



Guide to Schematron Rules and Patterns

IRM Schematron Guide

Version 2021-NOV

December 1, 2022

Distribution Notice:

This document has been approved for Public Release and is available for use without restriction.

Table of Contents

Chapter 1 - Introduction	1
1.1 - Purpose	1
1.2 - Overview	1
1.3 - Schematron	1
1.4 - Conformance	1
Chapter 2 - Rules	2
2.1 - ../Rules/IRM_ID_00002.sch	3
2.2 - ../Rules/IRM_ID_00005.sch	4
2.3 - ../Rules/IRM_ID_00006.sch	5
2.4 - ../Rules/IRM_ID_00007.sch	6
2.5 - ../Rules/IRM_ID_00010.sch	7
2.6 - ../Rules/IRM_ID_00015.sch	9
2.7 - ../Rules/IRM_ID_00016.sch	10
2.8 - ../Rules/IRM_ID_00017.sch	11
2.9 - ../Rules/IRM_ID_00019.sch	12
2.10 - ../Rules/IRM_ID_00020.sch	13
2.11 - ../Rules/IRM_ID_00021.sch	14
2.12 - ../Rules/IRM_ID_00022.sch	15
2.13 - ../Rules/IRM_ID_00023.sch	16
2.14 - ../Rules/IRM_ID_00024.sch	17
2.15 - ../Rules/IRM_ID_00025.sch	19
2.16 - ../Rules/IRM_ID_00029.sch	20
2.17 - ../Rules/IRM_ID_00030.sch	21
2.18 - ../Rules/IRM_ID_00033.sch	22
2.19 - ../Rules/IRM_ID_00034.sch	23
2.20 - ../Rules/IRM_ID_00036.sch	24
2.21 - ../Rules/IRM_ID_00040.sch	25
2.22 - ../Rules/IRM_ID_00041.sch	26
2.23 - ../Rules/IRM_ID_00042.sch	27
2.24 - ../Rules/IRM_ID_00043.sch	28
2.25 - ../Rules/IRM_ID_00044.sch	29
2.26 - ../Rules/IRM_ID_00045.sch	30
2.27 - ../Rules/IRM_ID_00046.sch	31
2.28 - ../Rules/IRM_ID_00047.sch	32
2.29 - ../Rules/IRM_ID_00048.sch	33
2.30 - ../Rules/IRM_ID_00053.sch	34
2.31 - ../Rules/IRM_ID_00054.sch	35
2.32 - ../Rules/IRM_ID_00055.sch	36
2.33 - ../Rules/IRM_ID_00062.sch	37
2.34 - ../Rules/IRM_ID_00063.sch	38
2.35 - ../Rules/IRM_ID_00064.sch	39
2.36 - ../Rules/IRM_ID_00065.sch	40
2.37 - ../Rules/IRM_ID_00068.sch	41
2.38 - ../Rules/IRM_ID_00070.sch	42
2.39 - ../Rules/IRM_ID_00071.sch	43
2.40 - ../Rules/IRM_ID_00072.sch	44

2.41 - ./Rules/IRM_ID_00073.sch	45
2.42 - ./Rules/IRM_ID_00074.sch	46
2.43 - ./Rules/IRM_ID_00076.sch	47
2.44 - ./Rules/IRM_ID_00077.sch	48
2.45 - ./Rules/IRM_ID_00078.sch	49
2.46 - ./Rules/IRM_ID_00079.sch	52
2.47 - ./Rules/IRM_ID_00080.sch	53
2.48 - ./Rules/IRM_ID_00081.sch	54
2.49 - ./Rules/IRM_ID_00086.sch	55
2.50 - ./Rules/IRM_ID_00087.sch	56
2.51 - ./Rules/IRM_ID_00088.sch	57
2.52 - ./Rules/IRM_ID_00089.sch	58
2.53 - ./Rules/IRM_ID_00090.sch	59
2.54 - ./Rules/IRM_ID_00091.sch	60
2.55 - ./Rules/IRM_ID_00092.sch	61
2.56 - ./Rules/IRM_ID_00093.sch	62
2.57 - ./Rules/IRM_ID_00094.sch	63
2.58 - ./Rules/IRM_ID_00095.sch	64
2.59 - ./Rules/IRM_ID_00096.sch	65
2.60 - ./Rules/IRM_ID_00098.sch	66
2.61 - ./Rules/IRM_ID_00099.sch	67
2.62 - ./Rules/IRM_ID_00100.sch	68
2.63 - ./Rules/IRM_ID_00101.sch	69
2.64 - ./Rules/IRM_ID_00102.sch	70
2.65 - ./Rules/IRM_ID_00103.sch	71
2.66 - ./Rules/IRM_ID_00104.sch	72
2.67 - ./Rules/IRM_ID_00105.sch	73
2.68 - ./Rules/IRM_ID_00106.sch	74
2.69 - ./Rules/IRM_ID_00107.sch	75
2.70 - ./Rules/IRM_ID_00108.sch	76
2.71 - ./Rules/IRM_ID_00109.sch	77
2.72 - ./Rules/IRM_ID_00110.sch	78
2.73 - ./Rules/IRM_ID_00111.sch	79
2.74 - ./Rules/IRM_ID_00112.sch	80
2.75 - ./Rules/IRM_ID_00113.sch	81
2.76 - ./Rules/IRM_ID_00114.sch	82
Chapter 3 - Abstract Patterns	83
3.1 - ./Lib/CompareDateTimes.sch	84
3.2 - ./Lib/ICIdentifierRestrictions.sch	86
3.3 - ./Lib/IsmEnforcement.sch	87
3.4 - ./Lib/ValidateValidationEnvCVE.sch	88
3.5 - ./Lib/ValidateValidationEnvSchema.sch	89
3.6 - ./Lib/ValidateValueExistenceInList.sch	90
Chapter 4 - Min Accessible Rules	91
4.1 - ./Rules/IRM_ID_00062.sch	92
4.2 - ./Rules/IRM_ID_00063.sch	93
4.3 - ./Rules/IRM_ID_00064.sch	94
4.4 - ./Rules/IRM_ID_00065.sch	95
Chapter 5 - Schematron Schema	96

5.1 - ./IRM_XML.sch	97
Chapter 6 - Removed Rules	112
6.1 - ./Rules/deleted/IRM_ID_00001.sch	112
6.2 - ./Rules/deleted/IRM_ID_00003.sch	112
6.3 - ./Rules/deleted/IRM_ID_00004.sch	112
6.4 - ./Rules/deleted/IRM_ID_00008.sch	112
6.5 - ./Rules/deleted/IRM_ID_00009.sch	112
6.6 - ./Rules/deleted/IRM_ID_00011.sch	112
6.7 - ./Rules/deleted/IRM_ID_00012.sch	112
6.8 - ./Rules/deleted/IRM_ID_00013.sch	113
6.9 - ./Rules/deleted/IRM_ID_00014.sch	113
6.10 - ./Rules/deleted/IRM_ID_00018.sch	113
6.11 - ./Rules/deleted/IRM_ID_00026.sch	113
6.12 - ./Rules/deleted/IRM_ID_00027.sch	113
6.13 - ./Rules/deleted/IRM_ID_00028.sch	113
6.14 - ./Rules/deleted/IRM_ID_00031.sch	113
6.15 - ./Rules/deleted/IRM_ID_00032.sch	114
6.16 - ./Rules/deleted/IRM_ID_00035.sch	114
6.17 - ./Rules/deleted/IRM_ID_00037.sch	114
6.18 - ./Rules/deleted/IRM_ID_00038.sch	114
6.19 - ./Rules/deleted/IRM_ID_00039.sch	114
6.20 - ./Rules/deleted/IRM_ID_00049.sch	114
6.21 - ./Rules/deleted/IRM_ID_00050.sch	114
6.22 - ./Rules/deleted/IRM_ID_00051.sch	115
6.23 - ./Rules/deleted/IRM_ID_00052.sch	115
6.24 - ./Rules/deleted/IRM_ID_00056.sch	115
6.25 - ./Rules/deleted/IRM_ID_00057.sch	115
6.26 - ./Rules/deleted/IRM_ID_00058.sch	115
6.27 - ./Rules/deleted/IRM_ID_00059.sch	115
6.28 - ./Rules/deleted/IRM_ID_00060.sch	115
6.29 - ./Rules/deleted/IRM_ID_00061.sch	116
6.30 - ./Rules/deleted/IRM_ID_00066.sch	116
6.31 - ./Rules/deleted/IRM_ID_00067.sch	116
6.32 - ./Rules/deleted/IRM_ID_00069.sch	116
6.33 - ./Rules/deleted/IRM_ID_00075.sch	116
6.34 - ./Rules/deleted/IRM_ID_00082.sch	116
6.35 - ./Rules/deleted/IRM_ID_00083.sch	116
6.36 - ./Rules/deleted/IRM_ID_00084.sch	117
6.37 - ./Rules/deleted/IRM_ID_00085.sch	117
6.38 - ./Rules/deleted/IRM_ID_00097.sch	117

Chapter 1 - Introduction

1.1 - Purpose

This is an informative supplement for IRM. This guide is generated from the IRM Schematron rules and provides a consolidated reference for the business rules of this specification.

1.2 - Overview

Chapter 2 is a listing of all the numbered rules in IRM. For each rule, there is a rule description, a code description, and a code block with the Schematron rule.

Chapter 3 is a listing of abstract patterns used in IRM. The abstract patterns may be used in numbered rules or provided as reference for use in rules developed by users of IRM. Each abstract pattern has a code description and a code block with the abstract Schematron pattern.

Chapter 4 is a listing of the master IRM Schematron file with all of the imports of rules and patterns. Many of the rules and patterns listed in Chapters 3 and 4 rely on functions and variables defined in the master file.

Chapter 5 is a listing of rules that have been deleted.

1.3 - Schematron

The business rules for IRM are encoded using ISO Schematron. Schematron is a rule-based validation language that uses XML Path Language to make assertions about an XML document.

IRM uses the XSLT 2.0 implementation of Schematron by Rick Jelliffe (2010-04-14) as its reference implementation. The only available identifying descriptors for this implementation are the implementer's name and date of release. This implementation may be found at the following URL: <http://code.google.com/p/schematron/>.



Important

The Schematron rules in this specification use XSLT 2.0 query binding.

1.4 - Conformance

This guide is informative. The Schematron rules listed here are normative in the sense that they convey criteria that a document **MUST** adhere to, exactly as English may be used to convey normative criteria. It is not necessary for implementers to use the specific Schematron encoding in this specification. Implementers **MAY** use any encodings, tools, or languages desired to implement validation schemes for conformance to this specification. However, to conform to the specification, validation schemes **MUST** match the behavior of the reference Schematron implementation. That is, a validator **MUST** find a document valid *if and only if* the reference Schematron implementation would find the document valid according to IRM's Schematron rules.

Chapter 2 - Rules

All of the numbered Rules for IRM are listed in this section. These rules may depend on patterns defined in the Abstract Patterns section or on variables defined in the Schematron Schema section.

Rules identifiers are all of the format IRM-ID-XXXXX, with rule files named IRM_ID_XXXXX.sch. Any other heading indicates a supporting file that may influence a rule but is not actually a numbered rule.

2.1 - ../Rules/IRM_ID_00002.sch

Rule Description

[IRM-ID-00002][Error] For every attribute in the namespace [urn:us:gov:ic:irm], a non-whitespace value must be specified. Human Readable:

Code Description

For each element which specifies an attribute in the IRM namespace, ensure that each attribute in the IRM namespace specifies a non-whitespace value.

Schematron Code

```
<?ICEA pattern?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00002">
    <sch:rule id="IRM-ID-00002-R1"
        context="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage//*[*[namespace-uri() = ('urn:us:gov:ic:irm')]]">
        <sch:assert test="every $attribute in @[namespace-uri() = ('urn:us:gov:ic:irm')] satisfies normalize-space(string($attribute))"
            flag="error"
            role="error">[IRM-ID-00002][Error] For every attribute in the namespace [urn:us:gov:ic:irm], a non-whitespace value must be specified. Human Readable:</
sch:assert>
            </sch:rule>
        </sch:pattern>
```


2.2 - ../Rules/IRM_ID_00005.sch

Rule Description

[IRM-ID-00005][Error] If element irm:language has the attribute @irm:qualifier value of [urn:us:gov:ic:cvenum:irm:iso639:digraph] then the value of attribute @irm:value must be in CVEnumIRMISO639Digraph.xml and no country code portion may be specified in the @irm: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 irm:language's value, then the value of element irm:language will not be found in the appropriate list and the assertion will fail as expected.

Schematron Code

```
<?ICEA pattern?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00005" is-a="ValidateValueExistenceInList">
    <sch:param name="ruleText"
        value="" [IRM-ID-00005][Error] If element irm:language has the attribute @irm:qualifier value of [urn:us:gov:ic:cvenum:irm:iso639:digraph] then the value of
attribute @irm:value must be in CVEnumIRMISO639Digraph.xml and no country code portion may be specified in the irm: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 $list is a variable defined in the main document (IRM_XML.sch), which reads values from a
specific CVE file. ""/>
    <sch:param name="context"
        value="irm:language[@irm:qualifier='urn:us:gov:ic:cvenum:irm:iso639:digraph']"/>
    <sch:param name="searchTerm" value="@irm:value"/>
    <sch:param name="list" value="$iso639DigraphList"/>
    <sch:param name="errMsg"
        value="" [IRM-ID-00005][Error] If element irm:language has the attribute @irm:qualifier value of [urn:us:gov:ic:cvenum:irm:iso639:digraph] then the value of
attribute @irm:value must be in CVEnumIRMISO639Digraph.xml and no country code portion may be specified in the irm:language element value. Human Readable: ISO 639 digraph language codes must
be in the ISO 639 digraph CVE. ""/>
</sch:pattern>
```

2.3 - ../Rules/IRM_ID_00006.sch

Rule Description

[IRM-ID-00006][Error] If element irm:language has the attribute @irm:qualifier value of [urn:us:gov:ic:cvenum:irm:iso639-2:trigraph] then the value of attribute @irm:value must be in CVEnumIRMISO639-2Trigraph.xml and no country code portion may be specified in the irm: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 irm:language is in the appropriate list because if a country code portion is specified in the element irm:language's value, then the value of element irm:language will not be found in the appropriate list and the assertion will fail as expected.

Schematron Code

```
<?ICEA pattern?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00006" is-a="ValidateValueExistenceInList">
    <sch:param name="ruleText"
        value="" [IRM-ID-00006][Error] If element irm:language has the attribute @irm:qualifier value of [urn:us:gov:ic:cvenum:irm:iso639-2:trigraph] then the value of
attribute @irm:value must be in CVEnumIRMISO639-2Trigraph.xml and no country code portion may be specified in the irm: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 $list is a variable defined in the main document (IRM_XML.sch), which reads values from a
specific CVE file. ""/>
    <sch:param name="context"
        value="irm:language[@irm:qualifier='urn:us:gov:ic:cvenum:irm:iso639-2:trigraph']"/>
    <sch:param name="searchTerm" value="@irm:value"/>
    <sch:param name="list" value="$iso639-2TrigraphList"/>
    <sch:param name="errMsg"
        value="" [IRM-ID-00006][Error] If element irm:language has the attribute @irm:qualifier value of [urn:us:gov:ic:cvenum:irm:iso639-2:trigraph] then the value of
attribute @irm:value must be in CVEnumIRMISO639-2Trigraph.xml and no country code portion may be specified in the irm:value attribute value. Human Readable: ISO 639-2 trigraph language codes
must be in the ISO 639-2 trigraph CVE. ""/>
    </sch:pattern>
```

2.4 - ../Rules/IRM_ID_00007.sch

Rule Description

[IRM-ID-00007][Error] If element irm:language has the attribute @irm:qualifier value of [urn:us:gov:ic:cvenum:irm:iso639-3:trigraph] then the value of attribute @irm:value must be in CVENumIRMISO639-3Trigraph.xml and no country code portion may be specified in the irm: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 irm:language's value, then the value of element irm:language will not be found in the appropriate list and the assertion will fail as expected.

Schematron Code

```
<?ICEA pattern?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00007" is-a="ValidateValueExistenceInList">
    <sch:param name="ruleText"
        value="" [IRM-ID-00007][Error] If element irm:language has the attribute @irm:qualifier value of [urn:us:gov:ic:cvenum:irm:iso639-3:trigraph] then the value of
attribute @irm:value must be in CVENumIRMISO639-3Trigraph.xml and no country code portion may be specified in the irm: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 $list is a variable defined in the main document (IRM_XML.sch), which reads values from a
specific CVE file. ""/>
    <sch:param name="context"
        value="irm:language[@irm:qualifier='urn:us:gov:ic:cvenum:irm:iso639-3:trigraph']"/>
    <sch:param name="searchTerm" value="@irm:value"/>
    <sch:param name="list" value="$iso639-3TrigraphList"/>
    <sch:param name="errMsg"
        value="" [IRM-ID-00007][Error] If element irm:language has the attribute @irm:qualifier value of [urn:us:gov:ic:cvenum:irm:iso639-3:trigraph] then the value of
attribute @irm:value must be in CVENumIRMISO639-3Trigraph.xml and no country code portion may be specified in the irm:value attribute value. Human Readable: ISO 639-3 trigraph language codes
must in the the ISO 639-3 trigraph CVE. ""/>
    </sch:pattern>
```

2.5 - ../Rules/IRM_ID_00010.sch

Rule Description

[IRM-ID-00010][Error] If element irm:language has the attribute @irm:qualifier value of [RFC5646] then the language code portion of the @irm: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: RFC5646 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 irm:language elements and checks its qualifier attribute for a value of [RFC5646]. If this value is found it will ensure that the value of the element's irm: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 - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00010">
    <sch:rule id="IRM-ID-00010-R1" context="irm:language[@irm:qualifier='RFC5646']"><!-- Tokenize the element Language value into a list -->

        <sch:let name="tokens" value="tokenize(@irm:value, '-')"/>
            <!-- For convenience and readability, save the primary and secondary subtags
            as defined in RFC 5646 -->

        <sch:let name="primarySubtag" value="$tokens[1]"/>
            <sch:let name="secondarySubtag" value="$tokens[2]"/>
            <sch:let name="badPrimaryValues"
                value="if($primarySubtag and ( index-of($iso639-2TrigraphList,lower-case($primarySubtag))>0 or index-of($iso639DigraphList,$primarySubtag)>0)) then null
else $primarySubtag"/>
            <sch:let name="badSecondaryValues"
                value="if($secondarySubtag and index-of($coverageIso3166DigraphList,$secondarySubtag)>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="$primarySubtag and count($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"
                role="error">[IRM-ID-00010][Error] If element irm:language has the attribute @irm:qualifier value of [RFC5646] then the language code portion of the
@irm: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: RFC5646 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>
```

2.6 - ../Rules/IRM_ID_00015.sch

Rule Description

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

Code Description

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

Schematron Code

```
<?ICEA pattern?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00015">
    <sch:rule id="IRM-ID-00015-R1"
        context="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage[$IC_COMPLIANCE]//irm:dates">
        <sch:assert test="@irm:created or @irm:posted" flag="error" role="error">[IRM-ID-00015][Error] Every irm:date must have at least one of @irm:created or @irm:posted.</
sch:assert>
        </sch:rule>
    </sch:pattern>
```

2.7 - ../Rules/IRM_ID_00016.sch

Rule Description

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

Code Description

This pattern uses abstract rules to consolidate logic. For attributes, ensure that each date contained within \$dateList has a year value within the range \$minYear and \$maxYear, inclusive.

Schematron Code

```
<?ICEA pattern?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00016"><!-- Use abstract rule to handle required attributes -->

<sch:rule id="IRM-ID-00016-R1"
      context="/tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage[$IC_COMPLIANCE]//*[ @irm:approvedOn | @irm:dateProcessed | @irm:receivedOn |
@irm:posted | @irm:created | @irm:infoCutOff]">
      <sch:let name="minYear" value="1901"/>
      <sch:let name="maxYear" value="$currentYear"/>
      <sch:let name="dateList"
            value="(string(@irm:approvedOn), string(@irm:dateProcessed), string(@irm:receivedOn), string(@irm:posted), string(@irm:created), string(@irm: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 @irm:approvedOn, @irm:dateProcessed,
@irm:receivedOn, @irm:infoCutOff, @irm:posted, and @irm:created. Human Readable: Dates must be after 1901 and in the past for @irm:approvedOn, @irm:dateProcessed, @irm:receivedOn,
@irm:infoCutOff, @irm:posted, and @irm:created. '"/>
            <sch:extends rule="abs.dateListYearRangeRule"/>
      </sch:rule>
</sch:pattern>
```

2.8 - ../Rules/IRM_ID_00017.sch

Rule Description

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

Code Description

This pattern uses abstract rules to consolidate logic. For attributes, ensure that each date contained within \$dateList has a year value within the range \$minYear and \$maxYear, inclusive.

Schematron Code

```
<?ICEA pattern?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00017"><!-- Use abstract rule to handle required attributes -->

<sch:rule id="IRM-ID-00017-R1"
          context="/tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage[$IC_COMPLIANCE]//*[@irm:validTil]">
    <sch:let name="minYear" value="1901"/>
    <sch:let name="maxYear" value="9999"/>
    <sch:let name="dateList" value="(string(@irm:validTil))"/>
    <sch:let name="errMsg"
          value="' [IRM-ID-00017][Error] The permissible values for the year range are 1901 through 9999 for attribute @irm:validTil. Human Readable: @irm:validTil must
be after 1901. '"/>
    <sch:extends rule="abs.dateListYearRangeRule"/>
  </sch:rule>
</sch:pattern>
```


2.9 - ../Rules/IRM_ID_00019.sch

Rule Description

[IRM-ID-00019][Warning] @irm:approvedOn must not be later than @irm:created and @irm:posted. Human Readable: The date held in the approvedOn attribute must not be later then the dates held by either the created or posted attributes.

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 - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00019" is-a="CompareDateTimes">
  <sch:param name="ruleText"
    value="" [IRM-ID-00019][Warning] @irm:approvedOn must not be later than @irm:created and @irm: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 $operator) and makes sure that each comparison returns true. Implementation details for the abstract pattern can be
found in the abstract pattern definition file located in the Lib directory. '"/>
  <sch:param name="context"
    value="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage[$IC_COMPLIANCE]/*[@irm:approvedOn]"/>
  <sch:param name="primaryDate" value="@irm:approvedOn"/>
  <sch:param name="operator" value="'&lt;='"/>
  <sch:param name="secondaryDateList" value="(@irm:created, @irm:posted)"/>
  <sch:param name="flag" value="'warning'"/>
</sch:pattern>
```

2.10 - ../Rules/IRM_ID_00020.sch

Rule Description

[IRM-ID-00020][Error] @irm:infoCutOff must not be later than @irm:created, and @irm:posted. Human Readable: The date held in the infoCutOff attribute must not be later then the dates held by either the created or posted attributes.

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 - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00020" is-a="CompareDateTimes">
  <sch:param name="ruleText"
    value="' [IRM-ID-00020][Error] @irm:infoCutOff must not be later than @irm:created, and @irm: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 $operator) and makes sure that each comparison returns true. Implementation details for the abstract pattern can be
found in the abstract pattern definition file located in the Lib directory. '"/>
  <sch:param name="context"
    value="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage[$IC_COMPLIANCE]/*[@irm:infoCutOff]"/>
  <sch:param name="primaryDate" value="@irm:infoCutOff"/>
  <sch:param name="operator" value="'&lt;='"/>
  <sch:param name="secondaryDateList" value="(@irm:created, @irm:posted)"/>
  <sch:param name="flag" value="'error'"/>
</sch:pattern>
```

2.11 - ./Rules/IRM_ID_00021.sch

Rule Description

[IRM-ID-00021][Warning] @irm:validTil must not be earlier than @irm:created, @irm:posted, @irm:infoCutOff, and @irm:approvedOn. Human Readable: The date held by the validTil attribute must not be earlier than the dates in the created, posted, infoCutOff and approvedOn attributes.

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 - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00021" is-a="CompareDateTimes">
  <sch:param name="ruleText"
    value="" [IRM-ID-00021][Warning] @irm:validTil must not be earlier than @irm:created, @irm:posted, @irm:infoCutOff, and @irm: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 $operator) and makes sure that each comparison returns true. Implementation details for the abstract pattern can be
found in the abstract pattern definition file located in the Lib directory. '"/>
  <sch:param name="context"
    value="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage[$IC_COMPLIANCE]//*[@irm:validTil]"/>
  <sch:param name="primaryDate" value="@irm:validTil"/>
  <sch:param name="operator" value="'>='"/>
  <sch:param name="secondaryDateList"
    value="(@irm:created, @irm:posted, @irm:infoCutOff, @irm:approvedOn)"/>
  <sch:param name="flag" value="'warning'"/>
</sch:pattern>
```

2.12 - ./Rules/IRM_ID_00022.sch

Rule Description

[IRM-ID-00022][Error] For any element irm:temporalCoverage, child element irm:start must not be later than child element irm: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 - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00022" is-a="CompareDateTimes">
  <sch:param name="ruleText"
    value="" [IRM-ID-00022][Error] For any element irm:temporalCoverage, child element irm:start must not be later than child element irm: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="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage//irm:temporalCoverage"/>
  <sch:param name="primaryDate" value="irm:start"/>
  <sch:param name="operator" value="'&lt;='"/>
  <sch:param name="secondaryDateList" value="(irm:end)"/>
  <sch:param name="flag" value="'error'"/>
</sch:pattern>
```

2.13 - ../Rules/IRM_ID_00023.sch

Rule Description

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

Code Description

This pattern uses abstract rules to consolidate logic. For attributes, ensure that each date contained within \$dateList has a year value within the range \$minYear and \$maxYear, inclusive.

Schematron Code

```
<?ICEA pattern?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00023"><!-- Use abstract rule to handle required attributes -->

<sch:rule id="IRM-ID-00023-R1"
      context="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage//irm:start | tdf:*/tdf:Assertion/tdf:StructuredStatement/
irm:ICResourceMetadataPackage//irm:end">
      <sch:let name="minYear" value="0001"/>
      <sch:let name="maxYear" value="9999"/>
      <sch:let name="dateList" value="string(.)"/>
      <sch:let name="errMsg"
            value="' [IRM-ID-00023][Error] The permissible values for the year range are 0001 through 9999 for elements irm:start and irm:end. Human Readable: irm:start
and irm:end must be positive integers less than 10,000. '"/>
      <sch:extends rule="abs.dateListYearRangeRule"/>
    </sch:rule>
  </sch:pattern>
```

2.14 - `./Rules/IRM_ID_00024.sch`

Rule Description

[IRM-ID-00024][Warning] For elements `irm:start` and `irm:end` and attributes `@irm:approvedOn`, `@irm:dateProcessed`, `@irm:receivedOn`, `@irm:infoCutOff`, `@irm:posted`, `@irm:validTil`, and `@irm: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 `irm:start` and `irm:end` elements, as well as any element that contains one or more attributes `@irm:approvedOn`, `@irm:infoCutOff`, `@irm:posted`, and `@irm: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 - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<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"
                role="warning">[IRM-ID-00024][Warning] For elements and attributes of date-time types, if the time designator (T) is specified, it is recommended that time
zone be specified.</sch:assert>
    </sch:rule>
    <!-- Begin using abstract rule on required elements and attributes -->

<sch:rule id="IRM-ID-00024-R2"
          context="/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage//irm:start">
    <sch:let name="dateList" value="."/>
    <sch:extends rule="abs.rule00024"/>
</sch:rule>
<sch:rule id="IRM-ID-00024-R3"
          context="/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage//irm:end">
    <sch:let name="dateList" value="."/>
    <sch:extends rule="abs.rule00024"/>
</sch:rule>
<sch:rule id="IRM-ID-00024-R4"
          context="/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage/* [@irm:approvedOn | @irm:dateProcessed | @irm:receivedOn | @irm:posted |
@irm:created | @irm:infoCutOff | @irm:validTil]">
    <sch:let name="dateList"
            value="(@irm:approvedOn, @irm:dateProcessed, @irm:receivedOn, @irm:posted, @irm:created, @irm:infoCutOff, @irm:validTil)"/>
    <sch:extends rule="abs.rule00024"/>
</sch:rule>
</sch:pattern>
```

2.15 - ./Rules/IRM_ID_00025.sch

Rule Description

[IRM-ID-00025][Error] The attribute @ism:excludeFromRollup must not be specified for any element in the namespace [urn:us:gov:ic:irm] in a TDF IRM assertion. Human readable: Elements in IRM TDF assertions are not allowed to be excluded from roll-up because the security markings are now in the TDF assertion statement metadata.

Code Description

The attribute @ism:excludeFromRollup must not be specified for any element in the namespace [urn:us:gov:ic:irm] in a TDF IRM assertion.

Schematron Code

```
<?ICEA pattern?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00025">
    <sch:rule id="IRM-ID-00025-R1"
        context="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage//irm:*">
        <sch:assert test="not(@ism:excludeFromRollup)" flag="error" role="error">[IRM-ID-00025][Error]The attribute @ism:excludeFromRollup must not be specified for any element
in the namespace [urn:us:gov:ic:irm] in a TDF IRM assertion. Human readable: Elements in IRM TDF assertions are not allowed to be excluded from roll-up because the security markings are now in
the TDF assertion statement metadata.</sch:assert>
        </sch:rule>
    </sch:pattern>
```


2.16 - ../Rules/IRM_ID_00029.sch

Rule Description

[IRM-ID-00029][Error] If element irm:geospatialCoverage has attribute @irm:precedence with a value of [Secondary], there must be at least one sibling element irm:geospatialCoverage for which attribute @irm: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], 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 - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00029">
    <sch:rule id="IRM-ID-00029-R1"
        context="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage//irm:geospatialCoverage[@irm:precedence = 'Secondary']">
        <sch:assert test="../irm:geospatialCoverage[@irm:precedence = 'Primary']"
            flag="error"
            role="error">[IRM-ID-00029][Error] If element irm:geospatialCoverage has attribute @irm:precedence with a value of [Secondary], there must be at least one
sibling element irm:geospatialCoverage for which attribute @irm:precedence has a value of [Primary]. Human Readable: If a secondary country code is provided, there must also be a primary
country code.</sch:assert>
        </sch:rule>
    </sch:pattern>
```

2.17 - ../Rules/IRM_ID_00030.sch

Rule Description

[IRM-ID-00030][Error] If attribute @irm:order is specified with integer value N, there must exist other @irm:order attributes with values 1 to N-1 with no duplicates. Human Readable: The values of attribute @irm: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 return false because a duplicate exists. Otherwise, ensure 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 return true. Otherwise, false.

Schematron Code

```
<?ICEA pattern?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00030">
    <sch:rule id="IRM-ID-00030-R1"
        context="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadadataPackage[//@irm:order]">
        <sch:let name="orderList"
            value="tokenize(string-join(//@irm: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"
            role="error">[IRM-ID-00030][Error] If attribute @irm:order is specified with integer value N, there must exist other @irm:order attributes with values 1 to
N-1 with no duplicates. Human Readable: The values of attribute @irm:order must be numbered sequentially with no duplicates, beginning at 1.</sch:assert>
        </sch:rule>
    </sch:pattern>
```

2.18 - ../Rules/IRM_ID_00033.sch

Rule Description

[IRM-ID-00033][Error] If element irm:mimeType is specified, it must match any explicit values or the media type wildcard regex from CVEnumMIMEType.xml.

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 irm:mimeType is specified, it must match any explicit values or the media type wildcard regex from CVEnumMIMEType.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 $list is a variable defined in the main document (IRM_XML.sch), which reads values from a
specific CVE file. '"/>
  <sch:param name="context"
    value="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage[$IC_COMPLIANCE]//irm:mimeType"/>
  <sch:param name="searchTerm" value="."/>
  <sch:param name="list" value="$mimeTypeList"/>
  <sch:param name="errMsg"
    value="" [IRM-ID-00033][Error] If element irm:mimeType is specified, it must match any explicit values or the media type wildcard regex from CVEnumMIMEType.xml.
'"/>
  </sch:pattern>
```

2.19 - ../Rules/IRM_ID_00034.sch

Rule Description

[IRM-ID-00034][Error] For element irm:language, attribute irm: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 irm:language, attribute irm: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 $list is a variable defined in the main document (IRM_XML.sch), which reads values from a
specific CVE file. '"/>
  <sch:param name="context"
    value="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage[$IC_COMPLIANCE]//irm:language"/>
  <sch:param name="searchTerm" value="@irm:qualifier"/>
  <sch:param name="list" value="$compoundLanguageQualifierTypeList"/>
  <sch:param name="errMsg"
    value="" [IRM-ID-00034][Error] For element irm:language, attribute irm:qualifier must have a value in CVEnumIRMCompoundLanguageQualifierType.xml. '"/>
</sch:pattern>
```

2.20 - ../Rules/IRM_ID_00036.sch

Rule Description

[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 - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00036">
    <sch:rule id="IRM-ID-00036-R1"
        context="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage//*[xlink:*]">
        <sch:assert test="normalize-space(string(@xlink:type)) or normalize-space(string(@xlink:href))"
            flag="error"
            role="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. Human Readable: If any XLink attributes are specified for an element, then the type and/or URL of the link must also be specified.</sch:assert>
        </sch:rule>
    </sch:pattern>
```

2.21 - .//Rules/IRM_ID_00040.sch

Rule Description

[IRM-ID-00040][Error] If element irm:type is specified with a qualifier of [urn:us:gov:ic:cenum:irm:activity], then attribute ism:classification must also be specified. Human Readable: If the type element has the qualifier attribute with the value 'urn:us:gov:ic:cenum:irm:activity' then the type element must also have the classification attribute that is not empty.

Code Description

For each irm:type element which specifies attribute irm:qualifier with a value of [urn:us:gov:ic:cenum:irm:activity], ensure that attribute ism:classification is specified.

Schematron Code

```
<?ICEA pattern?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern is-a="IsmEnforcement" id="IRM-ID-00040">
    <sch:param name="ruleText"
               value="[IRM-ID-00040][Error] If element irm:type is specified with a qualifier of [urn:us:gov:ic:cenum:irm:activity], then attribute ism:classification must
also be specified."/>
    <sch:param name="codeDesc"
               value="' For each irm:type element which specifies attribute irm:qualifier with a value of [urn:us:gov:ic:cenum:irm:activity], ensure that attribute
ism:classification is specified. '"/>
    <sch:param name="context"
               value="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage//irm:type[@irm:qualifier='urn:us:gov:ic:cenum:irm:activity']"/>
    <sch:param name="errMsg"
               value="' [IRM-ID-00040][Error] If element irm:type is specified with a qualifier of [urn:us:gov:ic:cenum:irm:activity], then attribute ism:classification must
also be specified. '"/>
    </sch:pattern>
```

2.22 - ../Rules/IRM_ID_00041.sch

Rule Description

IRM-ID-00041][Error] If element irm:type is specified with a qualifier of [urn:us:gov:ic:cenum:intdis:inteldiscipline], then attribute ism:classification must also be specified. Human Readable: If the type element has the qualifier attribute with the value 'urn:us:gov:ic:cenum:intdis:inteldiscipline' then the type element must also have the classification attribute that is not empty.

Code Description

For each irm:type element which specifies attribute irm:qualifier with a value of [urn:us:gov:ic:cenum:intdis:inteldiscipline], ensure that attribtue ism:classification is specified.

Schematron Code

```
<?ICEA pattern?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern is-a="IsmEnforcement" id="IRM-ID-00041">
    <sch:param name="ruleText"
        value="'[IRM-ID-00041][Error] If element irm:type is specified with a qualifier of [urn:us:gov:ic:cenum:intdis:inteldiscipline], then attribute
ism:classification must also be specified.'"/>
    <sch:param name="codeDesc"
        value="' For each irm:type element which specifies attribute irm:qualifier with a value of [urn:us:gov:ic:cenum:intdis:inteldiscipline], ensure that attribtue
ism:classification is specified.'"/>
    <sch:param name="context"
        value="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage//irm:type[@irm:qualifier='urn:us:gov:ic:cenum:intdis:inteldiscipline']"/>
    <sch:param name="errMsg"
        value="' [IRM-ID-00041][Error] If element irm:type is specified with a qualifier of [urn:us:gov:ic:cenum:intdis:inteldiscipline], then attribute
ism:classification must also be specified. '"/>
</sch:pattern>
```

2.23 - ../Rules/IRM_ID_00042.sch

Rule Description

[IRM-ID-00042][Error] If element irm:type is specified with a qualifier of [urn:us:gov:ic:cenum:intdis:inteldiscipline:component], then attribute ism:classification must also be specified. Human Readable: If the type element has the qualifier attribute with the value 'urn:us:gov:ic:cenum:intdis:inteldiscipline:component' then the type element must also have the classification attribute that is not empty.

Code Description

For each irm:type element which specifies attribute irm:qualifier with a value of [urn:us:gov:ic:cenum:intdis:inteldiscipline:component], ensure that attribtue ism:classification is specified.

Schematron Code

```
<?ICEA pattern?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern is-a="IsmEnforcement" id="IRM-ID-00042">
    <sch:param name="ruleText"
        value="'[IRM-ID-00042][Error] If element irm:type is specified with a qualifier of [urn:us:gov:ic:cenum:intdis:inteldiscipline:component], then attribute
ism:classification must also be specified.'"/>
    <sch:param name="codeDesc"
        value="' For each irm:type element which specifies attribute irm:qualifier with a value of [urn:us:gov:ic:cenum:intdis:inteldiscipline:component], ensure that
attribtue ism:classification is specified.'"/>
    <sch:param name="context"
        value="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage//
irm:type[@irm:qualifier='urn:us:gov:ic:cenum:intdis:inteldiscipline:component']"/>
    <sch:param name="errMsg"
        value="' [IRM-ID-00042][Error] If element irm:type is specified with a qualifier of [urn:us:gov:ic:cenum:intdis:inteldiscipline:component], then attribute
ism:classification must also be specified. '"/>
</sch:pattern>
```


2.24 - ../Rules/IRM_ID_00043.sch

Rule Description

If element `irm:type` is specified with a qualifier of `[urn:us:gov:ic:cenum:intdis:inteldiscipline:component:technique]`, then attribute `ism:classification` must also be specified. Human Readable: If the type element has the qualifier attribute with the value 'urn:us:gov:ic:cenum:intdis:inteldiscipline:component:technique' then the type element must also have the classification attribute that is not empty.

Code Description

For each `irm:type` element which specifies attribute `irm:qualifier` with a value of `[urn:us:gov:ic:cenum:intdis:inteldiscipline:component:technique]`, ensure that attribtue `ism:classification` is specified.

Schematron Code

```
<?ICEA pattern?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern is-a="IsmEnforcement" id="IRM-ID-00043">
    <sch:param name="ruleText"
        value="'[IRM-ID-00043][Error] If element irm:type is specified with a qualifier of [urn:us:gov:ic:cenum:intdis:inteldiscipline:component:technique], then
attribute ism:classification must also be specified.'"/>
    <sch:param name="codeDesc"
        value="' For each irm:type element which specifies attribute irm:qualifier with a value of [urn:us:gov:ic:cenum:intdis:inteldiscipline:component:technique],
ensure that attribtue ism:classification is specified.'"/>
    <sch:param name="context"
        value="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage//
irm:type[@irm:qualifier='urn:us:gov:ic:cenum:intdis:inteldiscipline:component:technique']"/>
    <sch:param name="errMsg"
        value="' [IRM-ID-00043][Error] If element irm:type is specified with a qualifier of [urn:us:gov:ic:cenum:intdis:inteldiscipline:component:technique], then
attribute ism:classification must also be specified. '"/>
</sch:pattern>
```

2.25 - ../Rules/IRM_ID_00044.sch

Rule Description

[IRM-ID-00044][Error] If element irm:type is specified with a qualifier of [urn:us:gov:ic:irm:productline] then attribute ism:classification must also be specified. Human Readable: If the type element has the qualifier attribute with the value 'urn:us:gov:ic:irm:productline' then the type element must also have the classification attribute that is not empty.

Code Description

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

Schematron Code

```
<?ICEA pattern?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern is-a="IsmEnforcement" id="IRM-ID-00044">
    <sch:param name="ruleText"
        value="'[IRM-ID-00044][Error] If element irm: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 irm:type element which specifies attribute irm:qualifier with a value of [urn:us:gov:ic:irm:productline], ensure that attribtue
ism:classification is specified.'"/>
    <sch:param name="context"
        value="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage//irm:type[@irm:qualifier='urn:us:gov:ic:irm:productline']"/>
    <sch:param name="errMsg"
        value="' [IRM-ID-00044][Error] If element irm:type is specified with a qualifier of [urn:us:gov:ic:irm:productline] then attribute ism:classification must also
be specified. '"/>
    </sch:pattern>
```

2.26 - ../Rules/IRM_ID_00045.sch

Rule Description

[IRM-ID-00045][Error] Element irm:geospatialCoverage must have ISM classification markings. Human Readable: The geospatialCoverage element must have a classification attribute.

Code Description

For each irm:geospatialCoverage element, ensure that attribute ism:classification is specified.

Schematron Code

```
<?ICEA pattern?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00045">
    <sch:rule id="IRM-ID-00045-R1"
        context="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage[$IC_COMPLIANCE]//irm:geospatialCoverage">
        <sch:assert test="@ism:classification" flag="error" role="error">[IRM-ID-00045][Error] Element irm:geospatialCoverage must have ISM classification markings. Human
Readable: The geospatialCoverage element must have a classification attribute.</sch:assert>
        </sch:rule>
    </sch:pattern>
```

2.27 - ../Rules/IRM_ID_00046.sch

Rule Description

[IRM-ID-00046][Error] If element irm:type has attribute @irm:qualifier specified as [urn:us:gov:ic:cvenum:intdis:inteldiscipline:component:technique] the attribute @irm:value must be in CVENumIntelDisciplineComponentTechnique.xml. Human Readable: Intel Discipline Component Techniques must be in the CVENumIntelDisciplineComponentTechnique 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 - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern is-a="ValidateValueExistenceInList" id="IRM-ID-00046">
  <sch:param name="ruleText"
    value="" [IRM-ID-00046][Error] If element irm:type has attribute @irm:qualifier specified as [urn:us:gov:ic:cvenum:intdis:inteldiscipline:component:technique]
the attribute @irm:value must be in CVENumIntelDisciplineComponentTechnique.xml. Human Readable: Intel Discipline Component Techniques must be in the CVENumIntelDisciplineComponentTechnique
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 $list is a variable defined in the main document (IRM_XML.sch), which reads values from a
specific CVE file. ""/>
    <sch:param name="context"
      value="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage//
irm:type[@irm:qualifier='urn:us:gov:ic:cvenum:intdis:inteldiscipline:component:technique']"/>
    <sch:param name="searchTerm" value="@irm:value"/>
    <sch:param name="list" value="$intelDisciplineComponentTechniqueList"/>
    <sch:param name="errMsg"
      value="" [IRM-ID-00046][Error] If element irm:type has attribute @irm:qualifier specified as [urn:us:gov:ic:cvenum:intdis:inteldiscipline:component:technique]
the attribute @irm:value must be in CVENumIntelDisciplineComponentTechnique.xml. Human Readable: Intel Discipline Component Techniques must be in the CVENumIntelDisciplineComponentTechnique
CVE. ""/>
  </sch:pattern>
```

2.28 - ../Rules/IRM_ID_00047.sch

Rule Description

[IRM-ID-00047][Error] If element irm:type has attribute @irm:qualifier specified as [urn:us:gov:ic:cvenum:intdis:inteldiscipline:component] the attribute @irm:value must be in CVENumIntelDisciplineComponent.xml. Human Readable: Intel Discipline Components must be in the CVENumIntelDisciplineComponent 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 - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern is-a="ValidateValueExistenceInList" id="IRM-ID-00047">
    <sch:param name="ruleText"
        value="" [IRM-ID-00047][Error] If element irm:type has attribute @irm:qualifier specified as [urn:us:gov:ic:cvenum:intdis:inteldiscipline:component] the
attribute @irm:value must be in CVENumIntelDisciplineComponent.xml. Human Readable: Intel Discipline Components must be in the CVENumIntelDisciplineComponent 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 $list is a variable defined in the main document (IRM_XML.sch), which reads values from a
specific CVE file. '"/>
    <sch:param name="context"
        value="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage//
irm:type[@irm:qualifier='urn:us:gov:ic:cvenum:intdis:inteldiscipline:component']"/>
    <sch:param name="searchTerm" value="@irm:value"/>
    <sch:param name="list" value="$intelDisciplineComponentList"/>
    <sch:param name="errMsg"
        value="" [IRM-ID-00047][Error] If element irm:type has attribute @irm:qualifier specified as [urn:us:gov:ic:cvenum:intdis:inteldiscipline:component] the
attribute @irm:value must be in CVENumIntelDisciplineComponent.xml. Human Readable: Intel Discipline Components must be in the CVENumIntelDisciplineComponent CVE. '"/>
</sch:pattern>
```

2.29 - ../Rules/IRM_ID_00048.sch

Rule Description

[IRM-ID-00048][Error] If element irm:type has attribute @irm:qualifier specified as [urn:us:gov:ic:cvenum:intdis:inteldiscipline] the attribute @irm:value must be in CVEnumIntelDiscipline.xml. Human Readable: Intel Disciplines must be in the CVEnumIntelDiscipline 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 - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern is-a="ValidateValueExistenceInList" id="IRM-ID-00048">
  <sch:param name="ruleText"
    value="" [IRM-ID-00048][Error] If element irm:type has attribute @irm:qualifier specified as [urn:us:gov:ic:cvenum:intdis:inteldiscipline] the attribute
@irm:value must be in CVEnumIntelDiscipline.xml. Human Readable: Intel Disciplines must be in the CVEnumIntelDiscipline 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 $list is a variable defined in the main document (IRM_XML.sch), which reads values from a
specific CVE file. '"/>
  <sch:param name="context"
    value="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage//irm:type[@irm:qualifier='urn:us:gov:ic:cvenum:intdis:inteldiscipline']"/>
  <sch:param name="searchTerm" value="@irm:value"/>
  <sch:param name="list" value="$intelDisciplineList"/>
  <sch:param name="errMsg"
    value="" [IRM-ID-00048][Error] If element irm:type has attribute @irm:qualifier specified as [urn:us:gov:ic:cvenum:intdis:inteldiscipline] the attribute
@irm:value must be in CVEnumIntelDiscipline.xml. Human Readable: Intel Disciplines must be in the CVEnumIntelDiscipline CVE. '"/>
</sch:pattern>
```

2.30 - ../Rules/IRM_ID_00053.sch

Rule Description

[IRM-ID-00053][Error] If element irm:type has attribute @irm:qualifier specified as [urn:us:gov:ic:cvenum:irm:activity] the attribute @irm: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 irm:type has attribute @irm:qualifier specified as [urn:us:gov:ic:cvenum:irm:activity] the attribute @irm:value 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 $list is a variable defined in the main document (IRM_XML.sch), which reads values from a
specific CVE file. '"/>
  <sch:param name="context"
    value="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage//irm:type[@irm:qualifier='urn:us:gov:ic:cvenum:irm:activity']"/>
  <sch:param name="searchTerm" value="@irm:value"/>
  <sch:param name="list" value="$activityList"/>
  <sch:param name="errMsg"
    value="" [IRM-ID-00053][Error] If element irm:type has attribute @irm:qualifier specified as [urn:us:gov:ic:cvenum:irm:activity] the attribute @irm:value must
be in CVENumIRMActivity.xml. Human Readable: Activity must be in the CVENumIRMActivity CVE. '"/>
</sch:pattern>
```

2.31 - ./Rules/IRM_ID_00054.sch

Rule Description

[IRM-ID-00054][Error] If element irm:geospatialCoverage has attribute @irm:precedence specified, then the value must be in CVEnumIRMCoveragePrecedence.xml. Human Readable: Geospatial Coverage Precedence 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 irm:geospatialCoverage has attribute @irm:precedence specified, then the value must be in
CVEnumIRMCoveragePrecedence.xml. Human Readable: Geospatial Coverage Precedence 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 $list is a variable defined in the main document (IRM_XML.sch), which reads values from a
specific CVE file. '"/>
  <sch:param name="context"
    value="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage[$IC_COMPLIANCE]//irm:geospatialCoverage[@irm:precedence]"/>
  <sch:param name="searchTerm" value="@irm:precedence"/>
  <sch:param name="list" value="$coveragePrecedenceList"/>
  <sch:param name="errMsg"
    value="" [IRM-ID-00054][Error] If element irm:geospatialCoverage has attribute @irm:precedence specified, then the value must be in
CVEnumIRMCoveragePrecedence.xml. Human Readable: Geospatial Coverage Precedence must be in the CVEnumIRMCoveragePrecedence CVE. '"/>
</sch:pattern>
```


2.32 - ./Rules/IRM_ID_00055.sch

Rule Description

[IRM-ID-00055][Error] If irm:geospatialCoverage/@order is specified then there must be one and only one of irm:geospatialIdentifier/irm:countryCode or irm:geospatialIdentifier/irm: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 irm:countryCode or order irm:subDivisionCode when irm:geospatialCoverage uses the order attribute.

Schematron Code

```
<?ICEA pattern?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00055">
  <sch:rule id="IRM-ID-00055-R1"
    context="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage[$IC_COMPLIANCE]//irm:geospatialCoverage[@irm:order]">
    <sch:assert id="IRM-00055"
      test="count(irm:geographicIdentifier/irm:countryCode) + count(irm:geographicIdentifier/irm:subDivisionCode) = 1"
      flag="error"
      role="error">[IRM-ID-00055][Error] If irm:geospatialCoverage/@order is specified then there must be one and only one of irm:geospatialIdentifier/
irm:countryCode or irm:geospatialIdentifier/irm:subDivisionCode. Human Readable: A single order value must be applied to one country code or one subdivision code</sch:assert>
    </sch:rule>
  </sch:pattern>
```

2.33 - ./Rules/IRM_ID_00062.sch

Rule Description

[IRM-ID-00062][Error] The value of an IC-ID identifier must follow standardized convention. Human Readable: The IC-ID identifier value has to follow standardized convention.

Code Description

This rule uses an abstract pattern that contains the logic for ensuring the value found if a given context exists, both provided as parameters from this implementation, follows the IC-ID identifier standardized convention.

Schematron Code

```
<?ICEA pattern?>
<?ICEA min_accessible?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00062" is-a="ICIdentifierRestrictions">
    <sch:param name="context"
               value="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage//irm:identifier[@irm:qualifier='IC-ID']"/>
    <sch:param name="value" value="string(@irm:value)"/>
    <sch:param name="errorMessage"
               value="'[IRM-ID-00062][Error] The value of an IC-ID identifier must follow standardized convention. Human Readable: The IC-ID identifier value has to follow
standardized convention.'"/>
</sch:pattern>
```

2.34 - ../Rules/IRM_ID_00063.sch

Rule Description

[IRM-ID-00063][Error] Element irm:ICResourceMetadataPackage/irm:creator/irm:organization must specify attribute @irm:acronym. Human Readable: The IRM card must specify a creator organization with an IC agency acronym for the referenced resource.

Code Description

Make sure that element irm:ICResourceMetadataPackage/irm:creator/irm:organization exists and specifies attribute @irm:acronym.

Schematron Code

```
<?ICEA pattern?>
<?ICEA min_accessible?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00063">
    <sch:rule id="IRM-ID-00063-R1"
        context="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage[$IC_COMPLIANCE]">
        <sch:assert test="irm:creator/irm:organization/@irm:acronym"
            flag="error"
            role="error">[IRM-ID-00063][Error] Element irm:ICResourceMetadataPackage/irm:creator/irm:organization must specify attribute @irm:acronym. Human Readable:
The IRM card must specify a creator organization with an IC agency acronym for the referenced resource.</sch:assert>
        </sch:rule>
    </sch:pattern>
```

2.35 - ./Rules/IRM_ID_00064.sch

Rule Description

[IRM-ID-00064][Error] Element irm:ICResourceMetadataPackage/irm:dates must specify attribute @irm:created. Human Readable: The IRM card must specify the date on which the referenced resource was created.

Code Description

Make sure that element irm:ICResourceMetadataPackage/irm:dates exists and specifies attribute @irm:created.

Schematron Code

```
<?ICEA pattern?>
<?ICEA min_accessible?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00064">
    <sch:rule id="IRM-ID-00064-R1"
              context="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage[$IC_COMPLIANCE]">
        <sch:assert test="irm:dates/@irm:created" flag="error" role="error">[IRM-ID-00064][Error] Element irm:ICResourceMetadataPackage/irm:dates must specify attribute
@irm:created. Human Readable: The IRM card must specify the date on which the referenced resource was created.</sch:assert>
        </sch:rule>
    </sch:pattern>
```

2.36 - ../Rules/IRM_ID_00065.sch

Rule Description

[IRM-ID-00065][Error] Attribute irm:ICResourceMetadataPackage/irm:dates/@irm:created must be castable as an xs:dateTime type. Human Readable: The date on which the referenced resource was created must be a dateTime type.

Code Description

Make sure that attribute dms:resource/irm:dates/@irm:created is castable as an xs:dateTime type.

Schematron Code

```
<?ICEA pattern?>
<?ICEA min_accessible?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00065">
    <sch:rule id="IRM-ID-00065-R1"
        context="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage/irm:dates[@irm:created]">
        <sch:assert test="@irm:created castable as xs:dateTime"
            flag="error"
            role="error">[IRM-ID-00065][Error] Attribute irm:ICResourceMetadataPackage/irm:dates/@irm:created must be castable as an xs:dateTime type. Human Readable:
The date on which the referenced resource was created must be a dateTime type.</sch:assert>
        </sch:rule>
    </sch:pattern>
```

2.37 - ../Rules/IRM_ID_00068.sch

Rule Description

[IRM-ID-00068][Error] For element irm:taskID, if attribute xlink:href exists, then the attribute must have a non-null value. Human Readable:

Code Description

The normalize-spaced value of attribute xlink:href is checked to make sure the length of the resulting string is greater than zero, which indicates non-whitespace content.

Schematron Code

```
<?ICEA pattern?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00068">
    <sch:rule id="IRM-ID-00068-R1"
        context="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage//irm:taskID[@xlink:href]">
        <sch:assert test="normalize-space(string(@xlink:href))"
            flag="error"
            role="error">[IRM-ID-00068][Error] For element irm:taskID if attribute xlink:href exists, then the attribute must have a non-null value. Human Readable:</
sch:assert>
        </sch:rule>
    </sch:pattern>
```

2.38 - ../Rules/IRM_ID_00070.sch

Rule Description

[IRM-ID-00070][Error] If element irm:type has attribute @irm:qualifier specified as [urn:us:gov:ic:cvenum:irm:executableindicator] the attribute @irm:value must be in CVENumIRMExecutableIndicator.xml. Human Readable: Executable Indicator Value must be in the CVENumIRMExecutableIndicator 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-00070">
  <sch:param name="ruleText"
    value="" [IRM-ID-00070][Error] If element irm:type has attribute @irm:qualifier specified as [urn:us:gov:ic:cvenum:irm:executableindicator] the attribute
@irm:value must be in CVENumIRMExecutableIndicator.xml. Human Readable: Executable Indicator Value must be in the CVENumIRMExecutableIndicator 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 $list is a variable defined in the main document (IRM_XML.sch), which reads values from a
specific CVE file. '"/>
  <sch:param name="context"
    value="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage//irm:type[@irm:qualifier='urn:us:gov:ic:cvenum:irm:executableindicator']"/>
  <sch:param name="searchTerm" value="@irm:value"/>
  <sch:param name="list" value="$executableIndicatorList"/>
  <sch:param name="errMsg"
    value="" [IRM-ID-00070][Error] If element irm:type has attribute @irm:qualifier specified as [urn:us:gov:ic:cvenum:irm:executableindicator] the attribute
@irm:value must be in CVENumIRMExecutableIndicator.xml. HHuman Readable: Executable Indicator Value must be in the CVENumIRMExecutableIndicator CVE. '"/>
</sch:pattern>
```

2.39 - ../Rules/IRM_ID_00071.sch

Rule Description

[IRM-ID-00071][Error] If element irm:type has attribute @irm:qualifier specified as [urn:us:gov:ic:cvenum:irm:maliciouscodeindicator] the attribute @irm:value must be in CVENumIRMMaliciousCodeIndicator.xml. Human Readable: Malicious Code Indicator values must be in the CVENumIRMMaliciousCodeIndicator 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-00071">
  <sch:param name="ruleText"
    value="" [IRM-ID-00071][Error] If element irm:type has attribute @irm:qualifier specified as [urn:us:gov:ic:cvenum:irm:maliciouscodeindicator] the attribute
@irm:value must be in CVENumIRMMaliciousCodeIndicator.xml. Human Readable: Malicious Code Indicator values must be in the CVENumIRMMaliciousCodeIndicator 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 $list is a variable defined in the main document (IRM_XML.sch), which reads values from a
specific CVE file. ""/>
  <sch:param name="context"
    value="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage//irm:type[@irm:qualifier='urn:us:gov:ic:cvenum:irm:maliciouscodeindicator']"/>
  <sch:param name="searchTerm" value="@irm:value"/>
  <sch:param name="list" value="$maliciousCodeIndicatorList"/>
  <sch:param name="errMsg"
    value="" [IRM-ID-00071][Error] If element irm:type has attribute @irm:qualifier specified as [urn:us:gov:ic:cvenum:irm:maliciouscodeindicator] the attribute
@irm:value must be in CVENumIRMMaliciousCodeIndicator.xml. Human Readable: Malicious Code Indicator values must be in the CVENumIRMMaliciousCodeIndicator CVE. ""/>
</sch:pattern>
```


2.40 - ../Rules/IRM_ID_00072.sch

Rule Description

[IRM-ID-00072][Error] For element irm:searchableDate, irm:start must be earlier than irm:end. Human Readable: Within the searchableDate element, the date within the start element must be earlier than the date within the end element.

Code Description

For each element irm:searchableDate, the child elements irm:start and irm:end are cast as xs:dateTime types, then compared to ensure start is less than end.

Schematron Code

```
<?ICEA pattern?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00072">
  <sch:rule id="IRM-ID-00072-R1"
    context="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage//irm:searchableDate">
    <sch:assert test="if((irm:start castable as xs:dateTime) and (irm:end castable as xs:dateTime)) then xs:dateTime(irm:start) < xs:dateTime(irm:end) else false()"
      flag="error"
      role="error">[IRM-ID-00072][Error] For element irm:searchableDate, irm:start must be earlier than irm:end. Human Readable: Within the searchableDate
element, the date within the start element must be earlier than the date within the end element.</sch:assert>
    </sch:rule>
  </sch:pattern>
```

2.41 - ../Rules/IRM_ID_00073.sch

Rule Description

[IRM-ID-00073][Error] If element irm:type has attribute @irm:qualifier specified as [urn:us:gov:ic:cvenum:irm:positive:intel] the attribute @irm:value must be in CVEnumIRMPositiveIntel.xml. Human Readable: Positive Intel values must be in the CVEnumIRMPositiveIntel 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 - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern is-a="ValidateValueExistenceInList" id="IRM-ID-00073">
    <sch:param name="ruleText"
        value="" [IRM-ID-00073][Error] If element irm:type has attribute @irm:qualifier specified as [urn:us:gov:ic:cvenum:irm:positive:intel] the attribute @irm:value
must be in CVEnumIRMPositiveIntel.xml. Human Readable: Positive Intel values must be in the CVEnumIRMPositiveIntel 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 $list is a variable defined in the main document (IRM_XML.sch), which reads values from a
specific CVE file. ""/>
    <sch:param name="context"
        value="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage//irm:type[@irm:qualifier='urn:us:gov:ic:cvenum:irm:positive:intel']"/>
    <sch:param name="searchTerm" value="@irm:value"/>
    <sch:param name="list" value="$positiveIntelList"/>
    <sch:param name="errMsg"
        value="" [IRM-ID-00073][Error] If element irm:type has attribute @irm:qualifier specified as [urn:us:gov:ic:cvenum:irm:positive:intel] the attribute @irm:value
must be in CVEnumIRMPositiveIntel.xml. Human Readable: Positive Intel values must be in the CVEnumIRMPositiveIntel CVE. ""/>
</sch:pattern>
```

2.42 - ../Rules/IRM_ID_00074.sch

Rule Description

[IRM-ID-00074][Error] For element irm:searchableDate, elements irm:start and irm:end must match the xsd:dateTime format. Human Readable: Within the searchableDate element, the start and end elements values must conform to the xsd:dateTime format.

Code Description

For each element irm:searchableDate, ensure that elements irm:start and irm:end are each castable as xs:dateTime type.

Schematron Code

```
<?ICEA pattern?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00074">
    <sch:rule id="IRM-ID-00074-R1"
        context="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage//irm:searchableDate">
        <sch:assert test="(irm:start castable as xs:dateTime) and (irm:end castable as xs:dateTime)"
            flag="error"
            role="error">[IRM-ID-00074][Error] For element irm:searchableDate, elements irm:start and irm:end must match the xsd:dateTime format. Human Readable: Within
the searchableDate element, the start and end elements values must conform to the xsd:dateTime format.</sch:assert>
        </sch:rule>
    </sch:pattern>
```

2.43 - ../Rules/IRM_ID_00076.sch

Rule Description

[IRM-ID-00076][Error] If the irm:acquiredOn element exists, at least one of its child elements irm:description, irm:approximableDate, or irm:searchableDate must be present. Human Readable: The acquiredOn element must have at least one of the following child elements: description, approximableDate and searchableDate.

Code Description

For element irm:acquiredOn, ensure that one or more of the child elements irm:description, irm:approximableDate, irm:searchableDate/irm:start, or irm:searchableDate/irm:end is specified with non-whitespace content.

Schematron Code

```
<?ICEA pattern?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00076">
    <sch:rule id="IRM-ID-00076-R1"
        context="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage//irm:acquiredOn">
        <sch:assert test="normalize-space(string(irm:description)) or normalize-space(string(irm:approximableDate)) or normalize-space(string(irm:searchableDate/irm:start)) or
normalize-space(string(irm:searchableDate/irm:end))"
            flag="error"
            role="error">[IRM-ID-00076][Error] If the irm:acquiredOn element exists, at least one of its child elements irm:description, irm:approximableDate, or
irm:searchableDate must be present. Human Readable: The acquiredOn element must have at least one of the following child elements: description, approximableDate and searchableDate.</sch:assert>
        </sch:rule>
    </sch:pattern>
```

2.44 - ../Rules/IRM_ID_00077.sch

Rule Description

[IRM-ID-00077][Error] For element irm:person at least one of the following child elements must have non-whitespace content: irm:surname, irm:userID, irm:name, irm:affiliation, irm:postalAddress, irm:phone irm:email.

Code Description

This pattern uses an abstract rule to consolidate logic. It normalizes the space of the value of the specified child elements and makes sure that the length of the resulting string is greater than zero, which indicates non-whitespace content. Element irm:postalAddress cannot contain text content, so count the number of its child elements that contain non-white space and make sure that the count is great than 0. The abstract rule is extended once for each required element in rule IRM_ID_00077.

Schematron Code

```
<?ICEA pattern?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00077"><!-- Abstract rule, which asserts that at least one of the listed child elements has non-whitespace content -->

<sch:rule abstract="true" id="abs.rule00077">
    <sch:assert test="irm:surname[normalize-space(string(text()))] or irm:userID[normalize-space(string(text()))] or irm:name[normalize-space(string(text()))] or
irm:affiliation[normalize-space(string(text()))] or irm:phone[normalize-space(string(text()))] or irm:email[normalize-space(string(text()))] or (some $token in ancestor::irm:postalAddress/*/
text() satisfies normalize-space(string($token)))"
        flag="error"
        role="error">[IRM-ID-00077][Error] For element irm:person at least one of the following child elements must have non-whitespace content: irm:surname,
irm:userID, irm:name, irm:affiliation, irm:postalAddress, irm:phone irm:email.</sch:assert>
    </sch:rule>
    <!-- Begin using abstract rule to check required elements -->

<sch:rule id="IRM-ID-00077-R2"
    context="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage//irm:person">
    <sch:extends rule="abs.rule00077"/>
</sch:rule>
</sch:pattern>
```

2.45 - `./Rules/IRM_ID_00078.sch`

Rule Description

[IRM-ID-00078][Error] For elements `irm:acquiredOn` and `irm:temporalCoverage` with child element name `[infoCutoff]`, the permissible values for the year range are 1901 through the current year for elements `irm:approximableDate`, `irm:start`, and `irm:end`. Human Readable: If elements `acquiredOn` and `temporalCoverage` have a child element `infoCutoff`, then the `approximableDate`, `start` and `end` elements must have a year value between 1901 and the current year.

Code Description

This pattern uses an abstract rule to consolidate logic. It makes sure that the date contained within `$dateValue` has a year value within the range `$minYear` and `$maxYear`, inclusive. The abstract rule is extended once for each element required in rule IRM-ID-00078.

Schematron Code

```
<?ICEA pattern?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00078">
  <sch:rule abstract="true" id="abs.rule00078">
    <sch:let name="minYear" value="1901"/>
    <sch:let name="maxYear" value="$currentYear"/>
    <sch:let name="dateValue" value="."/>
    <sch:let name="errMsg"
      value="' [IRM-ID-00078][Error] For elements irm:acquiredOn and irm:temporalCoverage with child element name [infoCutoff], the permissible values for the year
range are 1901 through the current year for elements irm:approximableDate, irm:start, and irm:end. '"/>
    <sch:extends rule="abs.dateYearRangeRule"/>
  </sch:rule>
  <!-- Begin using abstract rule to check required elements -->

  <sch:rule id="IRM-ID-00078-R2"
    context="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage//irm:temporalCoverage[irm:name='infoCutOff']/irm:approximableStart/
irm:searchableDate/irm:start">
    <sch:extends rule="abs.rule00078"/>
  </sch:rule>
  <sch:rule id="IRM-ID-00078-R3"
    context="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage//irm:temporalCoverage[irm:name='infoCutOff']/irm:approximableStart/
irm:searchableDate/irm:end">
    <sch:extends rule="abs.rule00078"/>
  </sch:rule>
  <sch:rule id="IRM-ID-00078-R4"
    context="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage//irm:temporalCoverage[irm:name='infoCutOff']/irm:approximableEnd/
irm:searchableDate/irm:start">
    <sch:extends rule="abs.rule00078"/>
  </sch:rule>
  <sch:rule id="IRM-ID-00078-R5"
    context="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage//irm:temporalCoverage[irm:name='infoCutOff']/irm:approximableEnd/
irm:searchableDate/irm:end">
    <sch:extends rule="abs.rule00078"/>
  </sch:rule>
  <sch:rule id="IRM-ID-00078-R6"
    context="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage//irm:temporalCoverage[irm:name='infoCutOff']/irm:start">
    <sch:extends rule="abs.rule00078"/>
  </sch:rule>
  <sch:rule id="IRM-ID-00078-R7"
    context="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage//irm:temporalCoverage[irm:name='infoCutOff']/irm:end">
    <sch:extends rule="abs.rule00078"/>
  </sch:rule>
  <sch:rule id="IRM-ID-00078-R8"
    context="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage//irm:acquiredOn/irm:approximableDate">
    <sch:extends rule="abs.rule00078"/>
  </sch:rule>
  <sch:rule id="IRM-ID-00078-R9"
    context="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage//irm:acquiredOn/irm:searchableDate/irm:start">
```

```
        <sch:extends rule="abs.rule00078"/>
    </sch:rule>
    <sch:rule id="IRM-ID-00078-R10"
        context="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage//irm:acquiredOn/irm:searchableDate/irm:end">
        <sch:extends rule="abs.rule00078"/>
    </sch:rule>
</sch:pattern>
```


2.46 - ../Rules/IRM_ID_00079.sch

Rule Description

[IRM-ID-00079][Error] Regardless of the version indicated on the instance document, the validation infrastructure MUST use a version of 'IC-ID' that is version '1' or later. NOTE: This is not an error of the instance document but of the validation environment itself.

Code Description

This rule uses an abstract pattern to consolidate logic. It verifies that the validation infrastructure is using the version specified in parameters.

Schematron Code

```
<?ICEA pattern?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00079" is-a="ValidateValidationEnvSchema">
    <sch:param name="MinVersion" value="'1'"/>
    <sch:param name="SpecToCheck" value="'IC-ID'"/>
    <sch:param name="pathToDocument" value="'../Schema/IC-ID/IC-ID.xsd'"/>
    <sch:param name="RuleID" value="'IRM-ID-00079'"/>
</sch:pattern>
```

2.47 - ../Rules/IRM_ID_00080.sch

Rule Description

[IRM-ID-00080][Error] Regardless of the version indicated on the instance document, the validation infrastructure MUST use a version of 'USAgency' that is version '201703.201802' (Version:2017-MAR Revision:2018-FEB) or later. NOTE: This is not an error of the instance document but of the validation environment itself.

Code Description

This rule uses an abstract pattern to consolidate logic. It verifies that the validation infrastructure is using the version specified in parameters.

Schematron Code

```
<?ICEA pattern?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00080" is-a="ValidateValidationEnvCVE">
    <sch:param name="MinVersion" value="'201703.201802'"/>
    <sch:param name="SpecToCheck" value="'USAgency'"/>
    <sch:param name="pathToDocument"
        value="'../../../../CVE/USAgency/CVEnumUSAgencyAcronym.xml'"/>
    <sch:param name="RuleID" value="'IRM-ID-00080'"/>
</sch:pattern>
```

2.48 - ../Rules/IRM_ID_00081.sch

Rule Description

[IRM-ID-00081][Error] A irm:type element with @irm:qualifer ProductLine or Intel must not contain any text. Human Readable: IRM Types of ProductLine or Intel must not contain any text within the element.

Code Description

For all irm:type elements that contain a @irm:qualifier of urn:us:gov:ic:irm:productline or that start with urn:us:gov:ic:cvenum:intdis, verify that the element does not contain any text.

Schematron Code

```
<?ICEA pattern?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00081">
    <sch:rule id="IRM-ID-00081-R1"
        context="irm:type[$IC_COMPLIANCE][@irm:qualifier = 'urn:us:gov:ic:irm:productline'] | irm:type[$IC_COMPLIANCE][contains(@irm:qualifier,
'urn:us:gov:ic:cvenum:intdis')]">
        <sch:assert test="not(normalize-space(text()))" flag="error" role="error">[IRM-ID-00081][Error] A irm:type element with @irm:qualifer ProductLine or Intel must not
contain any text. Human Readable: IRM Types of ProductLine or Intel must not contain any text within the element.</sch:assert>
    </sch:rule>
</sch:pattern>
```

2.49 - ../Rules/IRM_ID_00086.sch

Rule Description

[IRM-ID-00086][Error] Regardless of the version indicated on the instance document, the validation infrastructure MUST use a version of 'INTDIS' that is version '201707' (Version:2017-JUL) or later. NOTE: This is not an error of the instance document but of the validation environment itself.

Code Description

This rule uses an abstract pattern to consolidate logic. It verifies that the validation infrastructure is using the version specified in parameters.

Schematron Code

```
<?ICEA pattern?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00086" is-a="ValidateValidationEnvCVE">
    <sch:param name="MinVersion" value="'201707'"/>
    <sch:param name="SpecToCheck" value="'INTDIS'"/>
    <sch:param name="pathToDocument"
        value="'../../CVE/INTDIS/CVEnumIntelDiscipline.xml'"/>
    <sch:param name="RuleID" value="'IRM-ID-00086'"/>
</sch:pattern>
```

2.50 - ../Rules/IRM_ID_00087.sch

Rule Description

[IRM-ID-00087][Error] If @irm:qualifier identifies a intelligence discipline URN, then @intdis:CESVersion must exist as well.

Code Description

Make sure that the INTDIS CVE version attribute exists if intelligence discipline resources are identified.

Schematron Code

```
<?ICEA pattern?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00087">
    <sch:rule id="IRM-ID-00087-R1"
        context="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage//irm:type[starts-
with(@irm:qualifier,'urn:us:gov:ic:cvenum:intdis:intelldiscipline')]">
        <sch:assert test="ancestor-or-self::tdf:*/@intdis:CESVersion"
            flag="error"
            role="error">[IRM-ID-00087][Error] If @irm:qualifier identifies a intelligence discipline URN, then @intdis:CESVersion must exist as well.</sch:assert>
        </sch:rule>
    </sch:pattern>
```

2.51 - ../Rules/IRM_ID_00088.sch

Rule Description

[IRM-ID-00088][Error] Regardless of the version indicated on the instance document, the validation infrastructure MUST use a version of 'MIME' that is version '202010' (Version:2020-OCT) or later. NOTE: This is not an error of the instance document but of the validation environment itself.

Code Description

This rule uses an abstract pattern to consolidate logic. It verifies that the validation infrastructure is using the version specified in parameters.

Schematron Code

```
<?ICEA pattern?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00088" is-a="ValidateValidationEnvCVE">
    <sch:param name="MinVersion" value="'202010'"/>
    <sch:param name="SpecToCheck" value="'MIME'"/>
    <sch:param name="pathToDocument" value="'../CVE/MIME/CVEnumMIMEType.xml'"/>
    <sch:param name="RuleID" value="'IRM-ID-00088'"/>
</sch:pattern>
```

2.52 - ../Rules/IRM_ID_00089.sch

Rule Description

[IRM-ID-00089][Error] If @irm:mimeType exists, then @mime:CESVersion must exist as well.

Code Description

Make sure that the MIME CVE version attribute exists if the internet media type of the resource is defined.

Schematron Code

```
<?ICEA pattern?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00089">
    <sch:rule id="IRM-ID-00089-R1"
        context="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage//irm:mimeType">
        <sch:assert test="//@mime:CESVersion" flag="error" role="error">[IRM-ID-00089][Error] If @irm:mimeType exists, then @mime:CESVersion must exist as well.</sch:assert>
    </sch:rule>
</sch:pattern>
```

2.53 - ../Rules/IRM_ID_00090.sch

Rule Description

[IRM-ID-00090][Error] If subDivisionCode codespace is GENC codespace, then value must be in the GENC subDivisionCode cve.

Code Description

If subDivisionCode codespace is GENC codespace, then value must be in the GENC subDivisionCode cve.

Schematron Code

```
<sch:pattern id="IRM-ID-00090">
  <sch:rule id="IRM-ID-00090-R1"
    context="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage//irm:geospatialCoverage//irm:subDivisionCode">
    <sch:let name="isGENCSubDivision"
      value="matches(normalize-space(./@irm:codespace), '^as:GENC:6:(ed3|3-[1-9][0-9]*)$')"/>
    <sch:assert test="not($isGENCSubDivision) or (some $token in $gencSubDivisionList satisfies $token = normalize-space(./@irm:code))"
      flag="error"
      role="error">[IRM-ID-00090][Error] If subDivisionCode codespace is GENC codespace, then value must be in the GENC subDivisionCode cve.</sch:assert>
    </sch:rule>
  </sch:pattern>
```


2.54 - ../Rules/IRM_ID_00091.sch

Rule Description

[IRM-ID-00091][Warning] Deprecated MIME types should not be used.

Code Description

Deprecated MIME types should not be used.

Schematron Code

```
<?ICEA pattern?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00091">
    <sch:rule id="IRM-ID-00091-R1"
        context="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage//irm:mimeType">
        <sch:assert test="not(some $deprecatedMimeTypeValue in $deprecatedMimeTypeList satisfies . = $deprecatedMimeTypeValue)"
            flag="error"
            role="error">[IRM-ID-00091][Warning] Deprecated MIME types should not be used.</sch:assert>
    </sch:rule>
</sch:pattern>
```

2.55 - ../Rules/IRM_ID_00092.sch

Rule Description

[IRM-ID-00092][Error] irm:NonStateActor should contain a value from the NonStateActor CVE

Code Description

Make sure that the value within NonStateActor is a value from the CVE.

Schematron Code

```
<sch:pattern id="IRM-ID-00092">
  <sch:rule id="IRM-ID-00092-R1"
    context="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage//irm:nonStateActor[@irm:qualifier='urn:us:gov:ic:cvenum:pm:nonstateactors']">
    <sch:assert test="some $token in $nonStateActorsList satisfies $token = normalize-space(./text())"
      flag="error"
      role="error">[IRM-ID-00092][Error] irm:NonStateActor should contain a value from the NonStateActor CVE</sch:assert>
    </sch:rule>
  </sch:pattern>
```

2.56 - ../Rules/IRM_ID_00093.sch

Rule Description

[IRM-ID-00093][Error] If present, at least one irm:countryCode in a irm:geographicIdentifier must be a GENC ED3 codespace: ^ge:GENC:3:(ed3|3-[1-9][0-9]*)\$

Code Description

If present, at least one irm:countryCode in a irm:geographicIdentifier must be a GENC ED3 codespace: ^ge:GENC:3:(ed3|3-[1-9][0-9]*)\$

Schematron Code

```
<sch:pattern id="IRM-ID-00093">
  <sch:rule id="IRM-ID-00093-R1"
    context="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage//irm:geospatialCoverage">
    <sch:let name="hasCountryCodes"
      value="count(irm:geographicIdentifier/irm:countryCode) > 0"/>
    <sch:assert test="not($hasCountryCodes) or (some $countryCode in irm:geographicIdentifier/irm:countryCode satisfies matches(normalize-space($countryCode/
@irm:codespace), '^ge:GENC:3:(ed3|3-[1-9][0-9]*)$'))"
      flag="error"
      role="error">[IRM-ID-00093][Error] irm:countryCode must be a GENC ED3 codespace: ^ge:GENC:3:(ed3|3-[1-9][0-9]*)$</sch:assert>
    </sch:rule>
  </sch:pattern>
```

2.57 - ../Rules/IRM_ID_00094.sch

Rule Description

[IRM-ID-00094][Error] If countrycode codespace is GENC codespace, then value must be in the GENC countrycode cve.

Code Description

If countrycode codespace is GENC codespace, then value must be in the GENC countrycode cve

Schematron Code

```
<sch:pattern id="IRM-ID-00094">
  <sch:rule id="IRM-ID-00094-R1"
    context="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage//irm:geospatialCoverage//irm:countryCode">
    <sch:let name="isGENCCountryCode"
      value="matches(normalize-space(./@irm:codespace), '^ge:GENC:3:(ed3|3-[1-9][0-9]*)$')"/>
    <sch:assert test="not($isGENCCountryCode) or (some $token in $gencCountryCodeList satisfies $token = normalize-space(./@irm:code))"
      flag="error"
      role="error">[IRM-ID-00094][Error] irm:countryCode must be in GENC countrycode cve.</sch:assert>
  </sch:rule>
</sch:pattern>
```

2.58 - ../Rules/IRM_ID_00095.sch

Rule Description

[IRM-ID-00095][Error] If present, at least one irm:subDivisionCode in a irm:geographicIdentifier must be a GENC ED3 codespace: ^as:GENC:6:(ed3|3-[1-9][0-9]*)\$

Code Description

If present, at least one irm:subDivisionCode in a irm:geographicIdentifier must be a GENC ED3 codespace: ^as:GENC:6:(ed3|3-[1-9][0-9]*)\$

Schematron Code

```
<sch:pattern id="IRM-ID-00095">
  <sch:rule id="IRM-ID-00095-R1"
    context="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage//irm:geospatialCoverage">
    <sch:let name="hasSubDivisions"
      value="count(irm:geographicIdentifier/irm:subDivisionCode) > 0"/>
    <sch:assert test="not($hasSubDivisions) or (some $subDivisionCode in irm:geographicIdentifier/irm:subDivisionCode satisfies matches(normalize-space($subDivisionCode/
@irm:codespace), '^as:GENC:6:(ed3|3-[1-9][0-9]*)$'))"
      flag="error"
      role="error">[IRM-ID-00095][Error] irm:subDivisionCode must be a GENC ED3 codespace: ^as:GENC:6:(ed3|3-[1-9][0-9]*)$</sch:assert>
    </sch:rule>
  </sch:pattern>
```

2.59 - ../Rules/IRM_ID_00096.sch

Rule Description

[IRM-ID-00096][Error] Regardless of the version indicated on the instance document, the validation infrastructure MUST use a version of 'ISM' that is version '202111' (Version:2021-NOV) or later. NOTE: This is not an error of the instance document but of the validation environment itself.

Code Description

This rule uses an abstract pattern to consolidate logic. It verifies that the validation infrastructure is using the version specified in parameters.

Schematron Code

```
<?ICEA pattern?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00096" is-a="ValidateValidationEnvCVE">
    <sch:param name="MinVersion" value="'202111'"/>
    <sch:param name="SpecToCheck" value="'ISM'"/>
    <sch:param name="pathToDocument"
        value="'../../../../CVE/ISM/CVEnumISMClassificationAll.xml'"/>
    <sch:param name="RuleID" value="'IRM-ID-00096'"/>
</sch:pattern>
```

2.60 - ../Rules/IRM_ID_00098.sch

Rule Description

[IRM-ID-00098][Error] Use of a GENC codespace requires the presence of the IC-GENC CESVersion attribute.

Code Description

If a codespace attribute is specified that contains a GENC codespace, then ensure that the IC-GENC CESVersion attribute is specified in the IRM assertion on the ICResourceMetadataPackage.

Schematron Code

```
<sch:pattern id="IRM-ID-00098">
  <sch:rule id="IRM-ID-00098-R1"
    context="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage//*[starts-with(@irm:codespace,'ge:GENC') or starts-
with(@irm:codespace,'as:GENC')]">
    <sch:assert test="ancestor-or-self::tdf:*/tdf:Assertion//irm:ICResourceMetadataPackage/@genc:CESVersion"
      flag="error"
      role="error">[IRM-ID-00098][Error] Use of a GENC codespace requires the presence of the IC-GENC CESVersion attribute.</sch:assert>
    </sch:rule>
  </sch:pattern>
```

2.61 - ../Rules/IRM_ID_00099.sch

Rule Description

[IRM-ID-00099][Error] Regardless of the version indicated on the instance document, the validation infrastructure MUST use a version of 'IC-GENC' that is version '201909' (Version:2019-SEP) or later. NOTE: This is not an error of the instance document but of the validation environment itself.

Code Description

This rule uses an abstract pattern to consolidate logic. It verifies that the validation infrastructure is using the version specified in parameters.

Schematron Code

```
<?ICEA pattern?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00099" is-a="ValidateValidationEnvCVE">
    <sch:param name="MinVersion" value="'201909'"/>
    <sch:param name="SpecToCheck" value="'IC-GENC'"/>
    <sch:param name="pathToDocument"
        value="'../../../../CVE/IC-GENC/CVEnumGENCCountryCode.xml'"/>
    <sch:param name="RuleID" value="'IRM-ID-00099'"/>
</sch:pattern>
```


2.62 - ../Rules/IRM_ID_00100.sch

Rule Description

[IRM-ID-00100][Error] If a irm:ICResourceMetadataPackage element is present in a tdf:StructuredStatement within a tdf:Assertion, then the tdf:Assertion must also contain a tdf:StatementMetadata element.

Code Description

If a irm:ICResourceMetadataPackage element is present in a tdf:StructuredStatement within a tdf:Assertion, then the tdf:Assertion must also contain a tdf:StatementMetadata element.

Schematron Code

```
<sch:pattern id="IRM-ID-00100">
  <sch:rule id="IRM-ID-00100-R1"
    context="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage">
    <sch:let name="hasStatementMetadata"
      value="count(..preceding-sibling::tdf:StatementMetadata) > 0"/>
    <sch:assert test="$hasStatementMetadata" flag="error" role="error">[IRM-ID-00100][Error] tdf:Assertion must contain tdf:StatementMetadata when tdf:StructuredStatement
contains irm:ICResourceMetadataPackage.</sch:assert>
  </sch:rule>
</sch:pattern>
```

2.63 - ./Rules/IRM_ID_00101.sch

Rule Description

[IRM-ID-00101][Error] If element irm:organization has attribute @irm:acronym specified and the acronym begins with "USA.", then the organization value must be defined by the USAgency CES. Human Readable: Utilized agency acronyms beginning with "USA." must be defined in the USAgency CES.

Code Description

For irm:organization with @irm:acronym specified, if @irm:acronym begins with 'USA.', then the country value must be defined by the USAgency CES.

Schematron Code

```
<?ICEA pattern?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00101">
    <sch:rule id="IRM-ID-00101-R1"
        context="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage[$IC_COMPLIANCE]//irm:organization[@irm:acronym]">
        <sch:assert test="if (starts-with(normalize-space(@irm:acronym),'USA.')) then (some $token in $USAgencyAcronymList satisfies $token = normalize-space( @irm:acronym ))
else true()"
            flag="error"
            role="error">[IRM-ID-00101][Error] If element irm:organization has attribute @irm:acronym specified and the acronym begins with "USA.", then the
organization value must be defined by the USAgency CES. Found
        <sch:value-of select="normalize-space( @irm:acronym )"/>
        </sch:assert>
    </sch:rule>
</sch:pattern>
```

2.64 - ../Rules/IRM_ID_00102.sch

Rule Description

[IRM-ID-00102][Error] If element irm:organization has attribute @irm:acronym specified, then the country value must be defined by the GENC CES. Human Readable: Utilized agency acronyms country component must have the country defined in the GENC CES.

Code Description

For irm:organization with @irm:acronym specified, if @irm:acronym contains a country component which is not 'USA', then the country value must be defined by the GENC CES.

Schematron Code

```
<?ICEA pattern?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00102">
  <sch:rule id="IRM-ID-00102-R1"
    context="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage[$IC_COMPLIANCE]//irm:organization[@irm:acronym]">
    <sch:let name="CC" value="tokenize(normalize-space(@irm:acronym),'\.')[1]"/>
    <sch:assert test="some $token in $gencCountryCodeList satisfies $token = normalize-space($CC)"
      flag="error"
      role="error">[IRM-ID-00102][Error] If element irm:organization has attribute @irm:acronym specified, then the country value must be defined by the GENC CES.
    </sch:assert>
  </sch:rule>
</sch:pattern>
```

Found

2.65 - ../Rules/IRM_ID_00103.sch

Rule Description

[IRM-ID-00103][Error] If element irm:organization has attribute @irm:acronym specified, then attribute @irm:acronym must be of type NmToken.

Code Description

For irm:organization that have @irm:acronym, @irm:acronym must be of type NmToken.

Schematron Code

```
<?ICEA pattern?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00103">
    <sch:rule id="IRM-ID-00103-R1"
        context="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage[$IC_COMPLIANCE]//irm:organization[@irm:acronym]">
        <sch:assert test="util:meetsType(@irm:acronym, $NmTokenPattern)"
            flag="error"
            role="error">[IRM-ID-00103][Error] If element irm:organization has attribute @irm:acronym specified, then attribute @irm:acronym must be of type NmToken.</
sch:assert>
        </sch:rule>
    </sch:pattern>
```

2.66 - ../Rules/IRM_ID_00104.sch

Rule Description

[IRM-ID-00104][Error] If element irm:organization has attribute @irm:acronym specified and @irm:acronym has a country prefix, then the agency suffix after the dot delimiter must be of type NmToken.

Code Description

For irm:organization that have @irm:acronym, the agency suffix after the dot delimiter must be of type NmToken.

Schematron Code

```
<?ICEA pattern?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00104">
    <sch:rule id="IRM-ID-00104-R1"
        context="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage[$IC_COMPLIANCE]//irm:organization[@irm:acronym]">
        <sch:assert test="util:meetsType(substring-after(@irm:acronym, '.'), $NmTokenPattern)"
            flag="error"
            role="error">[IRM-ID-00104][Error] If element irm:organization has attribute @irm:acronym specified, then the agency suffix after the dot delimiter must be
of type NmToken. Found
        <sch:value-of select="normalize-space(@irm:acronym)"/>substring was:
        <sch:value-of select="substring-after(@irm:acronym, '.')"/>
        </sch:assert>
    </sch:rule>
</sch:pattern>
```

2.67 - ../Rules/IRM_ID_00105.sch

Rule Description

[IRM-ID-00105][Error] Use of the @irm:qualifier on the irm:nonStateActor element is required.

Code Description

If the irm:nonStateActor element is specified within a TDF assertion affected by an IRM assertion, then verify that the @irm:qualifier attribute is present on the element.

Schematron Code

```
<?ICEA pattern?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00105">
    <sch:rule id="IRM-ID-00105-R1"
        context="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage[$IC_COMPLIANCE]//irm:nonStateActor">
        <sch:assert test="@irm:qualifier" flag="error" role="error">[IRM-ID-00105][Error] Use of the @irm:qualifier on the irm:nonStateActor element is required.</sch:assert>
    </sch:rule>
</sch:pattern>
```

2.68 - ../Rules/IRM_ID_00106.sch

Rule Description

[IRM-ID-00106][Error] Regardless of the version indicated on the instance document, the validation infrastructure MUST use a version of 'PMA' that is version '201903.201909' (Version:2019-MAR Revision: 2019-SEP) or later. NOTE: This is not an error of the instance document but of the validation environment itself.

Code Description

This rule uses an abstract pattern to consolidate logic. It verifies that the validation infrastructure is using the version specified in parameters.

Schematron Code

```
<?ICEA pattern?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00106" is-a="ValidateValidationEnvSchema">
    <sch:param name="MinVersion" value="'201903.201909'"/>
    <sch:param name="SpecToCheck" value="'PMA'"/>
    <sch:param name="pathToDocument" value="'../Schema/PMA/PMA-XML.xsd'"/>
    <sch:param name="RuleID" value="'IRM-ID-00106'"/>
</sch:pattern>
```

2.69 - ../Rules/IRM_ID_00107.sch

Rule Description

[IRM-ID-00107][Error] Regardless of the version indicated on the instance document, the validation infrastructure MUST use a version of 'VIRT' that is version '202010' (Version:2020-OCT) or later. NOTE: This is not an error of the instance document but of the validation environment itself.

Code Description

This rule uses an abstract pattern to consolidate logic. It verifies that the validation infrastructure is using the version specified in parameters.

Schematron Code

```
<?ICEA pattern?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00107" is-a="ValidateValidationEnvSchema">
    <sch:param name="MinVersion" value="'202010'"/>
    <sch:param name="SpecToCheck" value="'VIRT'"/>
    <sch:param name="pathToDocument" value="'../Schema/VIRT/VIRT.xsd'"/>
    <sch:param name="RuleID" value="'IRM-ID-00107'"/>
</sch:pattern>
```


2.70 - ../Rules/IRM_ID_00108.sch

Rule Description

[IRM-ID-00108][Warning] irm:DESVersion attribute SHOULD be specified as version 202111 (Version:2021-NOV) with an optional extension.

Code Description

This rule supports extending the version identifier with an optional trailing hyphen and up to 23 additional characters. The version must match the regular expression “^202111(-.{1,23})?\$”.

Schematron Code

```
<?ICEA pattern?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00108">
    <sch:rule id="IRM-ID-00108-R1" context="*[@irm:DESVersion]">
        <sch:assert test="matches(@irm:DESVersion, '^202111(-.{1,23})?$')"
            flag="warning"
            role="warning">[IRM-ID-00108][Warning] irm:DESVersion attribute SHOULD be specified as version 202111 (Version:2021-NOV) with an optional extension. The
value provided was:
        <sch:value-of select="@irm:DESVersion"/>
    </sch:assert>
    </sch:rule>
</sch:pattern>
```

2.71 - ../Rules/IRM_ID_00109.sch

Rule Description

[IRM-ID-00109][Error] If the tokens in @irm:compliesWith starts with "USA", they must be a value that exists or starts with a value from CVENumIRMCompliesWithUSA.

Code Description

If @irm:compliesWith starts-with USA, must be or start with a value from CVE values start-with USA.

Schematron Code

```
<?ICEA pattern?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00109">
    <sch:rule id="IRM-ID-00109-R1"
        context="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage">
        <sch:assert test="util:checkUSATokenValidity(@irm:compliesWith,$compliesWithUSATypeList)"
            flag="error"
            role="error">[IRM-ID-00109][Error] If the tokens in @irm:compliesWith starts with "USA", they must be a value that exists or starts with a value from
CVENumIRMCompliesWithUSA.
        <sch:value-of select="concat('[',@irm:compliesWith,']')" />
        </sch:assert>
    </sch:rule>
</sch:pattern>
```

2.72 - ../Rules/IRM_ID_00110.sch

Rule Description

[IRM-ID-00110][Error] The string before the first underscore of each @irm:compliesWith token must be in IC-GENC.

Code Description

Checks if the substring before the first underscore of each token in @irm:compliesWith is a country code in IC-GENC.

Schematron Code

```
<?ICEA pattern?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00110">
    <sch:rule id="IRM-ID-00110-R1"
        context="/tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage">
        <sch:assert test="util:containsOnlyTheTokensSubstringBefore('_',@irm:compliesWith,$gencCountryCodeList)"
            flag="error"
            role="error">[IRM-ID-00110][Error] The string before the first underscore of each @irm:compliesWith token must be in IC-GENC.
        <sch:value-of select="concat('[@irm:compliesWith,']')"/>
        </sch:assert>
    </sch:rule>
</sch:pattern>
```

2.73 - ../Rules/IRM_ID_00111.sch

Rule Description

[IRM-ID-00111][Error] If irm:virtualCoverage exists, then it must include at least @virt:protocol or @virt:address.

Code Description

If irm:virtualCoverage exists, then it must include at least @virt:protocol or @virt:address.

Schematron Code

```
<?ICEA pattern?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00111">
  <sch:rule id="IRM-ID-00111-R1" context="irm:virtualCoverage">
    <sch:assert test="exists(./@virt:protocol) or exists(./@virt:address)"
              flag="error"
              role="error">[IRM-ID-00110][Error] If irm:virtualCoverage exists, then it must include at least @virt:protocol or @virt:address.</sch:assert>
  </sch:rule>
</sch:pattern>
```

2.74 - ../Rules/IRM_ID_00112.sch

Rule Description

[IRM-ID-00112][Warning] If element `irm:mimeType` is specified, it SHOULD have a value from `CVEnumMimeType.xml` other than the wildcard entry. The value is not explicitly defined in `CVEnumMimeType.xml` and is a match ONLY because of the wildcard entry.

Code Description

If element `irm:mimeType` is specified, it SHOULD have a value from `CVEnumMimeType.xml` other than the wildcard entry. The value is not explicitly defined in `CVEnumMimeType.xml` and is a match ONLY because of the wildcard entry.

Schematron Code

```
<sch:pattern id="IRM-ID-00112">
  <sch:rule id="IRM-ID-00112-R1"
    context="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage[$IC_COMPLIANCE]//irm:mimeType">
    <sch:assert test="some $token in $nonRegexMimeTypeList satisfies . = $token or matches(., concat('^\',$token,'$'))"
      flag="warning"
      role="warning">[IRM-ID-00112][Warning] If element irm:mimeType is specified, it SHOULD have a value from CVEnumMimeType.xml other than the wildcard entry.
      MIME type [
        <sch:value-of select="."/>] is not explicitly defined in CVEnumMimeType.xml and is a match ONLY because of the wildcard entry.
    </sch:assert>
  </sch:rule>
</sch:pattern>
```

2.75 - ../Rules/IRM_ID_00113.sch

Rule Description

[IRM-ID-00113][Error] If present, at least one irm:waterBody in a irm:geographicIdentifier must be a CWW ED1 codespace: ^wb:CWW:3:(ed1)\$

Code Description

If present, at least one irm:waterBody in a irm:geographicIdentifier must be a CWW ED3 codespace: ^wb:CWW:3:(ed1)\$

Schematron Code

```
<sch:pattern id="IRM-ID-00113">
  <sch:rule id="IRM-ID-00113-R1"
    context="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage//irm:geospatialCoverage">
    <sch:let name="hasWaterBody"
      value="count(irm:geographicIdentifier/irm:waterBody) > 0"/>
    <sch:assert test="not($hasWaterBody) or (some $waterBody in irm:geographicIdentifier/irm:waterBody satisfies matches(normalize-space($waterBody/@irm:codespace),
' ^wb:CWW:3:(ed1)$' ))"
      flag="error"
      role="error">[IRM-ID-00113][Error] irm:waterBody must be a GENC ED3 codespace: ^wb:CWW:3:(ed1)$</sch:assert>
    </sch:rule>
  </sch:pattern>
```

2.76 - ../Rules/IRM_ID_00114.sch

Rule Description

[IRM-ID-00114][Error] If waterBody codespace is CWW codespace, then value must be in the CWW waterBody cve.

Code Description

If waterBody codespace is CWW waterBody, then value must be in the CWW waterBody cve

Schematron Code

```
<sch:pattern id="IRM-ID-00114">
  <sch:rule id="IRM-ID-00114-R1"
    context="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage//irm:geospatialCoverage//irm:waterBody">
    <sch:let name="isCWWWaterBody"
      value="matches(normalize-space(./@irm:codespace), '^wb:CWW:3:(ed1)$')"/>
    <sch:assert test="not($isCWWWaterBody) or (some $token in $waterBodyList satisfies $token = normalize-space(./@irm:code))"
      flag="error"
      role="error">[IRM-ID-00114][Error] irm:waterBody must be in CWW waterBody cve.</sch:assert>
    </sch:rule>
  </sch:pattern>
```

Chapter 3 - Abstract Patterns

All of the Abstract Patterns for IRM are listed in this section. These patterns may depend on variables defined in the Schematron Schema section.

3.1 - ./Lib/CompareDateTimes.sch

Code Description

\$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

First, ensure that the primaryDate is an allowable date format. If the primary date is not a valid date format, then return true because there is no guarantee the value provided is not allowed. Then, for each date in \$secondaryDateList, perform the same check for a valid date format and compare the secondaryDate to the primaryDate. To perform comparisons between dates, take the comparison operator contained in the param \$operator and ensure that all comparisons between primary and secondary dates return true.

Schematron Code

```
<?ICEA abstractPattern?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->
<!--
  $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, ensure that the primaryDate is an allowable date format. If the primary date is not a valid
  date format, then return true because there is no guarantee the value provided is not allowed. Then, for
  each date in $secondaryDateList, perform the same check for a valid date format and compare the
  secondaryDate to the primaryDate. To perform comparisons between dates, take the comparison operator
  contained in the param $operator and ensure that all comparisons between primary and secondary dates
  return true.
-->

<sch:pattern abstract="true" id="CompareDateTimes">
  <sch:rule id="CompareDateTimes-R1" 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"
      role="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"
      role="error">
      <sch:value-of select="$ruleText"/>
    </sch:assert>
  </sch:rule>
</sch:pattern>
```


3.2 - ./Lib/ICIdentifierRestrictions.sch

Code Description

A valid IC-Identifier must meet the following criteria: (1) The id must begin with 'guide:/' (2) The prefix for the id is an integer up to 16 digits with no leading zeros allowed (3) The suffix is an alphanumeric string limited to 36 characters of the set that includes A-Z, a-z, 0-9, underscore, hyphen, and period (4) There are no additional characters proceeding the ID. In order to determine the provided IC-Identifier meets these criteria, the value parameter is matched against the following regex: ^guide://([1-9][0-9]{0,15}|0)/[A-Za-z0-9_-\.\.]{1,36}\$.

Schematron Code

```
<sch:pattern abstract="true" id="ICIdentifierRestrictions">
  <sch:rule id="ICIdentifierRestrictions-R1" context="$context">
    <sch:let name="icidRegEx"
      value="'^guide://([1-9][0-9]{0,15}|0)/[A-Za-z0-9_-\.\.]{1,36}$'"/>
    <sch:assert test="matches(string($value),$icidRegEx)" flag="error" role="error">
      <sch:value-of select="$errorMessage"/>
    </sch:assert>
  </sch:rule>
</sch:pattern>
```

3.3 - ../Lib/IsmEnforcement.sch

Code Description

\$qualifier := the qualifier value that requires ism to be present

\$errMsg := the error message text to display when the assertion fails

This abstract pattern checks that if a particular qualifier is specified on a irm:type element that ism:classification is also specified.

Schematron Code

```
<?ICEA abstractPattern?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
      -->
<!--
      This abstract pattern checks that if a particular qualifier is specified on a
      irm: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 xmlns:ism="urn:us:gov:ic:ism" 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="irm:type[@irm:qualifier=$qualifier]"/>
      <sch:param name="errMsg" value="'
      [IRM-ID-00039][Error]
      If irm: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 id="IsmEnforcement-R1" context="$context">
        <sch:assert test="@ism:classification" flag="error" role="error">
            <sch:value-of select="$errMsg"/>
        </sch:assert>
    </sch:rule>
</sch:pattern>
```

3.4 - ./Lib/ValidateValidationEnvCVE.sch

Code Description

This abstract pattern checks to see if the validation environment has at least the version / revision of the CVE as of the writing of this specification. The calling rule must pass in \$MinVersion, \$SpecToCheck, \$pathToDocument, \$RuleID.

Schematron Code

```
<!--
  This abstract pattern checks to see the version of a CVE is greater than or equal to a passed in parameter.

  $MinVersion      := the version that SpecToCheck must be equal to or greater than.
  $SpecToCheck      := Name the spec whose version in the infrastructure is being checked.
  $pathToDocument   := Relative path to the document cve that has ther version string
  $RuleID           := The number of the rule in the concrete file.
-->

<sch:pattern xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
              abstract="true"
              id="ValidateValidationEnvCVE">
  <sch:rule id="ValidateValidationEnvCVE-R1" context="/">
    <sch:assert test="document($pathToDocument)//cve:CVE//@specVersion castable as xs:double and document($pathToDocument)//cve:CVE//@specVersion >= $MinVersion"
               flag="error"
               role="error">[
  <sch:value-of select="$RuleID"/>][Error] Version [
  <sch:value-of select="document($pathToDocument)//cve:CVE//@specVersion"/>] of
  <sch:value-of select="$SpecToCheck"/>found; Version [
  <sch:value-of select="$MinVersion"/>] or later is required. The latest version of
  <sch:value-of select="$SpecToCheck"/>is not being used in the validation infrastructure. Regardless of the version indicated on the instance document, the validation infrastructure needs
to use a version of
  <sch:value-of select="$SpecToCheck"/>that is version [
  <sch:value-of select="$MinVersion"/>] or later. NOTE: This is not an error of the instance document but of the validation environment itself. The incorrect value was found in
  <sch:value-of select="document-uri(document($pathToDocument))"/>
    </sch:assert>
  </sch:rule>
</sch:pattern>
```

3.5 - ./Lib/ValidateValidationEnvSchema.sch

Code Description

This abstract pattern checks to see if the validation environment has at least the version / revision of the Schema as of the writing of this specification. The calling rule must pass in \$MinVersion, \$SpecToCheck, \$pathToDocument, \$RuleID.

Schematron Code

```
<!--
  This abstract pattern checks to see the version of a Schema is greater than or equal to a passed in parameter.

  $MinVersion      := the version that SpecToCheck must be equal to or greater than.
  $SpecToCheck     := Name the spec whose version in the infrastructure is being checked.
  $pathToDocument  := Relative path to the document xsd that has ther version string
  $RuleID          := The number of the rule in the concrete file.
-->

<sch:pattern xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
              abstract="true"
              id="ValidateValidationEnvSchema">
  <sch:rule id="ValidateValidationEnvSchema-R1" context="/">
    <sch:assert test="document($pathToDocument)//xsd:schema/@version castable as xs:double and document($pathToDocument)//xsd:schema/@version >= $MinVersion"
              flag="error"
              role="error">[
  <sch:value-of select="$RuleID"/>][Error] Version [
  <sch:value-of select="document($pathToDocument)//xsd:schema/@version"/>] of
  <sch:value-of select="$SpecToCheck"/>found; Version [
  <sch:value-of select="$MinVersion"/>] or later is required. The latest version of
  <sch:value-of select="$SpecToCheck"/>is not being used in the validation infrastructure. Regardless of the version indicated on the instance document, the validation infrastructure needs
to use a version of
  <sch:value-of select="$SpecToCheck"/>that is version [
  <sch:value-of select="$MinVersion"/>] or later. NOTE: This is not an error of the instance document but of the validation environment itself. The incorrect value was found in
  <sch:value-of select="document-uri(document($pathToDocument))"/>
    </sch:assert>
  </sch:rule>
</sch:pattern>
```

3.6 - ./Lib/ValidateValueExistenceInList.sch

Code Description

\$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

This abstract pattern checks to see if an attribute of an element exists in a list.

Schematron Code

```
<?ICEA abstractPattern?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->
<!--
      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 xmlns:ism="urn:us:gov:ic:ism" is-a="ValidateValueExistenceInList" id="IRM_ID_00027" xmlns:sch="http://purl.oclc.org/dsdl/schematron">
        <sch:param name="context" value="tdf:*[tdf:Assertion//irm:ICResourceMetadataPackage]/tdf:Assertion/tdf:StructuredStatement/irm:resource//irm:language"/>
        <sch:param name="searchTerm" value="@irm:qualifier"/>
        <sch:param name="list" value="$compoundLanguageQualifierTypeList"/>
        <sch:param name="errMsg" value="'
          [IRM-ID-00034][Error] For element irm:language, attribute irm:qualifier must have a
          value in CVEnumIRMCompoundLanguageQualifierType.xml.'"/>
      </sch:pattern>

      Note: $iso4217TrigraphList is defined in the main document, IRM_XML.xml.
-->

<sch:pattern abstract="true" id="ValidateValueExistenceInList">
  <sch:rule id="ValidateValueExistenceInList-R1" context="$context">
    <sch:assert test="some $token in $list satisfies $token = $searchTerm or matches($searchTerm, concat('^',$token,'$'))"
      flag="error"
      role="error">
      <sch:value-of select="$errMsg"/>
    </sch:assert>
  </sch:rule>
</sch:pattern>
```

Chapter 4 - Min Accessible Rules

All of the numbered Rules for IRM that are specifically required for the MIN_ACCESSIBLE compliesWith mode are listed in this section. These rules are also enforced when the compliesWith mode is set to MIN_DISCOVERABLE since that is a super set. There are other rules that could come into play when using MIN_ACCESSIBLE if you add any data that would trigger a rule. This set of rules is called out to help understand what MIN_ACCESSIBLE means. These rules may depend on patterns defined in the Abstract Patterns section or on variables defined in the Schematron Schema section.

4.1 - ../Rules/IRM_ID_00062.sch

Rule Description

[IRM-ID-00062][Error] The value of an IC-ID identifier must follow standardized convention. Human Readable: The IC-ID identifier value has to follow standardized convention.

Code Description

This rule uses an abstract pattern that contains the logic for ensuring the value found if a given context exists, both provided as parameters from this implementation, follows the IC-ID identifier standardized convention.

Schematron Code

```
<?ICEA pattern?>
<?ICEA min_accessible?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00062" is-a="ICIdentifierRestrictions">
    <sch:param name="context"
               value="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage//irm:identifier[@irm:qualifier='IC-ID']"/>
    <sch:param name="value" value="string(@irm:value)"/>
    <sch:param name="errorMessage"
               value="'[IRM-ID-00062][Error] The value of an IC-ID identifier must follow standardized convention. Human Readable: The IC-ID identifier value has to follow
standardized convention.'"/>
</sch:pattern>
```

4.2 - ../Rules/IRM_ID_00063.sch

Rule Description

[IRM-ID-00063][Error] Element irm:ICResourceMetadataPackage/irm:creator/irm:organization must specify attribute @irm:acronym. Human Readable: The IRM card must specify a creator organization with an IC agency acronym for the referenced resource.

Code Description

Make sure that element irm:ICResourceMetadataPackage/irm:creator/irm:organization exists and specifies attribute @irm:acronym.

Schematron Code

```
<?ICEA pattern?>
<?ICEA min_accessible?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00063">
    <sch:rule id="IRM-ID-00063-R1"
        context="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage[$IC_COMPLIANCE]">
        <sch:assert test="irm:creator/irm:organization/@irm:acronym"
            flag="error"
            role="error">[IRM-ID-00063][Error] Element irm:ICResourceMetadataPackage/irm:creator/irm:organization must specify attribute @irm:acronym. Human Readable:
The IRM card must specify a creator organization with an IC agency acronym for the referenced resource.</sch:assert>
        </sch:rule>
    </sch:pattern>
```

4.3 - ../Rules/IRM_ID_00064.sch

Rule Description

[IRM-ID-00064][Error] Element irm:ICResourceMetadataPackage/irm:dates must specify attribute @irm:created. Human Readable: The IRM card must specify the date on which the referenced resource was created.

Code Description

Make sure that element irm:ICResourceMetadataPackage/irm:dates exists and specifies attribute @irm:created.

Schematron Code

```
<?ICEA pattern?>
<?ICEA min_accessible?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00064">
    <sch:rule id="IRM-ID-00064-R1"
              context="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage[$IC_COMPLIANCE]">
        <sch:assert test="irm:dates/@irm:created" flag="error" role="error">[IRM-ID-00064][Error] Element irm:ICResourceMetadataPackage/irm:dates must specify attribute
@irm:created. Human Readable: The IRM card must specify the date on which the referenced resource was created.</sch:assert>
        </sch:rule>
    </sch:pattern>
```

4.4 - ../Rules/IRM_ID_00065.sch

Rule Description

[IRM-ID-00065][Error] Attribute irm:ICResourceMetadataPackage/irm:dates/@irm:created must be castable as an xs:dateTime type. Human Readable: The date on which the referenced resource was created must be a dateTime type.

Code Description

Make sure that attribute dms:resource/irm:dates/@irm:created is castable as an xs:dateTime type.

Schematron Code

```
<?ICEA pattern?>
<?ICEA min_accessible?>
<!-- Notices - Distribution Notice:
      This document has been approved for Public Release and is available for use without restriction.
-->

<sch:pattern id="IRM-ID-00065">
    <sch:rule id="IRM-ID-00065-R1"
        context="tdf:*/tdf:Assertion/tdf:StructuredStatement/irm:ICResourceMetadataPackage/irm:dates[@irm:created]">
        <sch:assert test="@irm:created castable as xs:dateTime"
            flag="error"
            role="error">[IRM-ID-00065][Error] Attribute irm:ICResourceMetadataPackage/irm:dates/@irm:created must be castable as an xs:dateTime type. Human Readable:
The date on which the referenced resource was created must be a dateTime type.</sch:assert>
        </sch:rule>
    </sch:pattern>
```

Chapter 5 - Schematron Schema

The top level Schematron file for IRM is in this section. This file imports all of the others and also defines many global variables they are all dependent on.

5.1 - ../IRM_XML.sch

Code Description

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

Schematron Code

```
<!--UNCLASSIFIED-->
<?ICEA master?>
<!-- UNCLASSIFIED -->
<!-- Notices - Distribution Notice:
    This document has been approved for Public Release and is available for use without restriction.
-->
<!-- WARNING:
    Once compiled into an XSLT the result will
    be the aggregate classification of all the CVES
    and included .sch files
-->

<sch:schema 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:virt" prefix="virt"/>
    <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:gov:ic:ntk" prefix="ntk"/>
    <sch:ns uri="urn:us:gov:ic:tdf" prefix="tdf"/>
    <sch:ns uri="urn:us:gov:ic:id" prefix="icid"/>
    <sch:ns uri="urn:us:gov:ic:usagency" prefix="usagency"/>
    <sch:ns uri="urn:us:gov:ic:pm" prefix="pm"/>
    <sch:ns uri="urn:us:gov:ic:intdis" prefix="intdis"/>
    <sch:ns uri="http://www.example.com/functions/local" prefix="local"/>
    <sch:ns uri="urn:us:gov:ic:mime" prefix="mime"/>
    <sch:ns uri="urn:us:gov:ic:icgenc" prefix="genc"/>
    <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"/>
    <sch:ns prefix="util" uri="urn:us:gov:ic:irm:xsl:util"/>
    <sch:let name="IRM_COMPLIES_WITH"
        value="irm:ICResourceMetadataPackage/@irm:compliesWith"/>
    <sch:let name="MIN_DISCOVERABLE_OR_GREATER"
        value="util:containsAnyOfTheTokens($IRM_COMPLIES_WITH, ('MIN_DISCOVERABLE'))"/>
    <sch:let name="MIN_ACCESSIBLE_OR_GREATER"
        value="$MIN_DISCOVERABLE_OR_GREATER or util:containsAnyOfTheTokens($IRM_COMPLIES_WITH, ('MIN_ACCESSIBLE'))"/>
    <sch:let name="COMPLIANCE_ATTRIBUTE"
        value="//irm:ICResourceMetadataPackage/@irm:compliesWith"/>
    <sch:let name="IC_COMPLIANCE"
        value="some $token in $COMPLIANCE_ATTRIBUTE satisfies $token='USA_IC'"/>
    <!-- (U) Resources -->

<sch:let name="coverageIso3166DigraphList"
    value="document(' ../../CVE/IRM/CVEnumIRMCoverageISO3166Digraph.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/MIME/CVEnumMimeType.xml')//cve:Value"/>
<sch:let name="nonRegexMimeTypeList"
  value="document('.../CVE/MIME/CVEnumMimeType.xml')/cve:CVE/cve:Enumeration/cve:Term[not(./@deprecated)]/cve:Value[not(./@regularExpression)]"/>
<sch:let name="deprecatedMimeTypeList"
  value="document('.../CVE/MIME/CVEnumMimeType.xml')/cve:CVE/cve:Enumeration/cve:Term[./@deprecated]/cve:Value"/>
<sch:let name="compliesWithUSATypeList"
  value="document('.../CVE/IRM/CVEnumIRMCompliesWithUSA.xml')//cve:Value"/>
<sch:let name="compoundLanguageQualifierTypeList"
  value="document('.../CVE/IRM/CVEnumIRMCompoundLanguageQualifierType.xml')//cve:Value"/>
<sch:let name="intelDisciplineComponentTechniqueList"
  value="document('.../CVE/INTDIS/CVEnumIntelDisciplineComponentTechnique.xml')//cve:Value"/>
<sch:let name="intelDisciplineComponentList"
  value="document('.../CVE/INTDIS/CVEnumIntelDisciplineComponent.xml')//cve:Value"/>
<sch:let name="intelDisciplineList"
  value="document('.../CVE/INTDIS/CVEnumIntelDiscipline.xml')//cve:Value"/>
<sch:let name="positiveIntelList"
  value="document('.../CVE/IRM/CVEnumIRMPositiveIntel.xml')//cve:Value"/>
<sch:let name="activityList"
  value="document('.../CVE/IRM/CVEnumIRMActivity.xml')//cve:Value"/>
<sch:let name="executableIndicatorList"
  value="document('.../CVE/IRM/CVEnumIRMExecutableIndicator.xml')//cve:Value"/>
<sch:let name="maliciousCodeIndicatorList"
  value="document('.../CVE/IRM/CVEnumIRMMaliciousCodeIndicator.xml')//cve:Value"/>
<sch:let name="coveragePrecedenceList"
  value="document('.../CVE/IRM/CVEnumIRMCoveragePrecedence.xml')//cve:Value"/>
<sch:let name="USAgencyAcronymList"
  value="document('.../CVE/USAgency/CVEnumUSAgencyAcronym.xml')//cve:CVE/cve:Enumeration/cve:Term/cve:Value"/>
<sch:let name="nonStateActorsList"
  value="document('.../CVE/PM/CVEnumPMNonStateActors.xml')//cve:Value"/>
<sch:let name="gencCountryCodeList"
  value="document('.../CVE/IC-GENC/CVEnumGENCCountryCode.xml')//cve:Value"/>
<sch:let name="waterBodyList"
  value="document('.../CVE/IRM/CVEnumIRMWaterBody.xml')//cve:Value"/>
<sch:let name="gencSubDivisionList"
  value="document('.../CVE/IC-GENC/CVEnumGENCSubDivisionCode.xml')//cve:Value"/>
<!-- ***** -->
<!-- * General Global Variables * -->
<!-- ***** -->

<sch:let name="GMTTimeZoneOffset" value="'PT0H'"/>
  <sch:let name="currentYear"
    value="year-from-dateTime(adjust-dateTime-to-timezone(current-dateTime(), xs:dayTimeDuration($GMTTimeZoneOffset)))/>
  <sch:let name="timeZoneRegEx" value="'Z|[\+-]\d{2}:\d{2}'"/>
  <sch:let name="endsWithTimeZoneRegEx" value="concat('^.*',$timeZoneRegEx,$')"/>
  <sch:let name="startDateTimeTemplate" value="'0001-01-01T00:00:00.000'"/>
  <sch:let name="endDateTimeTemplate" value="'9999-12-01T23:59:59.999'"/>
  <sch:let name="defaultTimeZone" value="'Z'"/>
  <sch:let name="gYearRegEx" value="'^\d{4}(Z|[\+-]\d{2}:\d{2})?$'"/>
  <sch:let name="gYearMonthRegEx" value="'^\d{4}-\d{2}(Z|[\+-]\d{2}:\d{2})?$'"/>
  <sch:let name="dateRegEx" value="'^\d{4}-\d{2}-\d{2}(Z|[\+-]\d{2}:\d{2})?$'"/>
  <sch:let name="dateHourMinTypeRegEx"
    value="'^\d{4}-\d{2}-\d{2}T\d{2}:\d{2}(Z|[\+-]\d{2}:\d{2})?$'"/>
  <sch:let name="dateTimeRegEx"
```



```
        value="'^\\d{4}-\\d{2}-\\d{2}T\\d{2}:\\d{2}:\\d{2}(\\.\\d{1,3})?(Z|\\+|\\-|\\d{2}:\\d{2})?$',"/>
    <!-- ***** -->
<!-- * 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"/>
    <sch:include href="./Lib/ICIdentifierRestrictions.sch"/>
    <sch:include href="./Lib/TypeConstraintPatterns.sch"/>
    <sch:include href="./Lib/ValidateValidationEnvSchema.sch"/>
    <sch:include href="./Lib/ValidateValidationEnvCVE.sch"/>
    <!-- ***** -->

<!-- * Custom XSLT2 Function Definitions * -->
<!-- ***** -->
<!--
    Returns value of irm:CombinedDateType adjusted to GMT timezone.
-->

<xsl:function name="dtf:adjust-CombinedDate-to-GMT-timezone" as="xs:string">
    <xsl:param name="combinedDate" as="xs:string"/>
    <xsl:choose>
        <xsl:when test="matches($combinedDate, $timeZoneRegEx)">
            <xsl:choose>
                <xsl:when test="matches($combinedDate, $dateRegEx)">
                    <xsl:value-of select="adjust-date-to-timezone(xs:date($combinedDate), xs:dayTimeDuration($GMTTimeZoneOffset))"/>
                </xsl:when>
                <xsl:when test="matches($combinedDate, $dateHourMinTypeRegEx)">
                    <xsl:variable name="zeroSecondsPadding" select="':00'"/>
                    <xsl:variable name="combinedDatePadWithSeconds"
                        select="concat(substring($combinedDate, 1, 16), $zeroSecondsPadding, substring($combinedDate, 17))"/>
                    <xsl:value-of select="adjust-dateTime-to-timezone(xs:dateTime($combinedDatePadWithSeconds), xs:dayTimeDuration($GMTTimeZoneOffset))"/>
                </xsl:when>
                <xsl:when test="matches($combinedDate, $dateTimeRegEx)">
                    <xsl:value-of select="adjust-dateTime-to-timezone(xs:dateTime($combinedDate), xs:dayTimeDuration($GMTTimeZoneOffset))"/>
                </xsl:when>
                <xsl:otherwise>
                    <xsl:value-of select="$combinedDate"/>
                </xsl:otherwise>
            </xsl:choose>
        </xsl:when>
        <xsl:otherwise>
            <xsl:value-of select="$combinedDate"/>
        </xsl:otherwise>
    </xsl:choose>
</xsl:function>
<!--
    Returns true if the raw value match the provided regular expressions.
-->

<xsl:function name="util:meetsType" as="xs:boolean">
    <xsl:param name="value"/>
```

```

        <xsl:param name="typePattern" as="xs:string"/>
        <xsl:value-of select="matches(string($value), concat('^(', $typePattern, ')$'))"/>
    </xsl:function>
    <!--
    Returns true if any token in the attribute value matches at least one token in the provided list.
    -->

<xsl:function name="util:containsAnyOfTheTokens" as="xs:boolean">
    <xsl:param name="attribute"/>
    <xsl:param name="tokenList" as="xs:string+"/>
    <xsl:value-of select="some $attrToken in tokenize(normalize-space(string($attribute)), ' ') satisfies $attrToken = $tokenList"/>
</xsl:function>
    <!--
    Returns true if the substring-before with a specific delimiter of all tokens in the attribute value matches at least one token in the provided list.
    -->

<xsl:function name="util:containsOnlyTheTokensSubstringBefore" as="xs:boolean">
    <xsl:param name="delimiter" as="xs:string"/>
    <xsl:param name="attribute"/>
    <xsl:param name="tokenList" as="xs:string+"/>
    <xsl:value-of select="every $attrToken in tokenize(normalize-space(string($attribute)), ' ') satisfies substring-before($attrToken,$delimiter) = $tokenList"/>
</xsl:function>
    <!--
    Returns true if all tokens starting with USA are values in or start with USA values from $tokenList
    -->

<xsl:function name="util:checkUSATokenValidity" as="xs:boolean">
    <xsl:param name="attribute"/>
    <xsl:param name="tokenList" as="xs:string+"/>
    <xsl:value-of select="every $attrToken in tokenize(normalize-space(string($attribute)), ' ') satisfies if (starts-with($attrToken,'USA')) then some $token in $tokenList
satisfies $attrToken = $token or starts-with($attrToken,$token) else true()"/>
</xsl:function>
    <!--
    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($input)"/>
    <xsl:value-of select="matches($trimmedInput, $gYearRegEx) or matches($trimmedInput, $gYearMonthRegEx) or matches($trimmedInput, $dateRegEx) or matches($trimmedInput, $dateHourMinTypeRegEx) or matches($trimmedInput, $dateTimeRegEx)"/>
</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 <= $secondaryEnd and $secondaryStart <= $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 < $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 < $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="primaryAndSecondYearPortionsHaveFourDigits"
as="xs:boolean"
    select="dtf:yearPortionHasFourDigits($primary) and dtf:yearPortionHasFourDigits($secondary)"/>
  <xsl:choose>
    <xsl:when test="$primaryAndSecondYearPortionsHaveFourDigits">
      <xsl:variable name="primaryStart"
as="xs:dateTime"
        select="dtf:startDate($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 $operator = '<') and (($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 $operator = '>') and (($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 = '<') or $operator = '<='">
          <xsl:value-of select="dtf:isBefore($primary, $secondary) or dtf:overlaps($primary, $secondary)"/>
        </xsl:when>
        <!-- 'Greater Than' and 'Greater Than or Equal' -->

        <xsl:when test="$operator = 'gt' or $operator = '>') or $operator = '>='">
          <xsl:value-of select="dtf:isAfter($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>
<xsl:function name="local:getAbsolutePath" as="xs:string"><!-- Given a path resolves any ".." or "." terms to produce an absolute path -->

<xsl:param name="sourcePath" as="xs:string"/>
  <xsl:variable name="pathTokens"
    select="tokenize($sourcePath, '/')"
    as="xs:string*" />
  <xsl:if test="false()">
    <xsl:message>+ DEBUG local:getAbsolutePath(): Starting</xsl:message>
    <xsl:message>+ sourcePath="
  <xsl:value-of select="$sourcePath"/>"
</xsl:message>

    </xsl:if>
    <xsl:variable name="baseResult"
      select="string-join(local:makePathAbsolute($pathTokens, ()), '/')"
      as="xs:string" />
    <xsl:variable name="result"
      as="xs:string"
      select="if (starts-with($sourcePath, '/') and not(starts-with($baseResult, '/'))) then concat('/', $baseResult) else $baseResult" />
    <xsl:if test="false()">
      <xsl:message>+ DEBUG: result="
  <xsl:value-of select="$result"/>"
</xsl:message>

    </xsl:if>
    <xsl:value-of select="$result"/>
  </xsl:function>
  <xsl:function name="local:makePathAbsolute" as="xs:string*">
    <xsl:param name="pathTokens" as="xs:string*" />
    <xsl:param name="resultTokens" as="xs:string*" />
    <xsl:if test="false()">
      <xsl:message>+ DEBUG: local:makePathAbsolute(): Starting...</xsl:message>
      <xsl:message>+ DEBUG: pathTokens="
    <xsl:value-of select="string-join($pathTokens, ',')"/>"
    </xsl:message>

      <xsl:message>+ DEBUG: resultTokens="
    <xsl:value-of select="string-join($resultTokens, ',')"/>"
    </xsl:message>

    </xsl:if>
    <xsl:sequence select="if (count($pathTokens) = 0) then $resultTokens else if ($pathTokens[1] = '.') then local:makePathAbsolute($pathTokens[position() > 1],
    $resultTokens) else if ($pathTokens[1] = '..') then local:makePathAbsolute($pathTokens[position() > 1], $resultTokens[position() < last()]) else
    local:makePathAbsolute($pathTokens[position() > 1], ($resultTokens, $pathTokens[1]))" />
  </xsl:function>
  <!-- ***** -->
<!-- (U) IRM Phases -->
<!-- ***** -->
<!-- ***** -->

```

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

<sch:include href="./Rules/IRM_ID_00002.sch"/>
    <sch:include href="./Rules/IRM_ID_00005.sch"/>
    <sch:include href="./Rules/IRM_ID_00006.sch"/>
    <sch:include href="./Rules/IRM_ID_00007.sch"/>
    <sch:include href="./Rules/IRM_ID_00010.sch"/>
    <sch:include href="./Rules/IRM_ID_00015.sch"/>
    <sch:include href="./Rules/IRM_ID_00016.sch"/>
    <sch:include href="./Rules/IRM_ID_00017.sch"/>
    <sch:include href="./Rules/IRM_ID_00019.sch"/>
    <sch:include href="./Rules/IRM_ID_00020.sch"/>
    <sch:include href="./Rules/IRM_ID_00021.sch"/>
    <sch:include href="./Rules/IRM_ID_00022.sch"/>
    <sch:include href="./Rules/IRM_ID_00023.sch"/>
    <sch:include href="./Rules/IRM_ID_00024.sch"/>
    <sch:include href="./Rules/IRM_ID_00025.sch"/>
    <sch:include href="./Rules/IRM_ID_00029.sch"/>
    <sch:include href="./Rules/IRM_ID_00030.sch"/>
    <sch:include href="./Rules/IRM_ID_00033.sch"/>
    <sch:include href="./Rules/IRM_ID_00034.sch"/>
    <sch:include href="./Rules/IRM_ID_00036.sch"/>
    <sch:include href="./Rules/IRM_ID_00040.sch"/>
    <sch:include href="./Rules/IRM_ID_00041.sch"/>
    <sch:include href="./Rules/IRM_ID_00042.sch"/>
    <sch:include href="./Rules/IRM_ID_00043.sch"/>
    <sch:include href="./Rules/IRM_ID_00044.sch"/>
    <sch:include href="./Rules/IRM_ID_00045.sch"/>
    <sch:include href="./Rules/IRM_ID_00046.sch"/>
    <sch:include href="./Rules/IRM_ID_00047.sch"/>
    <sch:include href="./Rules/IRM_ID_00048.sch"/>
    <sch:include href="./Rules/IRM_ID_00053.sch"/>
    <sch:include href="./Rules/IRM_ID_00054.sch"/>
    <sch:include href="./Rules/IRM_ID_00055.sch"/>
    <sch:include href="./Rules/IRM_ID_00062.sch"/>
    <sch:include href="./Rules/IRM_ID_00063.sch"/>
    <sch:include href="./Rules/IRM_ID_00064.sch"/>
    <sch:include href="./Rules/IRM_ID_00065.sch"/>
    <sch:include href="./Rules/IRM_ID_00068.sch"/>
    <sch:include href="./Rules/IRM_ID_00070.sch"/>
    <sch:include href="./Rules/IRM_ID_00071.sch"/>
    <sch:include href="./Rules/IRM_ID_00072.sch"/>
    <sch:include href="./Rules/IRM_ID_00073.sch"/>
    <sch:include href="./Rules/IRM_ID_00074.sch"/>
    <sch:include href="./Rules/IRM_ID_00076.sch"/>
    <sch:include href="./Rules/IRM_ID_00077.sch"/>
    <sch:include href="./Rules/IRM_ID_00078.sch"/>
    <sch:include href="./Rules/IRM_ID_00079.sch"/>
    <sch:include href="./Rules/IRM_ID_00080.sch"/>
    <sch:include href="./Rules/IRM_ID_00081.sch"/>
    <sch:include href="./Rules/IRM_ID_00086.sch"/>
    <sch:include href="./Rules/IRM_ID_00087.sch"/>
```

```
<sch:include href="./Rules/IRM_ID_00088.sch"/>
<sch:include href="./Rules/IRM_ID_00089.sch"/>
<sch:include href="./Rules/IRM_ID_00090.sch"/>
<sch:include href="./Rules/IRM_ID_00091.sch"/>
<sch:include href="./Rules/IRM_ID_00092.sch"/>
<sch:include href="./Rules/IRM_ID_00093.sch"/>
<sch:include href="./Rules/IRM_ID_00094.sch"/>
<sch:include href="./Rules/IRM_ID_00095.sch"/>
<sch:include href="./Rules/IRM_ID_00096.sch"/>
<sch:include href="./Rules/IRM_ID_00098.sch"/>
<sch:include href="./Rules/IRM_ID_00099.sch"/>
<sch:include href="./Rules/IRM_ID_00100.sch"/>
<sch:include href="./Rules/IRM_ID_00101.sch"/>
<sch:include href="./Rules/IRM_ID_00102.sch"/>
<sch:include href="./Rules/IRM_ID_00103.sch"/>
<sch:include href="./Rules/IRM_ID_00104.sch"/>
<sch:include href="./Rules/IRM_ID_00105.sch"/>
<sch:include href="./Rules/IRM_ID_00106.sch"/>
<sch:include href="./Rules/IRM_ID_00107.sch"/>
<sch:include href="./Rules/IRM_ID_00108.sch"/>
<sch:include href="./Rules/IRM_ID_00109.sch"/>
<sch:include href="./Rules/IRM_ID_00110.sch"/>
<sch:include href="./Rules/IRM_ID_00111.sch"/>
<sch:include href="./Rules/IRM_ID_00112.sch"/>
<sch:include href="./Rules/IRM_ID_00113.sch"/>
<sch:include href="./Rules/IRM_ID_00114.sch"/>
<!--*****-->
<!-- (U) IRM Phases -->
<!--*****-->
</sch:schema>
<!--UNCLASSIFIED-->
```

Chapter 6 - Removed Rules

All of the numbered Rules for IRM that have been removed are listed in this section. This section is just a reference for what rule numbers have been dropped. In many but not all cases there will be a reason listed. In all cases the version that the rule was dropped in is listed.

6.1 - `./Rules/deleted/IRM_ID_00001.sch`

Rule Description

[IRM-ID-00001] Removed in V9 because CVEnumIRMCoverageFIPSDigraph.xml was no longer supported. Human Readable: Removed in V9.

6.2 - `./Rules/deleted/IRM_ID_00003.sch`

Rule Description

[IRM-ID-00003] Removed in V9. Human Readable: Removed in V9.

6.3 - `./Rules/deleted/IRM_ID_00004.sch`

Rule Description

[IRM-ID-00004] Removed in V4. Human Readable: Removed in V4.

6.4 - `./Rules/deleted/IRM_ID_00008.sch`

Rule Description

[IRM-ID-00008] Removed in 2016MAY.

6.5 - `./Rules/deleted/IRM_ID_00009.sch`

Rule Description

[IRM-ID-00009] Removed in 2016MAY.

6.6 - `./Rules/deleted/IRM_ID_00011.sch`

Rule Description

[IRM-ID-00011] Rule removed in V8 because it is covered by rule IRM-ID-00012. Human Readable: Rule removed in V8.

6.7 - `./Rules/deleted/IRM_ID_00012.sch`

Rule Description

[IRM-ID-00012] Removed in V9. Human Readable: Removed in V9.

6.8 - **./Rules/deleted/IRM_ID_00013.sch**

Rule Description

[IRM-ID-00013] Rule removed in V8 because it is covered by rule IRM-ID-00014. Human Readable: Removed in V8.

6.9 - **./Rules/deleted/IRM_ID_00014.sch**

Rule Description

[IRM-ID-00014] Removed in V9. Human Readable: Removed in V9.

6.10 - **./Rules/deleted/IRM_ID_00018.sch**

Rule Description

[IRM-ID-00018] Rule removed in V7.

6.11 - **./Rules/deleted/IRM_ID_00026.sch**

Rule Description

[IRM-ID-00026] Removed in V4. Human Readable: Removed in V4.

6.12 - **./Rules/deleted/IRM_ID_00027.sch**

Rule Description

[IRM-ID-00027] Removed in V6. Human Readable: Removed in V6.

6.13 - **./Rules/deleted/IRM_ID_00028.sch**

Rule Description

[IRM-ID-00028] Removed in V6. Human Readable: Removed in V6.

6.14 - **./Rules/deleted/IRM_ID_00031.sch**

Rule Description

[IRM-ID-00031] Removed in 2016MAY.

6.15 - **./Rules/deleted/IRM_ID_00032.sch**

Rule Description

[IRM-ID-00032][] Removed in V6. Human Readable: Removed in V6.

6.16 - **./Rules/deleted/IRM_ID_00035.sch**

Rule Description

[IRM-ID-00035][] Rule removed in V9. Human Readable: Rule removed in V9.

6.17 - **./Rules/deleted/IRM_ID_00037.sch**

Rule Description

[IRM-ID-00037][] Removed in V9. Human Readable: Rule removed in V9.

6.18 - **./Rules/deleted/IRM_ID_00038.sch**

Rule Description

[IRM-ID-00038][] Removed in V9. Human Readable: Rule removed in V9.

6.19 - **./Rules/deleted/IRM_ID_00039.sch**

Rule Description

[IRM-ID-00039][] Rule removed in V9. Human readable: Rule removed in V9.

6.20 - **./Rules/deleted/IRM_ID_00049.sch**

Rule Description

[IRM-ID-00049][] Removed in 2016MAY, replaced by IRM-ID-00090

6.21 - **./Rules/deleted/IRM_ID_00050.sch**

Rule Description

[IRM-ID-00050][] Rule removed in V2019-MAR due to DDMS merging into IRM. Human Readable: Rule removed in V2019-MAR due to DDMS merging into IRM.

6.22 - **./Rules/deleted/IRM_ID_00051.sch**

Rule Description

[IRM-ID-00051] Rule removed in V2019-MAR due to DDMS merging into IRM. Human Readable: Rule removed in V2019-MAR due to DDMS merging into IRM.

6.23 - **./Rules/deleted/IRM_ID_00052.sch**

Rule Description

[IRM-ID-00052] Removed in version 2021-NOV. All @irm:acronym now require a country component.

6.24 - **./Rules/deleted/IRM_ID_00056.sch**

Rule Description

[IRM-ID-00056] Rule removed in V9. Human Readable: Rule removed in V9.

6.25 - **./Rules/deleted/IRM_ID_00057.sch**

Rule Description

[IRM-ID-00057] Rule introduced in V8 and removed during V8 CR adjudication. Never part of a signed release. Human readable: Rule removed during V8 CR adjudication.

6.26 - **./Rules/deleted/IRM_ID_00058.sch**

Rule Description

[IRM-ID-00058] Rule introduced in V8 and removed during V8 CR adjudication. Never part of a signed release. Human Readable: Rule removed during V8 CR adjudication.

6.27 - **./Rules/deleted/IRM_ID_00059.sch**

Rule Description

[IRM-ID-00059] Removed in version 2019-MAR. The requirements for children of temporalCoverage are now enforced in schema.

6.28 - **./Rules/deleted/IRM_ID_00060.sch**

Rule Description

[IRM-ID-00060] Rule introduced in V8 and removed during V8 CR adjudication. Never part of a signed release. Human Readable: Rule removed during V8 CR adjudication.

6.29 - **./Rules/deleted/IRM_ID_00061.sch**

Rule Description

[IRM-ID-00061][] Rule removed in V9. Human Readable: Rule removed in V9.

6.30 - **./Rules/deleted/IRM_ID_00066.sch**

Rule Description

[IRM-ID-00066][] Removed in V9. Human Readable: Rule removed in V9.

6.31 - **./Rules/deleted/IRM_ID_00067.sch**

Rule Description

[IRM-ID-00067][] Removed in V9. Human Readable: Rule removed in V9.

6.32 - **./Rules/deleted/IRM_ID_00069.sch**

Rule Description

[IRM-ID-00069][] Rule removed in V9 because the otherNetwork attribute was removed. Networks Names are now controlled through the VIRT CVE, which uses a regex for other network names. Human Readable: Rule removed in V9.

6.33 - **./Rules/deleted/IRM_ID_00075.sch**

Rule Description

[IRM-ID-00075][] Removed in version 2019-MAR. The requirements for children of temporalCoverage are now enforced in schema.

6.34 - **./Rules/deleted/IRM_ID_00082.sch**

Rule Description

[IRM-ID-00082][] Rule removed in V2019-MAR due to DDMS merging into IRM. Human Readable: Rule removed in V2019-MAR due to DDMS merging into IRM.

6.35 - **./Rules/deleted/IRM_ID_00083.sch**

Rule Description

[IRM-ID-00083][] Rule removed in V2019-MAR due to DDMS merging into IRM. Human Readable: Rule removed in V2019-MAR due to DDMS merging into IRM.

6.36 - ~~./Rules/deleted/IRM_ID_00084.sch~~

Rule Description

[IRM-ID-00084] Removed in version 2019-MAR. Production Metrics moved out of IRM into its own assertion.

6.37 - ~~./Rules/deleted/IRM_ID_00085.sch~~

Rule Description

[IRM-ID-00085] Removed in version 2019-MAR. Production Metrics moved out of IRM into its own assertion.

6.38 - ~~./Rules/deleted/IRM_ID_00097.sch~~

Rule Description

[IRM-ID-00097] Removed in version 2021-NOV. With multiple TDF to pick from, no longer can validate environment against a specific TDF spec.