Guide to Schematron Rules and Patterns

MN 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 - Purpose

This is an informative supplement for MN. This guide is generated from the MN 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 MN. 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 MN. The abstract patterns may be used in numbered rules or provided as reference for use in rules developed by users of MN. Each abstract pattern has a code description and a code block with the abstract Schematron pattern.

Chapter 4 is a listing of the master MN 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 MN are encoded using ISO Schematron. Schematron is a rule-based validation language that uses XML Path Language to make assertions about an XML document.

MN 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 MN's Schematron rules.

Chapter 2 - Rules

All of the numbered Rules for MN 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 MN-ID-XXXXX, with rule files named MN_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/MN_ID_00001.sch

Rule Description

[MN-ID-00001][Warning] mn:CESVersion 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="MN-ID-00001">
            <sch:rule id="MN-ID-00001-R1" context="*[@mn:CESVersion]">
                <sch:assert test="matches(@mn:CESVersion,'^202111(\-.{1,23})?$')"
                            flag="warning"
                            role="warning">[MN-ID-00001][Warning] mn:CESVersion attribute SHOULD be specified as version 202111 (Version:2021-NOV) with an optional extension. Found :
      <sch:value-of select="./@mn:CESVersion"/>
                </sch:assert>
            </sch:rule>
        </sch:pattern>
```

Chapter 3 - Abstract Patterns

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

3.1 - .//Lib/AllowableIssueValue.sch

Code Description

This abstract pattern checks to see if a value exists in the Issue Acronym List. The following parameters are used by this pattern: $context := the context in which the searchValue exists. $searchTerm := the value which you want to verify is in the list. $errMsg := the error message text to display when the assertion fails.

Schematron Code

```
<sch:pattern abstract="true" id="AllowableIssueValues">
        <sch:rule context="$context">
            <sch:let name="Issue"
                      value="document('../../CVE/MN/CVEnumMNIssue.xml')//cve:CVE/cve:Enumeration/cve:Term/cve:Value"/>
            <sch:assert test="some $token in $Issue satisfies $token = $searchTerm or matches($searchTerm, concat('^',$token,'$'))"
                      flag="error">
                <sch:value-of select="$errMsg"/>
            </sch:assert>
        </sch:rule>
</sch:pattern>
```

3.2 - .//Lib/AllowableRegionValue.sch

Code Description

This abstract pattern checks to see if a value exists in the Region Acronym List. The following parameters are used by this pattern: $context := the context in which the searchValue exists. $searchTerm := the value which you want to verify is in the list. $errMsg := the error message text to display when the assertion fails.

Schematron Code

```
<sch:pattern abstract="true" id="AllowableRegionValues">
        <sch:rule context="$context">
            <sch:let name="Region"
                     value="document('../../CVE/MN/CVEnumMNRegion.xml')//cve:CVE/cve:Enumeration/cve:Term/cve:Value"/>
            <sch:assert test="some $token in $Region satisfies $token = $searchTerm or matches($searchTerm, concat('^',$token,'$'))"
                        flag="error">
                <sch:value-of select="$errMsg"/>
            </sch:assert>
        </sch:rule>
    </sch:pattern>
```

Chapter 4 - Schematron Schema

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

4.1 - .//MN_XML.sch

Schematron Code

```xml
<!--UNCLASSIFIED-->
<?ICEA master?>
<!-- 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 queryBinding="xslt2">
                    <sch:ns uri="urn:us:gov:ic:mn" prefix="mn"/>
                    <sch:ns uri="urn:us:gov:ic:region" prefix="region"/>
                    <sch:ns uri="urn:us:gov:ic:edh:xsl:util" prefix="util"/>
                    <sch:ns uri="http://www.w3.org/2001/XMLSchema" prefix="xs"/>
                    <!--====================-->
<!-- (U) Universal Lets -->
<!--====================-->
<!-- ************************************* -->
<!-- * Abstract Rule and Pattern Includes * -->
<!-- ************************************* -->

<sch:include href="Lib/AllowableRegionValue.sch"/>
                    <sch:include href="Lib/AllowableIssueValue.sch"/>
                    <!--*****************************-->
<!-- (U) MN Phases -->
<!--*****************************-->
<!--*****************************-->
<!-- (U) MN ID Rules -->
<!--*****************************-->
<!--(U) -->

<sch:include href="./Rules/MN_ID_00001.sch"/>
                    <!--*****************************-->
<!-- (U) MN Phases -->
<!--*****************************-->
</sch:schema>
                <!--UNCLASSIFIED-->
```

Chapter 5 - Removed Rules

There are no rules that have been removed for MN.