UK GEMINI Schematron Schema Guidance

An introduction to the UK GEMINI 2 Schematron Schema

Edition 1.3
February 2013

uk location

DOCUMENT CONTROL

Change Summary

Version	Date	Author/Editor	Change Summary
1.0	30/09/2010	Tim Manning	Base-lined version. Released through UK Location Resource Centre for 'beta' evaluation by UK Location Data Providers, Publishers and their suppliers, as part of deployment of UK Location Discovery Metadata Service.
1.1	24/01/2011	James Rapaport	Minor edits; change of copyright statement
1.2	09/05/2011	James Rapaport	Following changes to Schematron schema
1.3	25/02/2013	Peter Parslow	UKLP's own "ISO 19139 Table A" schematron

References

Ref.	Title/Version/Publication Date/Author
[1]	UK GEMINI Standard, Version 2.2, December 2012, AGI
[2]	UK GEMINI Encoding Guidance, Version 1.4a, 2012-01-24, UKLP (MWG)
[3]	DMS Operational Guidance, version 2.2a, 2012-01-24, UKLP (MWG)

CONTENTS

1 INTRODUCTION

1.1 Purpose of document

1 The purpose of this document is to provide an introduction to the GEMINI2 Schematron schema.

2 The UK Location GEMINI2 Schematron schema is provided at version 1.3 to facilitate the validation of UK Location GEMINI2 constraints.

3 The Schematron schema was based on GEMINI2.1, and requires no change for GEMINI 2.2. It is designed to validate metadata instances encoded according to the UKLP encoding guidance [2].

1.2 Assumed Knowledge

4 It is assumed that readers will be familiar with XML and its related technologies: XSD and XSL. Readers who require background information are referred to the W3Schools introductions to the technologies:

- XML - http://www.w3schools.com/xml/xml_whatis.asp
- XSD - http://www.w3schools.com/schema/schema_intro.asp
- XSL - http://www.w3schools.com/xsl/xsl_languages.asp

1.3 Terminology

ISO

5 International Organisation for Standardisation

MEDIN

6 Marine Environmental Data and Information Network

SVRL

7 Schematron Validation Report Language

XML

8 eXtensible Markup Language

XPath

9 XML Path Language

XSD

10 XML Schema Definition

XSL

11 eXtensible Stylesheet Language

XSLT

12 XSL Transformation

1.4 Schematron changes

13 It was recognised that GEMINI 2.1 was more strict in terms of its constraints than the INSPIRE metadata profile, particularly in respect of restricting the multiplicity on some metadata items. Therefore, while GEMINI 2.1 metadata instances could always claim INSPIRE conformance, it was not always certain that INSPIRE metadata instances would conform to GEMINI. This was problematic for some implementers who wanted to implement systems based on GEMINI 2.1 but also to exchange metadata with INSPIRE based systems.

14 The constraints in the Schematron schema at version 1.2 were relaxed in order that the schema does not reject valid INSPIRE metadata instances. Consequently, it is possible for a metadata instance to contain multiple instances of certain elements (e.g. geographic bounding box) where the GEMINI standard constraints were more strict. These multiplicity relaxations were adopted as improvements for GEMINI in GEMINI 2.2.

15 The UK Location profile of GEMINI, documented in the DMS Operational Guide, always required a metadata language – which is optional in GEMINI. Version 1.3 of the schematron enforces this such that metadata language must be supplied or the record will fail validation.

16 Changes to the Schematron schema between versions 1.1, 1.2, and 1.3 are detailed in Appendix C (see Table 4).

1.5 Acknowledgements

17 This document was originally written by James Rapaport of SeaZone Solutions Limited, now SeaZone Solutions.

2 SCHEMATRON

2.1 Introduction

18 ISO 19115 defines a set of elements for recording metadata and sets out a minimal set of constraints which must be achieved by all compliant profiles. A profile is a domain specific metadata set which can apply further constraints to ISO 19115 to achieve particular requirements. ISO 19139 is a technical specification which defines a set of schemas in XML Schema Definition language which set out an XML grammar for encoding ISO 19115 metadata in XML. XML can be assessed against schemas to ensure that the structure of the XML conforms to the structure defined in the schema. When the structure does conform the XML is said to be schema valid. A general limitation of XML schemas is that they are grammar based which means that they do not provide a means of validating element values or domain specific profiles of ISO 19115.

19 Schematron provides another way of validating XML by looking for tree patterns and element content. Schematron works by making assertions about the XML which resolve to true or false. If an assertion resolves to false it fails and the overall validation fails. The assertions are written using XPath in a Schematron schema (*.sch) which is itself expressed in XML. Schematron is designed to be used in conjunction with other validation processes and is based on XSLT and XPath so it is simple to implement.

20 Indeed, the ISO / TS 19139 standard refers to Schematron as a means of testing some constraints that are set by ISO 19115 but are not testable with XSD. These constraints are set out in Table A.1 of ISO / TS 19139.

21 Schematron has been standardised by ISO as ISO 19757 Part 3. It is this version of Schematron that has been used to create the GEMINI2 Schematron schema.

22 The home of Schematron is http://www.schematron.com/.

23 This report documents the Schematron schema for the validation of XML metadata sets encoded in ISO 19139 XML according to the GEMINI2 metadata profile.

2.2 Technical Specification

24 The GEMINI2 Schematron schema has been written for ISO 19757-3 Schematron and XSLT version 1.0. The schema will work with the ISO Schematron for XSLT 1.0 stylesheets.[1]

[1] http://www.schematron.com/tmp/iso-schematron-xslt1.zip

2.3 Validation

2.3.1 Schematron Stylesheets

25 Schematron is implemented as a concatenated series of XSL transformations. A set of XSL stylesheets is provided on the Schematron web site which underpin the validation process. The stylesheets are downloaded in a Zip archive and should be extracted into a common folder. The GEMINI2 Schematron schema is written for XSLT version 1.0 and the ISO Schematron stylesheets. Therefore it is the XSLT version 1.0 stylesheets that should be downloaded and used. Table 1 lists each of the stylesheets in the set.

Table 1 – ISO Schematron Stylesheets

Stylesheet Name	Description
iso_dsdl_include.xsl	This stylesheet is used to assemble a Schematron schema from various parts. The GEMINI2 Schematron schema is not in separate parts so this stylesheet is not used.
iso_abstract_expand.xsl	This stylesheet converts abstract patterns in a Schematron schema into real patterns. The GEMINI2 Schematron schema does contain abstract patterns so this stylesheet must be applied first.
iso_svrl_for_xslt1.xsl	This stylesheet is used to convert a Schematron schema into an XSLT stylesheet. The GEMINI2 Schematron schema must be processed against iso_svrl_for_xslt1.xsl, which in turn invokes iso_schematron_skeleton_for_xslt1.xsl.
iso_schematron_skeleton_for_xslt1.xsl	This stylesheet is used in the process to convert a Schematron schema into an XSLT stylesheet. It is invoked by iso_svrl_for_xslt1.xsl.
ExtractSchFromRNG.xsl	This stylesheet is used to generate a Schematron schema from Schematron patterns embedded in a RELAX NG schema. It is not used in the GEMINI2 validation process.
ExtractSchFromXSD.xsl	This stylesheet is used to generate a Schematron schema from Schematron patterns embedded in a W3C XML Schemas

Stylesheet Name	Description
	(XSD) schema. It is not used in the GEMINI2 validation process.
iso_schematron_message.xsl	Not used in the GEMINI2 validation process.

2.3.2 Schematron Validation Process

26 The process is shown in Figure 1. The red boxes represent XSL stylesheets that are provided as part of Schematron. The blue box represents the GEMINI2 Schematron schema. The blue stars represent the occurrence of an XSL transformation. The Schematron schema is first styled with the iso_abstract_expand.xsl stylesheet to produce Intermediate.sch. Schematron allows for the creation of abstract patterns which are encoded in a schema once but used many times. These patterns must be physically instantiated at run time and this is accomplished by this step. The Intermediate.sch is a Schematron schema. It may be saved to disk or held in memory. Intermediate.sch is then styled with iso_svrl_for_xslt1.xsl to produce TheSchema.xsl. TheSchema.xsl is in fact a stylesheet which is used to style a metadata dataset, represented here by IsoMetadataInstance.xml. This step is where the 'validation' occurs and the output is TheResult.xml which is encoded in SVRL.

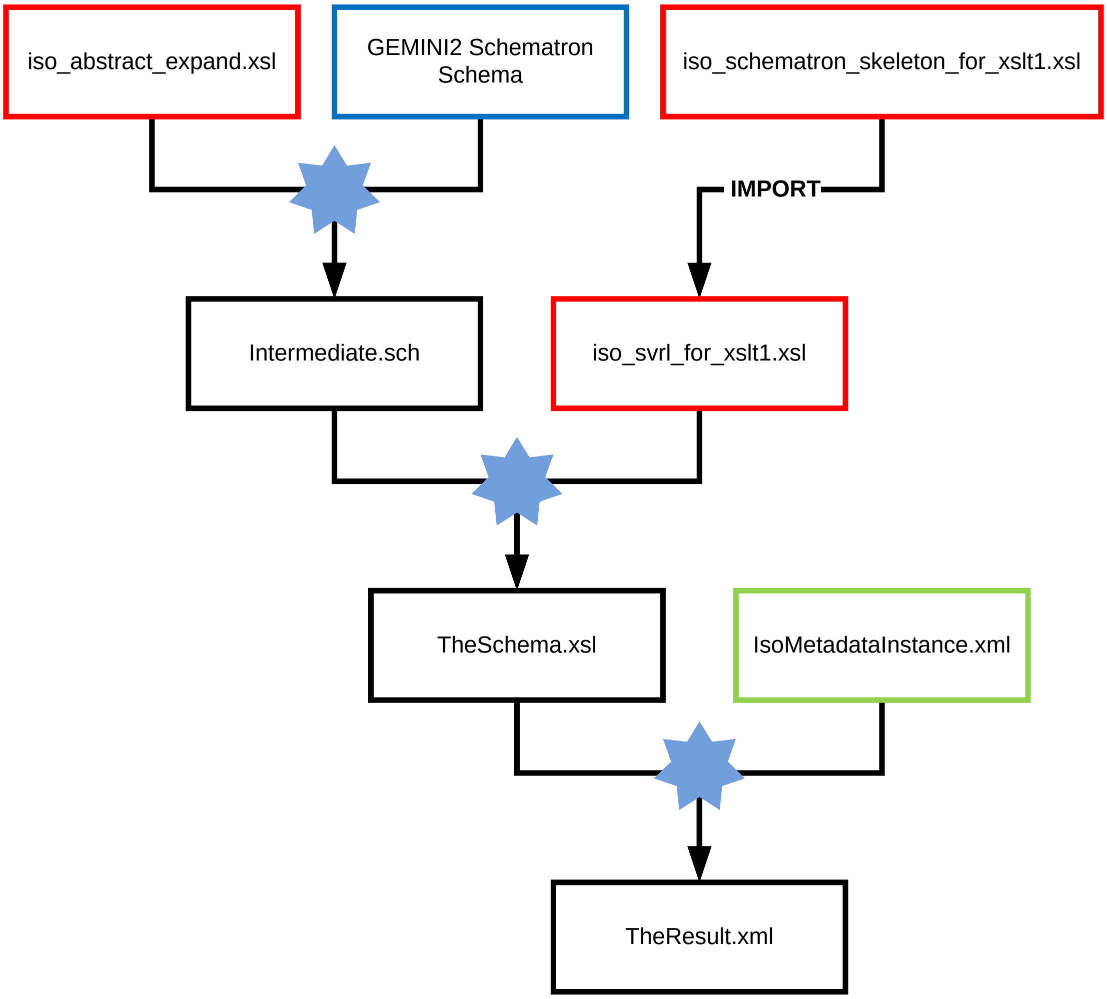

Figure 1 – Schematron Stylesheet Transformation Steps

2.3.3 Overall metadata validation process

27 With the introduction of the Schematron schema to validate the profile, the overall metadata validation workflow becomes a three stage process as shown in Figure 2. First, a candidate metadata set must be validated against the ISO 19139 schemas. If the metadata set proves to be schema valid, it can then be validated against the ISO 19139 Table A.1 Constraints Schematron schema. The Schematron schema relies on hardcoded XPath statements which will only work effectively on a schema valid XML set. Finally, if the XML is still valid it can be validated against the GEMINI2 UK Location Profile Schematron schema.

28 UK Location provides an ISO constraints Schematron schema which is available online.[2] It is based on one developed by MEDIN.

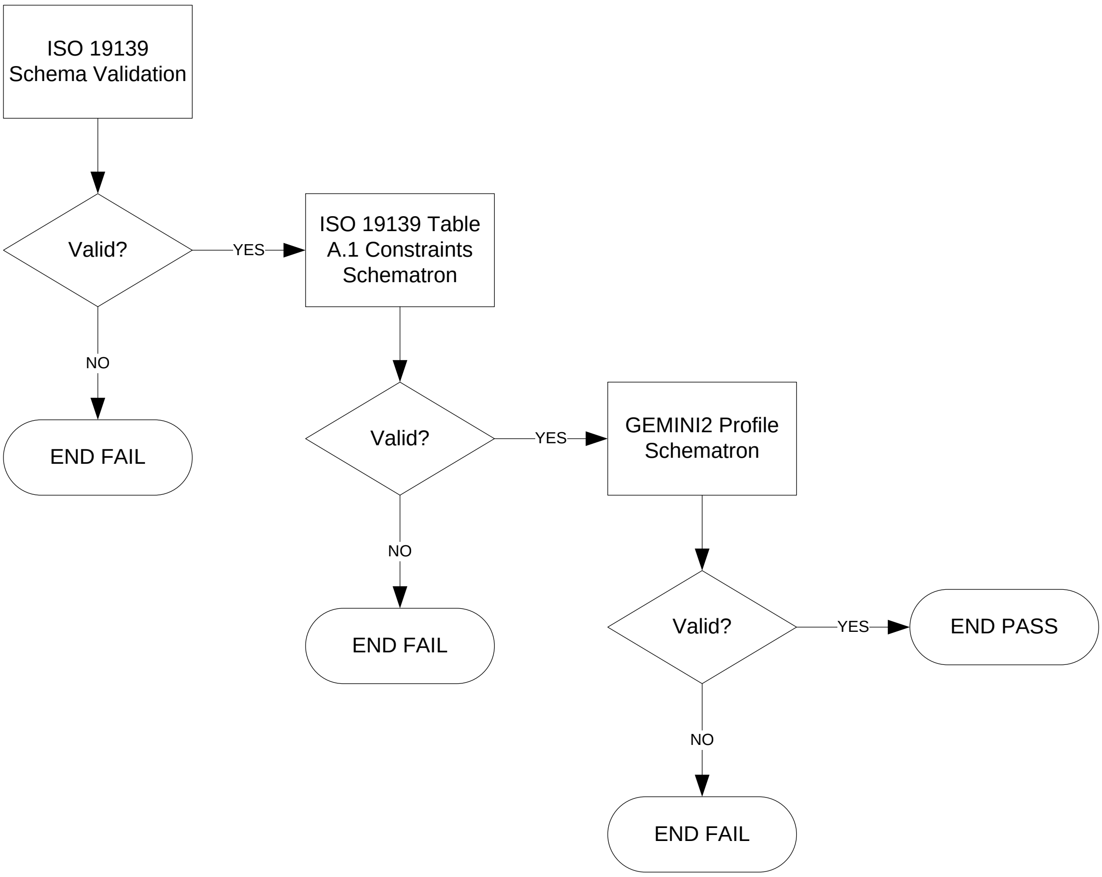
ISO 19139 Schema Validation
Valid?
YES
NO
END FAIL
ISO 19139 Table A.1 Constraints Schematron
Valid?
YES
NO
END FAIL
GEMINI2 Profile Schematron
Valid?
YES
END PASS
NO
END FAIL

Figure 2 – Metadata Validation Workflow

[2] http://data.gov.uk/library/uk-location-schematron-schema-constraints

2.4 Schematron components

2.4.1 Patterns

29 A pattern in Schematron is a high level structure for encapsulating a set of rules that are in some way related. Rules contain either assertions or reports both of which express tests written in XPath. An example of a pattern is shown in Figure 3. This pattern includes a title and a single rule. The rule contains two assertions and two reports which work in conjunction with each other so that if an assertion passes (returns true) the report also passes. In this case the value of the report gets written to the SVRL file so that it is obvious that a test has fired and passed. If the assertion fails its value is written to the SVRL file and the report does not fire. It is recommended that assertions and reports are created in pairs like this so that the output of a Schematron validation process assists in the development of an XML metadata instance. Failed assertions imply non-conformity with the profile. Reports are included for information.

30 The pattern shown in Figure 3 tests for the metadata item Topic Category. The first rule element contains an assert with a test that ensures that one or more gmd:topicCategory elements is encoded where the gmd:hierarchyLevel element has a code list value of 'dataset' or 'series'. The second assert element contains a test to ensure that the value of any gmd:topicCategory element is not null.

```
<!-- ==================================================================== -->
<!-- Metadata Item 5 - Topic Category                                     -->
<!-- ==================================================================== -->
<sch:pattern fpi="Gemini2-mi5">
  <sch:title>Topic Category</sch:title>
  <sch:rule context="/*/gmd:identificationInfo[1]/*">
    <sch:assert test="((../../gmd:hierarchyLevel[1]/*/@codeListValue =
'dataset' or ../../gmd:hierarchyLevel[1]/*/@codeListValue = 'series') and
count(gmd:topicCategory) >= 1) or
(../../gmd:hierarchyLevel[1]/*/@codeListValue != 'dataset' and
../../gmd:hierarchyLevel[1]/*/@codeListValue != 'series') or
count(../../gmd:hierarchyLevel) = 0">
        Topic category is mandatory for datasets and series. One or more
shall be provided.
    </sch:assert>
  </sch:rule>
  <sch:rule context="/*/gmd:identificationInfo[1]/*/gmd:topicCategory">
    <sch:assert test="((../../../gmd:hierarchyLevel[1]/*/@codeListValue =
'dataset' or ../../../gmd:hierarchyLevel[1]/*/@codeListValue = 'series') and
count(@gco:nilReason) = 0) or
(../../../gmd:hierarchyLevel[1]/*/@codeListValue != 'dataset' and
../../../gmd:hierarchyLevel[1]/*/@codeListValue != 'series') or
count(../../../gmd:hierarchyLevel) = 0">
      Topic Category shall not be null.
    </sch:assert>
  </sch:rule>
</sch:pattern>
```

Figure 3 – Schematron Pattern

2.4.2 Abstract Patterns

31 Abstract patterns allow the definition of patterns which can be used to test identical structures in XML that are used in different contexts. Examples of identical structures used in this way in ISO 19115 metadata are the geographic bounding box and responsible party data. Abstract patterns provide an efficient mechanism for checking the common properties of identical structures because the pattern is written only once in the schema but can be used many times.

32 Table 2 shows a pattern that invokes an abstract pattern. The pattern is a (indicated by the `is-a` attribute) type of TypeNotNillablePattern, which is an abstract pattern and is shown in Table 3. The pattern passes a parameter (`sch:param`) with the name 'context' to the abstract pattern. The value of the parameter in this case is the XPath of the XML element used to express the GEMINI2 metadata item Resource Title. The TypeNotNillablePattern is used to ensure that metadata elements have a value and are not given a nil reason. It is used many times in different contexts which is why is implemented as an abstract pattern.

Table 2 – Schematron pattern invoking an abstract pattern

```
<sch:pattern is-a="TypeNotNillablePattern" id="Gemini2-mi1-NotNillable">
  <sch:param name="context"
value="/*/gmd:identificationInfo[1]/*/gmd:citation/*/gmd:title"/>
</sch:pattern>
```

Table 3 – Schematron abstract pattern

```
<!-- Test that an element has a value - the value is not nillable -->
<sch:pattern abstract="true" id="TypeNotNillablePattern">
  <sch:rule context="$context">
    <sch:assert test="string-length(.) &gt; 0 and count(./@gco:nilReason) = 0">
      The <sch:name/> element is not nillable and shall have a value.
    </sch:assert>
  </sch:rule>
</sch:pattern>
```

2.5 Validation Output

33 The results of a validation process are output as SVRL. SVRL is a report language defined as part of ISO Schematron. SVRL is encoded in XML and provides a set of information resulting from a validation. An example of an SVRL instance is shown and described in Appendix B.

3 TOOLS

34 The following applications are known to implement some form of Schematron validation but the GEMINI2 Schematron schema has not been tested with them:

- Oxygen XML Editor - http://www.oxygenxml.com
- GeoNetwork Opensource - http://geonetwork-opensource.org/

35 In addition any library implementing XSLT 1.0 can also be used to perform Schematron validation.

APPENDIX A – SCHEMATRON SCHEMA EXAMPLE

An example of a Schematron schema is presented below. It contains two concrete patterns, one set to produce successful output and one known to produce unsuccessful output, given a particular XML instance to validate. The patterns include assert elements and report elements which include test attributes and text which will be output to the SVRL in the event that the test returns true. Note that it is possible to include references which are resolved at runtime in the text using elements such as value-of.

An abstract pattern is also included in the example. Abstract patterns are not dissimilar to concrete patterns. The key differences are that they have an abstract attribute which must be set to true and they take parameters which are identified by the dollar sign. Concrete patterns implement the abstract pattern and when this is done the concrete pattern must include an is-a attribute which identifies its base abstract pattern. The param elements set the value of the abstract pattern parameters.

```
<?xml version="1.0" encoding="utf-8" ?>
<sch:schema xmlns:sch="http://purl.oclc.org/dsdl/schematron"
queryBinding="xslt">

  <!-- Namespaces from ISO 19139 Metadata encoding -->
  <sch:ns prefix="gml" uri="http://www.opengis.net/gml/3.2" />
  <sch:ns prefix="gmd" uri="http://www.isotc211.org/2005/gmd"/>
  <sch:ns prefix="gco" uri="http://www.isotc211.org/2005/gco"/>
  <sch:ns prefix="gmx" uri="http://www.isotc211.org/2005/gmx"/>
  <sch:ns prefix="xlink" uri="http://www.w3.org/1999/xlink"/>

  <!-- ======================================================== -->
  <!-- Concrete Pattern - Set up to produce successful reports  -->
  <!-- ======================================================== -->

  <sch:pattern fpi="ExampleConcretePatternPass">
    <sch:title>Example Pattern - Pass</sch:title>
    <sch:rule context="//gmd:CI_ResponsibleParty">
      <sch:assert test="count(gmd:organisationName) +
                  count(gmd:individualName) &gt;= 1">
        At least one organisation name and / or individual
        name must be provided.
      </sch:assert>
      <sch:report test="count(gmd:organisationName) = 1">
        The value of organisationName is
        '<sch:value-of select="gmd:organisationName"/>'.
      </sch:report>
      <sch:report test="count(gmd:individualName) = 1">
        The value of individualName is
        '<sch:value-of select="gmd:individualName"/>'.
      </sch:report>
    </sch:rule>
  </sch:pattern>

  <!-- ======================================================== -->
  <!-- Concrete Pattern - Set up to produce failed asserts      -->
  <!-- ======================================================== -->

  <sch:pattern fpi="ExampleConcretePatternFail">
    <sch:title>Example Pattern - Fail</sch:title>
    <sch:rule context="//gmd:CI_ResponsibleParty">
      <sch:assert test="count(gmd:organisationName) +
                  count(gmd:individualName) &gt;= 10">
        At least 10 organisation names and / or individual
        names must be provided. Only
        <sch:value-of select="count(gmd:organisationName) +
                              count(gmd:individualName)"/> found.
      </sch:assert>
    </sch:rule>
  </sch:pattern>

  <!-- ======================================================== -->
  <!-- Concrete pattern implementing abstract pattern           -->
  <!-- ======================================================== -->

  <sch:pattern is-a="GcoTypeTestPattern" id="ExampleGcoTypeTest">
    <sch:title>Example Pattern - Call Abstract Pattern</sch:title>
    <sch:param name="context"
           value="//gmd:identificationInfo/*/gmd:citation/*/gmd:title"/>
    <sch:param name="element"
             value="gco:CharacterString"/>
  </sch:pattern>

  <!-- ==================================================== -->
  <!-- Abstract Pattern                                     -->
  <!-- ==================================================== -->

  <sch:pattern abstract="true" id="GcoTypeTestPattern">
    <sch:rule context="$context">
      <sch:assert test="(string-length($element) &gt; 0) or
                  (@gco:nilReason = 'inapplicable' or
                  @gco:nilReason = 'missing' or
                  @gco:nilReason = 'template' or
                  @gco:nilReason = 'unknown' or
                  @gco:nilReason = 'withheld')">
        The <sch:name/> element must have a value or a Nil Reason.
      </sch:assert>
      <sch:report test="(string-length($element) &gt; 0)">
        The <sch:name/> element has a value of
        '<sch:value-of select="$element"/>'.
      </sch:report>
      <sch:report test="(@gco:nilReason = 'inapplicable' or
                  @gco:nilReason = 'missing' or
                  @gco:nilReason = 'template' or
                  @gco:nilReason = 'unknown' or
                  @gco:nilReason = 'withheld')">
        The <sch:name/> element has a Nil Reason attribute with a value of
        '<sch:value-of select="@gco:nilReason"/>'.
      </sch:report>
    </sch:rule>
  </sch:pattern>
</sch:schema>
```

APPENDIX B – SVRL

The following is the output from the validation of an ISO 19139 encoded XML file using the Schematron schema shown in Appendix A. Note that it is incomplete and that some report elements have been replaced with an ellipsis to make it more easily readable. The data indicates that the pattern entitled "Example Pattern – Pass" has fired (see bold element active-pattern). The bold successful-report element indicates that the report with test count(gmd:organisationName) = 1 has returned successfully. Its text element indicates the output and that the value of the gmd:organisationName that was found is 'SeaZone Solutions Limited'. The full version of the file then contains a series of successful-report elements.

Early versions of the GEMINI2 Schematron schema used report elements to output metadata element values to provide user feedback. The successful-report element was interpreted as a validation pass rather than fail. However, subsequently it was found that the Oxygen XML[3] interprets successful-report elements in SVRL as validation failures. The intention is that the Schematron schema will be useable in a variety of systems so it was decided that the best course of action would be to remove all report elements from the schema. Consequently, patterns in the GEMINI2 schema now use assert elements only.

When the active pattern becomes "Example Pattern – Fail", tests begin to fail. This is identified from the presence of the failed-assert element. The test here is that the count of gmd:organisationName and gmd:individualName for a particular gmd:CI_ResponsibleParty element must be at least 10 (the &gt; escape sequence is interpreted in XML as >). The text element reports the failure and also the total number of the respective gmd elements that were found.

```
<?xml version="1.0" encoding="utf-16" standalone="yes"?>
<svrl:schematron-output title="" schemaVersion=""
                        xmlns:svrl="http://purl.oclc.org/dsdl/svrl"
                        xmlns:xs="http://www.w3.org/2001/XMLSchema"
                        xmlns:schold="http://www.ascc.net/xml/schematron"
                        xmlns:sch="http://www.ascc.net/xml/schematron"
                        xmlns:iso="http://purl.oclc.org/dsdl/schematron"
                        xmlns:gml="http://www.opengis.net/gml"
                        xmlns:gmd="http://www.isotc211.org/2005/gmd"
                        xmlns:gco="http://www.isotc211.org/2005/gco"
                        xmlns:gmx="http://www.isotc211.org/2005/gmx"
                        xmlns:xlink="http://www.w3.org/1999/xlink">

  <svrl:ns-prefix-in-attribute-values
      uri="http://www.opengis.net/gml" prefix="gml" />
<svrl:ns-prefix-in-attribute-values
      uri="http://www.isotc211.org/2005/gmd" prefix="gmd" />
  <svrl:ns-prefix-in-attribute-values
      uri="http://www.isotc211.org/2005/gco" prefix="gco" />
  <svrl:ns-prefix-in-attribute-values
      uri="http://www.isotc211.org/2005/gmx" prefix="gmx" />
  <svrl:ns-prefix-in-attribute-values
      uri="http://www.w3.org/1999/xlink" prefix="xlink" />

  <svrl:active-pattern name="Example Pattern - Pass" />
  <svrl:fired-rule context="//gmd:CI_ResponsibleParty" />
  <svrl:successful-report test="count(gmd:organisationName) = 1"
```

[3] http://www.oxygenxml.com

```
                          location="/*[local-name()='MD_Metadata' and namespace-
uri()='http://www.isotc211.org/2005/gmd']/*[local-name()='contact' and
namespace-uri()='http://www.isotc211.org/2005/gmd']/*[local-
name()='CI_ResponsibleParty' and namespace-
uri()='http://www.isotc211.org/2005/gmd']">
    <svrl:text>
        The value of organisationName is 'SeaZone Solutions Limited'.
      </svrl:text>
  </svrl:successful-report>
  ...
  <svrl:active-pattern name="Example Pattern - Fail" />
  <svrl:fired-rule context="//gmd:CI_ResponsibleParty" />
  <svrl:failed-assert test="count(gmd:organisationName) +
count(gmd:individualName) &gt;= 10"
                      location="...">
    <svrl:text>
        At least 10 organisation names and / or individual
        names must be provided. Only 2 found.
      </svrl:text>
  </svrl:failed-assert>
  ...
  <svrl:active-pattern id="ExampleGcoTypeTest" name="ExampleGcoTypeTest" />
  <svrl:fired-rule context="//gmd:identificationInfo/*/gmd:citation/*/gmd:title"
/>
  <svrl:successful-report test="(string-length(gco:CharacterString) &gt; 0)"
                          location="...">
    <svrl:text>
        The gmd:title element has a value of 'A Dataset Title'.
      </svrl:text>
  </svrl:successful-report>
</svrl:schematron-output>
```

APPENDIX C – CHANGES TO THE SCHEMATRON SCHEMA

36 This section lists changes that have been made to the Schematron schema between version 1.1 and 1.3 of the schema. Line numbers listed in the Change column refer to version 1.1 of the schema, except for the change from version 1.2 to 1.3, where the line number refers to version 1.2.

Table 4 – Changes at version 1.2 and 1.3

Number	Name	Change	Notes
7	Temporal Extent	Line 166 – remove rule	GEMINI multiplicity is single. INSPIRE multiplicity is 0..*
		Line 176 – amend assert for gml:TimePeriod or gml:TimeInstant	To allow for a temporal extent to be encoded as a single date
		Line 183 – remove assert	INSPIRE allows temporal extent to be encoded using a gml:TimePeriod element. Its children are [gml:begin or gml:beginPosition] and [gml:end or gml:endPosition]. INSPIRE allows any of these children (see A.9 of the ISO 19115 encoding guidelines from INSPIRE).
11	Geographic bounding box	Line 227 – change assert to allow one or more	INSPIRE multiplicity is 1..* for datasets and series, and 0..* for services (mandatory where the service has an explicit geographic extent). The GEMINI multiplicity was 1.
16	Vertical extent information	Line 285 – remove rule	GEMINI states that this is optional and the multiplicity is single. In ISO 19115 the multiplicity is 0..*. Vertical extent is not defined as part of INSPIRE but it is not excluded from INSPIRE either so an INSPIRE instance may have many vertical extent elements.
17	Spatial reference system	Line 302 – remove rule ensuring that one spatial reference system is provided if the metadata is for a dataset or series.	GEMINI states that this is mandatory for datasets and series, and that the multiplicity is single. In ISO 19115 the multiplicity is 0..*. Spatial reference system is not a part of INSPIRE but it is not excluded from INSPIRE either so an INSPIRE instance may have many spatial reference system elements.
18	Spatial resolution	Line 323 – remove rule	GEMINI states that the distance measurement is to be given in units of metres. INSPIRE has no such restriction on the units of the distance measurement.
24	Frequency of update	Line 403 – remove rule ensuring that one frequency of update is	GEMINI states that the obligation on this item is mandatory and that the multiplicity is single. In ISO 19115 the multiplicity is

Number	Name	Change	Notes
		provided if the metadata is for a dataset or series.	0..*. This item does not occur in INSPIRE but it is not excluded from INSPIRE either so an INSPIRE instance may have many frequency of update elements.
25	Unique resource identifier	Line 475 – change sch:assert test attribute to count >= 1	GEMINI states that the multiplicity of this item is single. In INSPIRE the multiplicity is 1..*.
43	Equivalent scale	Line 556 – remove rule	GEMINI states that the multiplicity of this item is 0..1. In INSPIRE the multiplicity is 0..*.
38	Coupled resource	Line 643 – remove rule	Coupled resource is effectively mandatory for view and download services because the link is made via dataset metadata and it is mandatory to make that dataset metadata available. The INSPIRE metadata regulation for spatial data services states that coupled resource (with a multiplicity of 0..*) is mandatory if a linkage to datasets on which the service operates is available.
In version 1.3			
33	Metadata language	Line 553 – insert rule	Inserted additional test to check that exactly one metadata language is present.