/cve:Term/cve:Value"/>
         <sch:let name="iso639-2TrigraphList" value="document('../../CVE/IRM/
CVEnumIRMISO639-2Trigraph.xml')//cve:CVE/cve:Enumeration/cve:Term/cve:Value"/>
         <sch:let name="iso639-3TrigraphList" value="document('../../CVE/IRM/
CVEnumIRMISO639-3Trigraph.xml')//cve:CVE/cve:Enumeration/cve:Term/cve:Value"/>
         <sch:let name="mimeTypeList" value="document('../../CVE/IRM/
```

```
CVEnumIRMMimeType.xml')//cve:Value"/>
         <sch:let name="compoundCountryCodeQualifierTypeList"
value="document('../../CVE/IRM/
CVEnumIRMCompoundCountryCodeQualifierType.xml')//cve:Value"/>
         <sch:let name="compoundLanguageQualifierTypeList"
value="document('../../CVE/IRM/CVEnumIRMCompoundLanguageQualifierType.xml')//
cve:Value"/>
            <sch:let name="subDisciplineTechniquesV1List"
value="document('../../CVE/IRM/CVEnumIRMIntelSubDisciplineTechniques.xml')//
cve:Value"/>
            <sch:let name="subDisciplineTechniquesList"
value="document('../../CVE/IRM/CVEnumIRMIntelSubDisciplineTechniques.xml')//
cve:Value"/>
            <sch:let name="intelSubDisciplinesList"
value="document('../../CVE/IRM/CVEnumIRMIntelSubDisciplines.xml')//
cve:Value"/>
            <sch:let name="intelDisciplinesList"
value="document('../../CVE/IRM/CVEnumIRMIntelDisciplines.xml')//cve:Value"/>
            <sch:let name="productionMetricsSubjectList"
value="document('../../CVE/IRM/CVEnumIRMProductionMetricsSubject.xml')//
cve:Value"/>
            <sch:let name="productionMetricsCoverageList"
value="document('../../CVE/IRM/CVEnumIRMProductionMetricsCoverage.xml')//
cve:Value"/>
         <sch:let name="activityList" value="document('../../CVE/IRM/
CVEnumIRMActivity.xml')//cve:Value"/>
         <sch:let name="coveragePrecedenceList"
value="document('../../CVE/IRM/CVEnumIRMCoveragePrecedence.xml')//cve:Value"/>

         <!-- ************************** -->
   <!-- * General Global Variables * -->
   <!-- ************************** -->

   <sch:let name="currentYear" value="year-from-dateTime(current-
dateTime())"/>
         <sch:let name="timeZoneRegEx" value="'Z|[\+-]\d{2}:\d{2}'"/>
         <sch:let name="endsWithTimeZoneRegEx" value="concat('^.*',
$timeZoneRegEx,'$')"/>
         <sch:let name="startDateTimeTemplate"
value="'0001-01-01T00:00:00.000'"/>
         <sch:let name="endDateTimeTemplate"
value="'9999-12-01T23:59:59.999'"/>
         <sch:let name="defaultTimeZone" value="'Z'"/>

         <!-- ********************************** -->
   <!-- * Abstract Rule and Pattern Includes * -->
   <!-- ********************************** -->

   <sch:include href="./Lib/ValidateValueExistenceInList.sch"/>
         <sch:include href="./Lib/DateListYearRangeRule.sch"/>
         <sch:include href="./Lib/DateYearRangeRule.sch"/>
         <sch:include href="./Lib/CompareDateTimes.sch"/>
         <sch:include href="./Lib/IsmEnforcement.sch"/>
```

```
            <!-- ********************************** -->
    <!-- * Custom XSLT2 Function Definitions * -->
    <!-- ********************************** -->
    <!--
        Returns the maximum day of the month for an xs:dateTime as an
xs:string.
        @param {xs:dateTime} date The date time from which to get the month
        @returns {xs:string} String representation of the maximum day of the
month
    -->
    <xsl:function name="dtf:getMaxDay" as="xs:string">
                <xsl:param name="date" as="xs:dateTime"/>
                <xsl:variable name="month"
select="number(dtf:getMonth(string($date)))"/>
                <xsl:choose>
                      <xsl:when test="$month = (1,3,5,7,8,10,12)">
                            <xsl:value-of select="31"/>
                      </xsl:when>
                      <xsl:when test="$month = (4,6,9,11)">
                            <xsl:value-of select="30"/>
                      </xsl:when>
                      <xsl:otherwise>
                            <xsl:choose>
                                  <xsl:when
test="dtf:isLeapYear(string($date))">
                                        <xsl:value-of select="29"/>
                                  </xsl:when>
                                  <xsl:otherwise>
                                        <xsl:value-of select="28"/>
                                  </xsl:otherwise>
                            </xsl:choose>
                      </xsl:otherwise>
                </xsl:choose>
            </xsl:function>

            <!--
        @param {xs:date} date String representation of a date
        @returns {xs:boolean} Returns true if the date provided occurs in a
            leap year; otherwise returns false.
    -->
    <xsl:function name="dtf:isLeapYear" as="xs:boolean">
                <xsl:param name="date" as="xs:string"/>
                <xsl:variable name="year" as="xs:integer"
select="xs:integer(dtf:getYear($date))"/>
                <xsl:choose>
                      <xsl:when test="$year mod 100 = 0">
                            <xsl:choose>
                                  <xsl:when test="$year mod 400 = 0">
                                        <xsl:value-of select="true()"/>
                                  </xsl:when>
                                  <xsl:otherwise>
                                        <xsl:value-of select="false()"/>
                                  </xsl:otherwise>
                            </xsl:choose>
```

```
                        </xsl:when>
                        <xsl:otherwise>
                              <xsl:choose>
                                    <xsl:when test="$year mod 4 = 0">
                                          <xsl:value-of select="true()"/>
                                    </xsl:when>
                                    <xsl:otherwise>
                                          <xsl:value-of select="false()"/>
                                    </xsl:otherwise>
                              </xsl:choose>
                        </xsl:otherwise>
                  </xsl:choose>
            </xsl:function>

            <!--
        Replaces the day portion of the provided dateTime with the new day
provided.
        @param {xs:dateTime} dateTime An xs:dateTime to be updated with new
day.
        @param {xs:string} newDayString String representation of day portion
of a date.
        @returns {xs:dateTime} Returns new xs:dateTime with updated day
portion.
        leap year; otherwise returns false.
    -->
    <xsl:function name="dtf:replaceDateTimeDay" as="xs:dateTime">
                  <xsl:param name="dateTime" as="xs:dateTime"/>
                  <xsl:param name="newDayString" as="xs:string"/>
                  <xsl:variable name="beforeDay"
select="substring(string($dateTime), 1, 8)"/>
                  <xsl:variable name="afterDay"
select="substring(string($dateTime), 11)"/>
                  <xsl:value-of select="concat($beforeDay, $newDayString,
$afterDay)"/>
            </xsl:function>

            <!--
        Returns a string representation of the year portion of the date
        represented by the provided string.
        @param {xs:string} dateString String representation of a date in one
            of the allowable formats.
        @returns {xs:string} String representation of the year portion of the
            date represented by the provided string.
    -->
    <xsl:function name="dtf:getYear" as="xs:string">
                  <xsl:param name="dateString" as="xs:string"/>
                  <xsl:value-of
select="substring(dtf:removeTimeZone($dateString), 1, 4)"/>
            </xsl:function>

            <!--
        Returns a string representation of the month portion of the date
        represented by the provided string.
        @param {xs:string} dateString String representation of a date in one
```

```
            of the allowable formats.
        @returns {xs:string} String representation of the month portion of
the
        date represented by the provided string.
    -->
    <xsl:function name="dtf:getMonth" as="xs:string">
                <xsl:param name="dateString" as="xs:string"/>
                <xsl:value-of
select="substring(dtf:removeTimeZone($dateString), 6, 2)"/>
            </xsl:function>

            <!--
        Returns a string representation of the day portion of the date
        represented by the provided string.
        @param {xs:string} dateString String representation of a date in one
            of the allowable formats.
        @returns {xs:string} String representation of the day portion of the
        date represented by the provided string.
    -->
    <xsl:function name="dtf:getDay" as="xs:string">
                <xsl:param name="dateString" as="xs:string"/>
                <xsl:value-of
select="substring(dtf:removeTimeZone($dateString), 9, 2)"/>
            </xsl:function>

            <!--
        Returns a string representation of the timezone portion of the date
        represented by the provided string.
        @param {xs:string} dateString String representation of a date in one
            of the allowable formats.
        @returns {xs:string} String representation of the timezone portion of
            the date represented by the provided string.
    -->
    <xsl:function name="dtf:getTimeZone" as="xs:string">
                <xsl:param name="dateString" as="xs:string"/>
                <xsl:variable name="dateTimeEndingWithTimezone"
as="xs:string" select="concat('^.*(',$timeZoneRegEx,')$')"/>
                <xsl:choose>
                      <xsl:when test="matches($dateString,
$dateTimeEndingWithTimezone)">
                            <xsl:value-of select="replace($dateString,
$dateTimeEndingWithTimezone, '$1')"/>
                      </xsl:when>
                      <xsl:otherwise>
                            <xsl:value-of select="$defaultTimeZone"/>
                      </xsl:otherwise>
                </xsl:choose>
            </xsl:function>

            <!--
        Returns true if the year portion of the date represented by the
provided
        string contains four (4) digits; otherwise returns false.
        @param {xs:string} dateString String representation of a date in one
```

```
            of the allowable formats.
        @returns {xs:string} true if the year portion of the date represented
by
            the provided string contains four (4) digits; otherwise returns
false.
    -->
    <xsl:function name="dtf:yearPortionHasFourDigits" as="xs:boolean">
                <xsl:param name="dateString" as="xs:string"/>
                <xsl:variable
name="dateWithOnlyFourDigitYearAndOptionalTimeZoneRegEx" as="xs:string"
select="concat('^\d{4}(',$timeZoneRegEx,')?$')"/>
                <xsl:variable name="dateStartingWithFourDigitYearRegEx"
as="xs:string" select="'^\d{4}-.*$'"/>
                <xsl:value-of select="matches($dateString,
$dateWithOnlyFourDigitYearAndOptionalTimeZoneRegEx)
or                          matches($dateString,
$dateStartingWithFourDigitYearRegEx)"/>
            </xsl:function>

            <!--
        Removes the timezone portion of the date represented by the provided
        string and returns all remaining portions.
        @param {xs:string} dateString String representation of a date in one
            of the allowable formats.
        @returns {xs:string} String representation of a date without a
timezone
            portion.
    -->
    <xsl:function name="dtf:removeTimeZone" as="xs:string">
                <xsl:param name="dateString" as="xs:string"/>
                <xsl:value-of select="replace($dateString, $timeZoneRegEx,
'')"/>
            </xsl:function>

            <!--
        Uses the template provided to fill in missing portions of the string
        representation of a dateTime provided and returns a full xs:dateTime.
        The dateString provided must not contain a timezone.
        @param {xs:string} dateString String representation of a date in one
            of the allowable formats.
        @param {xs:string} dateTemplateString String template of a default
date
            from which to pad missing portions of the dateString parameter.
        @returns {xs:dateTime} An xs:dateTime represented by the string date
provided.
    -->
    <xsl:function name="dtf:padDateTimeWithTemplate" as="xs:dateTime">
                <xsl:param name="dateString" as="xs:string"/>
                <xsl:param name="dateTemplateString" as="xs:string"/>
                <xsl:value-of select="concat($dateString,
substring($dateTemplateString, string-length(normalize-space($dateString))
+1))"/>
            </xsl:function>
```

```
            <!--
        Returns true if the string provided represents an allowable dateTime
        format; false, otherwise. The allowable dateTime formats are defined
        in the DES for the PUBS.XML specification.
        @returns {xs:boolean} Returns true if the string provided represents
an
            allowable dateTime format; false, otherwise.
    -->
    <xsl:function name="dtf:isAllowableDateTimeFormat" as="xs:boolean">
                <xsl:param name="input" as="xs:string"/>
                <xsl:variable name="trimmedInput" as="xs:string"
select="normalize-space($input)"/>

                <!-- year -->
        <xsl:variable name="YYYY" as="xs:string" select="'^\d{4}(Z|[\+-]\d{2}:
\d{2})?$'"/>

                <!-- year, month -->
        <xsl:variable name="YYYY-MM" as="xs:string" select="'^\d{4}-\d{2}(Z|[\
+-]\d{2}:\d{2})?$'"/>

                <!-- year, month, day -->
        <xsl:variable name="YYYY-MM-DD" as="xs:string" select="'^\d{4}-\d{2}-
\d{2}(Z|[\+-]\d{2}:\d{2})?$'"/>

                <!-- year, month, day, hour, minute -->
        <xsl:variable name="YYYY-MM-DDThh-mm" as="xs:string" select="'^\d{4}-
\d{2}-\d{2}T\d{2}:\d{2}(Z|[\+-]\d{2}:\d{2})?$'"/>

                <!-- year, month, day, hour, minute, seconds, optional
milliseconds -->
        <xsl:variable name="YYYY-MM-DDThh-mm-ss" as="xs:string" select="'^
\d{4}-\d{2}-\d{2}T\d{2}:\d{2}:\d{2}(\.\d{1,3})?(Z|[\+-]\d{2}:\d{2})?$'"/>

                <xsl:value-of select="    matches($trimmedInput, $YYYY)
or    matches($trimmedInput, $YYYY-MM) or    matches($trimmedInput, $YYYY-MM-
DD) or    matches($trimmedInput, $YYYY-MM-DDThh-mm) or
matches($trimmedInput, $YYYY-MM-DDThh-mm-ss)    "/>
            </xsl:function>

            <!--
        Returns the earliest xs:dateTime possible for the provided string
        representation of a dateTime. Fills in missing portions of the
        dateTime with the earliest possible values. Default values for missing
        portions:
        MM = 01
        DD = 01
        hh = 00
        mm = 00
        ss = 00
        s  = 000
        @param {xs:string} dateString String representation of a date in one
            of the allowable formats.
        @returns {xs:dateTime} The earliest xs:dateTime possible for the
```

```
            provided string representation of a dateTime.
    -->
    <xsl:function name="dtf:startDate" as="xs:dateTime">
                  <xsl:param name="dateString" as="xs:string"/>
                  <xsl:variable name="timeZonePortion"
select="dtf:getTimeZone($dateString)"/>
                  <xsl:variable name="dateTimePortion"
select="dtf:removeTimeZone($dateString)"/>
                  <xsl:variable name="outputDate"
select="dtf:padDateTimeWithTemplate($dateTimePortion,
$startDateTimeTemplate)"/>
                  <xsl:value-of select="concat($outputDate,
$timeZonePortion)"/>
            </xsl:function>

            <!--
        Returns the latest xs:dateTime possible for the provided string
        representation of a dateTime. Fills in missing portions of the
        dateTime with the latest possible values. Default values for missing
        portions:
        MM = 12
        DD = maximum day of the month
        hh = 23
        mm = 59
        ss = 59
        s  = 999
        @param {xs:string} dateString String representation of a date in one
            of the allowable formats.
        @returns {xs:dateTime} The latest xs:dateTime possible for the
            provided string representation of a dateTime.
    -->
    <xsl:function name="dtf:endDate" as="xs:dateTime">
                  <xsl:param name="input" as="xs:string"/>
                  <xsl:variable name="timeZonePortion"
select="dtf:getTimeZone($input)"/>
                  <xsl:variable name="dateTimePortion"
select="dtf:removeTimeZone($input)"/>
                  <xsl:variable name="outputDate"
select="dtf:padDateTimeWithTemplate($dateTimePortion, $endDateTimeTemplate)"/>
                  <xsl:variable name="outputWithCorrectedDay"
select="dtf:replaceDateTimeDay($outputDate, dtf:getMaxDay($outputDate))"/>
                  <xsl:choose>
                        <xsl:when test="dtf:getDay($input)">
                              <xsl:value-of select="concat($outputDate,
$timeZonePortion)"/>
                        </xsl:when>
                        <xsl:otherwise>
                              <xsl:value-of
select="concat($outputWithCorrectedDay, $timeZonePortion)"/>
                        </xsl:otherwise>
                  </xsl:choose>
            </xsl:function>

            <!--
```

```
        Calculates the date range implied for both primary and secondary and
        determines if there is any overlap between the two ranges. Overlap is
        defined as the start of primary date range less than or equal to the
        end of secondary date range, inclusive, and the start of the secondary
        date range less than or equal to the end of the primary date range.
        Returns true if there is any overlap; otherwise, returns false.
        @param {xs:string} primary String representation of a date in one
            of the allowable formats.
        @param {xs:string} secondary String representation of a date in one
            of the allowable formats.
        @returns {xs:boolean} Returns true if the date ranges implied by
primary
            and secondary overlap at all; otherwise, returns false.
    -->
    <xsl:function name="dtf:overlaps" as="xs:boolean">
                <xsl:param name="primary" as="xs:string"/>
                <xsl:param name="secondary" as="xs:string"/>
                <xsl:variable name="primaryStart" as="xs:dateTime"
select="dtf:startDate($primary)"/>
                <xsl:variable name="primaryEnd" as="xs:dateTime"
select="dtf:endDate($primary)"/>
                <xsl:variable name="secondaryStart" as="xs:dateTime"
select="dtf:startDate($secondary)"/>
                <xsl:variable name="secondaryEnd" as="xs:dateTime"
select="dtf:endDate($secondary)"/>
                <xsl:value-of select="$primaryStart &lt;= $secondaryEnd and
$secondaryStart &lt;= $primaryEnd"/>
            </xsl:function>

            <!--
        Determines if the date range implied by the string representation in
        primary is strictly before the date range implied by the string
        representation in secondary. Returns true if the end of the date
        range implied by primary is less than the start of the date range
        implied by secondary; otherwise, returns false.
        @param {xs:string} primary String representation of a date in one
            of the allowable formats.
        @param {xs:string} secondary String representation of a date in one
            of the allowable formats.
        @returns {xs:boolean} Returns true if the date range implied by
primary
            is strictly earlier than the date range implied by secondary;
otherwise,
            returns false.
    -->
    <xsl:function name="dtf:isBefore" as="xs:boolean">
                <xsl:param name="primary" as="xs:string"/>
                <xsl:param name="secondary" as="xs:string"/>
                <xsl:variable name="primaryEnd" as="xs:dateTime"
select="dtf:endDate($primary)"/>
                <xsl:variable name="secondaryStart" as="xs:dateTime"
select="dtf:startDate($secondary)"/>
                <xsl:value-of select="$primaryEnd &lt; $secondaryStart"/>
            </xsl:function>
```

```
            <!--
        Determines if the date range implied by the string representation in
        primary is strictly after the date range implied by the string
        representation in secondary. Returns true if the end of the date
        range implied by primary is less than the start of the date range
        implied by secondary; otherwise, returns false.
        @param {xs:string} primary String representation of a date in one
            of the allowable formats.
        @param {xs:string} secondary String representation of a date in one
            of the allowable formats.
        @returns {xs:boolean} Returns true if the date range implied by
primary
            is strictly later than the date range implied by secondary;
otherwise,
            returns false.
    -->
    <xsl:function name="dtf:isAfter" as="xs:boolean">
                <xsl:param name="primary" as="xs:string"/>
                <xsl:param name="secondary" as="xs:string"/>
                <xsl:variable name="primaryStart" as="xs:dateTime"
select="dtf:startDate($primary)"/>
                <xsl:variable name="secondaryEnd" as="xs:dateTime"
select="dtf:endDate($secondary)"/>
                <xsl:value-of select="$secondaryEnd &lt; $primaryStart"/>
            </xsl:function>

            <!--
        Determines if the date range implied by the string representation in
        primary satisfies the comparison to the date range implied by
secondary
        using the provided comparison operator; otherwise, returns false.

        Both primary and secondary must be in one of the allowable formats
        and represent dates with four digits in the year portion.
        @param {xs:string} primary String representation of a date in one
            of the allowable formats.
        @param {xs:string} secondary String representation of a date in one
            of the allowable formats.
        @returns {xs:boolean} Returns true if the date range implied by
primary
            satisfies the comparison to the date range implied by secondary
using
            the provided comparison operator; otherwise, returns false.
    -->
    <xsl:function name="dtf:compareDateTimeRanges" as="xs:boolean">
                <xsl:param name="primary" as="xs:string"/>
                <xsl:param name="operator" as="xs:string"/>
                <xsl:param name="secondary" as="xs:string"/>
                <xsl:variable
name="primaryAndSecondayYearPortionsHaveFourDigits" as="xs:boolean"
select="dtf:yearPortionHasFourDigits($primary) and
dtf:yearPortionHasFourDigits($secondary)"/>
                <xsl:choose>
```

```
                        <xsl:when
test="$primaryAndSecondayYearPortionsHaveFourDigits">
                              <xsl:variable name="primaryStart"
as="xs:dateTime" select="dtf:startDate($primary)"/>
                              <xsl:variable name="primaryEnd"
as="xs:dateTime" select="dtf:endDate($primary)"/>
                              <xsl:variable name="secondaryStart"
as="xs:dateTime" select="dtf:startDate($secondary)"/>
                              <xsl:variable name="secondaryEnd"
as="xs:dateTime" select="dtf:endDate($secondary)"/>
                              <xsl:choose>
                     <!-- 'Less Than' Edge Case -->
                     <!-- 2010-01-01T00:00:00.000Z < 2010 -->
                     <xsl:when test="($operator = 'lt' or $operator = '&lt;')
and              (($primaryStart = $primaryEnd and $primaryStart =
$secondaryStart) or              ($primaryStart = $primaryEnd and
$primaryStart = $secondaryEnd) or              ($secondaryStart =
$secondaryEnd and $primaryStart = $secondaryStart))">
                                          <xsl:value-of select="false()"/>
                                    </xsl:when>

                                    <!-- 'Greater Than' Edge Case -->
                     <!-- 2010-12-31T23:59:59.999Z > 2010 -->
                     <xsl:when test="($operator = 'gt' or $operator = '&gt;')
and              (($primaryStart = $primaryEnd and $primaryEnd =
$secondaryEnd) or              ($primaryStart = $primaryEnd and $primaryEnd =
$secondaryStart) or              ($secondaryStart = $secondaryEnd and
$primaryEnd = $secondaryEnd))">
                                          <xsl:value-of select="false()"/>
                                    </xsl:when>

                                    <!-- 'Less Than' and 'Less Than or Equal'
-->
                     <xsl:when test="$operator = 'lt' or $operator = '&lt;' or
$operator = '&lt;='">
                                          <xsl:value-of
select="dtf:isBefore($primary, $secondary) or dtf:overlaps($primary,
$secondary)"/>
                                    </xsl:when>

                                    <!-- 'Greater Than' and 'Greater Than or
Equal' -->
                     <xsl:when test="$operator = 'gt' or $operator = '&gt;' or
$operator = '&gt;='">
                                          <xsl:value-of
select="dtf:isAfter($primary, $secondary) or dtf:overlaps($primary,
$secondary)"/>
                                    </xsl:when>

                                    <!-- Default to false() -->
                     <xsl:otherwise>
                                          <xsl:value-of select="false()"/>
                                    </xsl:otherwise>
                              </xsl:choose>
```

```
                        </xsl:when>
                        <xsl:otherwise>
                              <xsl:value-of select="false()"/>
                  </xsl:otherwise>
               </xsl:choose>
         </xsl:function>

         <!--****************************-->
<!-- (U) IRM ID Rules -->
<!--****************************-->

<!--(U) dateTimeConstraints-->
   <sch:include href="./Rules/dateTimeConstraints/IRM_ID_00015.sch"/>
         <sch:include href="./Rules/dateTimeConstraints/IRM_ID_00016.sch"/>
         <sch:include href="./Rules/dateTimeConstraints/IRM_ID_00017.sch"/>
         <sch:include href="./Rules/dateTimeConstraints/IRM_ID_00018.sch"/>
         <sch:include href="./Rules/dateTimeConstraints/IRM_ID_00019.sch"/>
         <sch:include href="./Rules/dateTimeConstraints/IRM_ID_00020.sch"/>
         <sch:include href="./Rules/dateTimeConstraints/IRM_ID_00021.sch"/>
         <sch:include href="./Rules/dateTimeConstraints/IRM_ID_00022.sch"/>
         <sch:include href="./Rules/dateTimeConstraints/IRM_ID_00023.sch"/>
         <sch:include href="./Rules/dateTimeConstraints/IRM_ID_00024.sch"/>

         <!--(U) ddmsConstraints-->
   <sch:include href="./Rules/ddmsConstraints/IRM_ID_00011.sch"/>
         <sch:include href="./Rules/ddmsConstraints/IRM_ID_00012.sch"/>
         <sch:include href="./Rules/ddmsConstraints/IRM_ID_00013.sch"/>
         <sch:include href="./Rules/ddmsConstraints/IRM_ID_00014.sch"/>
         <sch:include href="./Rules/ddmsConstraints/IRM_ID_00029.sch"/>
         <sch:include href="./Rules/ddmsConstraints/IRM_ID_00030.sch"/>
         <sch:include href="./Rules/ddmsConstraints/IRM_ID_00032.sch"/>
         <sch:include href="./Rules/ddmsConstraints/IRM_ID_00035.sch"/>
         <sch:include href="./Rules/ddmsConstraints/IRM_ID_00037.sch"/>
         <sch:include href="./Rules/ddmsConstraints/IRM_ID_00038.sch"/>
         <sch:include href="./Rules/ddmsConstraints/IRM_ID_00039.sch"/>
         <sch:include href="./Rules/ddmsConstraints/IRM_ID_00055.sch"/>

         <!--(U) globalConstraints-->
   <sch:include href="./Rules/globalConstraints/IRM_ID_00002.sch"/>

         <!--(U) ismConstraints-->
   <sch:include href="./Rules/ismConstraints/IRM_ID_00025.sch"/>
         <sch:include href="./Rules/ismConstraints/IRM_ID_00026.sch"/>
         <sch:include href="./Rules/ismConstraints/IRM_ID_00040.sch"/>
         <sch:include href="./Rules/ismConstraints/IRM_ID_00041.sch"/>
         <sch:include href="./Rules/ismConstraints/IRM_ID_00042.sch"/>
         <sch:include href="./Rules/ismConstraints/IRM_ID_00043.sch"/>
         <sch:include href="./Rules/ismConstraints/IRM_ID_00044.sch"/>
         <sch:include href="./Rules/ismConstraints/IRM_ID_00045.sch"/>

         <!--(U) valueEnumerationConstraints-->
   <sch:include href="./Rules/valueEnumerationConstraints/IRM_ID_00001.sch"/>
         <sch:include href="./Rules/valueEnumerationConstraints/
IRM_ID_00003.sch"/>
```

```
         <sch:include href="./Rules/valueEnumerationConstraints/
IRM_ID_00004.sch"/>
         <sch:include href="./Rules/valueEnumerationConstraints/
IRM_ID_00005.sch"/>
         <sch:include href="./Rules/valueEnumerationConstraints/
IRM_ID_00006.sch"/>
         <sch:include href="./Rules/valueEnumerationConstraints/
IRM_ID_00007.sch"/>
         <sch:include href="./Rules/valueEnumerationConstraints/
IRM_ID_00008.sch"/>
         <sch:include href="./Rules/valueEnumerationConstraints/
IRM_ID_00009.sch"/>
         <sch:include href="./Rules/valueEnumerationConstraints/
IRM_ID_00010.sch"/>
         <sch:include href="./Rules/valueEnumerationConstraints/
IRM_ID_00027.sch"/>
         <sch:include href="./Rules/valueEnumerationConstraints/
IRM_ID_00028.sch"/>
         <sch:include href="./Rules/valueEnumerationConstraints/
IRM_ID_00031.sch"/>
         <sch:include href="./Rules/valueEnumerationConstraints/
IRM_ID_00033.sch"/>
         <sch:include href="./Rules/valueEnumerationConstraints/
IRM_ID_00034.sch"/>
         <sch:include href="./Rules/valueEnumerationConstraints/
IRM_ID_00046.sch"/>
         <sch:include href="./Rules/valueEnumerationConstraints/
IRM_ID_00047.sch"/>
         <sch:include href="./Rules/valueEnumerationConstraints/
IRM_ID_00048.sch"/>
         <sch:include href="./Rules/valueEnumerationConstraints/
IRM_ID_00049.sch"/>
         <sch:include href="./Rules/valueEnumerationConstraints/
IRM_ID_00050.sch"/>
         <sch:include href="./Rules/valueEnumerationConstraints/
IRM_ID_00051.sch"/>
         <sch:include href="./Rules/valueEnumerationConstraints/
IRM_ID_00052.sch"/>
         <sch:include href="./Rules/valueEnumerationConstraints/
IRM_ID_00053.sch"/>
         <sch:include href="./Rules/valueEnumerationConstraints/
IRM_ID_00054.sch"/>

         <!--(U) xlinkConstraints-->
   <sch:include href="./Rules/xlinkConstraints/IRM_ID_00036.sch"/>
      </sch:schema>
```