Validation Language

Submitted To: Program Manager
GeoConnections
Victoria, BC, Canada

Submitted By: Jody Garnett
Brent Owens
Refractions Research Inc.
Suite 400, 1207 Douglas Street
Victoria, BC, V8W-2E7
jgarnett@refractions.net
Phone: (250) 885-0632
Fax: (250) 383-2140

TABLE OF CONTENTS

TABLE OF FIGURES

INTRODUCTION

This document describes the XML Schemas required by The Validating Web Feature Server.

This document describes:

- A XML Schema for Plug-In configuration
- A XML Schema for the definition of Test Suites

For more information on Validation Plug-Ins and Test Suites, refer to the document on the Plug-In API.

1 VALIDATING WEB FEATURE SERVER REQUIREMENTS

A Web Feature Server is allows distribution and modification of feature information over the Web using XML.

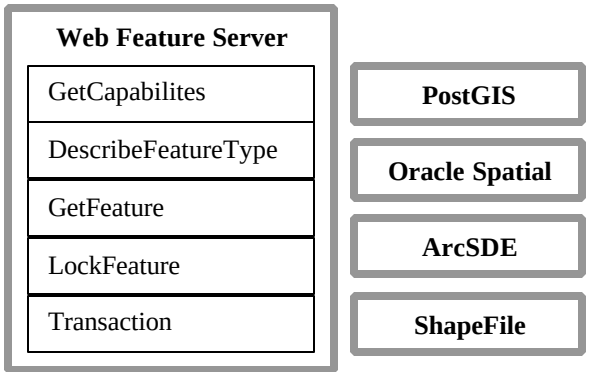

Figure 1 - Web Feature Server

The Validating Web Feature Server makes use of a Validation Processor to maintain database integrity across transaction operations.

The Validation Processor will perform a series of validation tests defined in test suites. Individual Tests will be performed Validation Plug-Ins.

Our configuration process needs to define the tests, and Plug-Ins required by the Validation Processor. Tests will be grouped into user defined Test Suites for configuration and error reporting.

Test Suite configuration files:

- Provide unique name for identification of the test suite
- Provide an optional description of the test suite
- Define tests that perform validation checks

Each Test defined by a Test Suite:

- Provides a unique name and optional description
- Provides the Plug-In used to execute the test
- Provides appropriate information to validate data

1.1 Open GIS Consortium XML Schema

Our configuration files refer to XML Schema defined by the OGC specifications:

- Geographic Markup Language (GML):
  Used to describe geometry arguments.
- Web Feature Server Implementation Specification:
  Used to describe constraints

2 TEST SUITE VALIDATION LANGUAGE

The test suite configuration file makes use of the following XML tags:

- name: provides a unique identifier
- description: provides an optional description for error messages and logging
- test: defines a single validation test

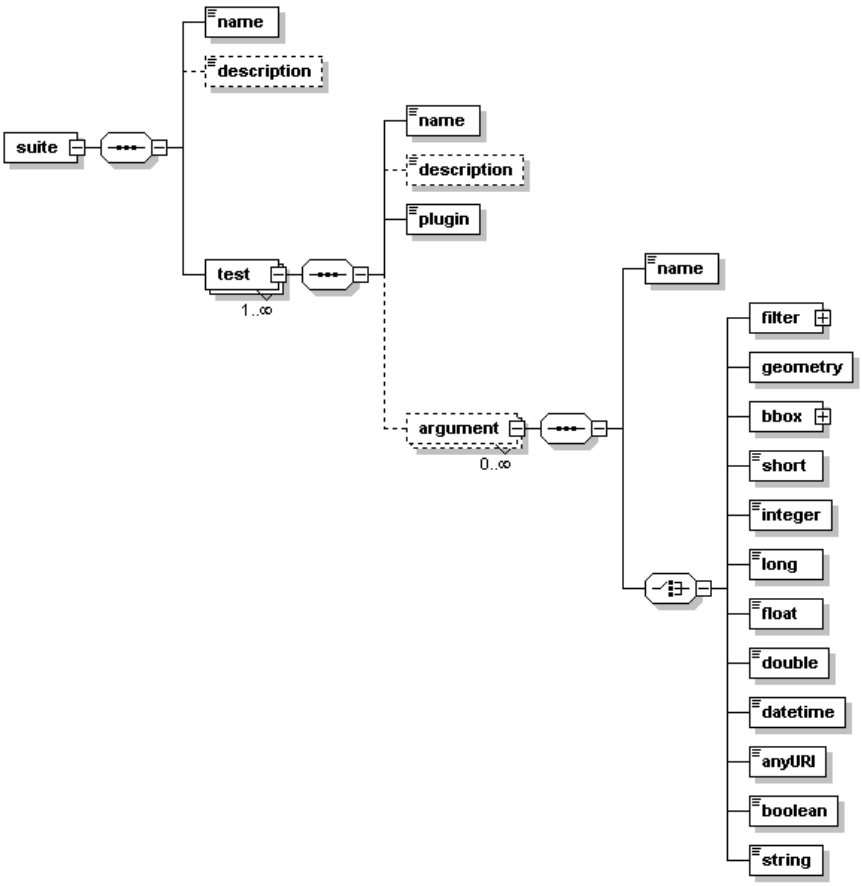

Figure 2 - Suite XML Schema

2.1.1 Validation Test

Each test requires the following tags:

- name: a unique identifier
- description: provides an optional description for error messages and logging
- plugin: Plug-In used to execute the validation test
- argument:
  Allows for additional validation test arguments required by a Plug-In

2.1.2 Test Argument

The value for argument tags can make use of:

- Filter elements to provide constraint information:

```
<filter>
  <ogc:Within>
    <ogc:PropertyName>myns:PATH</ogc:PropertyName>
     <gml:Box>
       <gml:coordinates>50,40 100,60</gml:coordinates>
     </gml:Box>
   </ogc:Within>
</filter
```

- Geometry elements to supply spatial information:

```
<geometry>
  <gml:Box>
    <gml:coordinates>50,40 100,60</gml:coordinates>
  </gml:Box>
<geometry>
```

- Bounding Box to supply envelope information:

```
<bbox><gml:coordinates>50,40 100,60</gml:coordinates></bbox>
```

- A subset of the Simple types as defined by the XML Schema Spec:

```
<short>12</short>
<integer>1</integer>
<long>1234567</long>
<float>1.3</float>
<double>3.14159</double>
<datetime>2003-02-24 14:57</datetime>
<anyURI>http://www.refractions.net</anyURI>
<boolean>TRUE</boolean>
<string>Roads</string
```

2.1.3 Test Suite Sample Configuration File

Here is a sample XML file for a test suite:

```
<?xml version="1.0" encoding="UTF-8"?>
<suite xmlns="suiteSchema"
xmlns:gml="http://www.opengis.net/gml"
xmlns:ogc="http://www.opengis.net/ogc"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="suiteSchema B:\geoinnovations\suite.xsd">
  <name>RoadTestSuite</name>
  <description>This test suite checks each road name to see
       that they are of appropriate length and checks to
       see if they are on the list of possible road names.
       It also checks to see if any roads are contained in
       a specified box.
  </description>
  <test>
    <name>NameInList</name>
    <description>Checks to see if the road name is in the list of
              possible names.</description>
    <plugin>nameLookup</plugin>
    <argument>
      <name>LUTLocation</name>
      <anyURI>
        ../www/LUTs/roadnames/
      </anyURI>
    </argument>
    <argument>
      <name>LUTName</name>
      <string>
        RoadNameLUT.xls
      </string>
    </argument>
  </test>
  <test>
    <name>RoadsInbox</name>
    <description>Checks to see if there are any roads in a given
       box</description>
    <plugin>Constraint</plugin>
    <argument>
      <name>passFilter</name>
      <filter>
        <ogc:Not>
          <ogc:Disjoint>
            <ogc:PropertyName></ogc:PropertyName>
            <gml:Box>
              <gml:coordinates>-57.9118,46.2023
                   -46.6873,51.8145</gml:coordinates>
            </gml:Box>
          </ogc:Disjoint>
        </ogc:Not>
      </filter>
    </argument>
  </test>
</suite>
```

2.2 Suite XML Schema

The following code is the schema file for the feature type validation files:

```
 1 <?xml version="1.0" encoding="UTF-8"?>
 2 <xs:schema targetNamespace="suiteSchema"
     xmlns="suiteSchema"
     xmlns:xs="http://www.w3.org/2001/XMLSchema"
     xmlns:ogc="http://www.opengis.net/ogc"
     xmlns:gml="http://www.opengis.net/gml"
     elementFormDefault="qualified"
     attributeFormDefault="unqualified">
 3   <xs:import namespace="http://www.opengis.net/gml"
        schemaLocation="geometry.xsd"/>
 5   <xs:import namespace="http://www.opengis.net/ogc"
        schemaLocation="filter.xsd"/>
 7   <xs:element name="suite">
 8     <xs:complexType>
 9       <xs:sequence>
10         <xs:element name="name" type="xs:string"/>
11         <xs:element name="description" type="xs:string" minOccurs="0"/>
12         <xs:element name="test" maxOccurs="unbounded">
13           <xs:complexType>
14             <xs:sequence>
15               <xs:element name="name" type="xs:string"/>
16               <xs:element name="description" type="xs:string"
                     minOccurs="0"/>
17               <xs:element name="plugin" type="xs:string"/>
18               <xs:element name="argument" minOccurs="0"
                     maxOccurs="unbounded">
19                 <xs:complexType>
20                   <xs:sequence>
21                     <xs:element name="name" type="xs:string"/>
22                     <xs:choice>
23                       <xs:element name="filter" type="ogc:FilterType"/>
24                       <xs:element name="geometry"
                             type="gml:AbstractGeometryType"/>
25                       <xs:element name="bbox" type="ogc:BBOXType"/>
26                       <xs:element name="short" type="xs:short"/>
27                       <xs:element name="integer" type="xs:integer"/>
28                       <xs:element name="long" type="xs:long"/>
29                       <xs:element name="float" type="xs:float"/>
30                       <xs:element name="double" type="xs:double"/>
31                       <xs:element name="datetime" type="xs:dateTime"/>
32                       <xs:element name="anyURI" type="xs:anyURI"/>
33                       <xs:element name="boolean" type="xs:boolean"/>
34                       <xs:element name="string" type="xs:string"/>
35                     </xs:choice>
36                   </xs:sequence>
37                 </xs:complexType>
38               </xs:element>
39             </xs:sequence>
40           </xs:complexType>
41         </xs:element>
42       </xs:sequence>
43     </xs:complexType>
44   </xs:element>
45 </xs:schema>
```

2.2.1 Namespace declaration

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <xs:schema targetNamespace="suiteSchema"
     xmlns="suiteSchema"
     xmlns:xs="http://www.w3.org/2001/XMLSchema"
     xmlns:ogc="http://www.opengis.net/ogc"
     xmlns:gml="http://www.opengis.net/gml"
     elementFormDefault="qualified"
     attributeFormDefault="unqualified">
3    <xs:import namespace="http://www.opengis.net/gml"
        schemaLocation="geometry.xsd"/>
5    <xs:import namespace="http://www.opengis.net/ogc"
        schemaLocation="filter.xsd"/>
```

Here is where our namespace is defined. We call our namespace “suiteSchema”. Other namespaces are brought in here as well.

2.2.2 FeatureType definition

```
7    <xs:element name="suite">
8      <xs:complexType>
9        <xs:sequence>
```

Here is where the suite schema is defined. Since it will contain many elements of various formats, it is declared a complex type. The sequence is defined so the elements that follow must appear in the order, in the XML file, that they are defined here.

2.2.3 Test Suite name

```
10         <xs:element name="name" type="xs:string"/>
```

This line is the name of the test suite.

2.2.4 Test Suite description

```
11         <xs:element name="description" type="xs:string" minOccurs="0"/>
```

Each test suite has the option of having a description. This can be useful for debugging and future reference of the validation files.

2.2.5 Test Definition

```
12          <xs:element name="test" maxOccurs="unbounded">
13             <xs:complexType>
14              <xs:sequence>
```

Each test consists of three elements, and therefore must be declared a complex type. The user can define as many checks as they please. The sequence of the elements that make up <test> begin here.

```
15                 <xs:element name="name" type="xs:string"/>
```

The first element of the test is its name. The name must be unique, as it will be used in reporting error information and in the future will be used in conditional statements.

```
16                 <xs:element name="description" type="xs:string"
                        minOccurs="0"/>
```

The test description is optional but is useful for debugging and describing the intent of the validation test. The description will be passed back as part of error reporting.

```
17                 <xs:element name="plugin" type="xs:string"/>
```

This defines the plug-in used to perform the test. The value is used to locate a valid plug-in file.

```
18                 <xs:element name="argument" minOccurs="0"
                       maxOccurs="unbounded">
19                   <xs:complexType>
20                     <xs:sequence>
```

Each test can take in a certain number of arguments, specific to each plug-in. The schema allows for any number of arguments to be passed in. If more arguments are supplied than the plug-in can accept, the first arguments will be taken.

The possible values for the argument are listed below:

```
22                       <xs:choice>
23                         <xs:element name="filter" type="ogc:FilterType"/>
24                         <xs:element name="geometry"
                               type="gml:AbstractGeometryType"/>
25                         <xs:element name="bbox" type="ogc:BBOXType"/>
26                         <xs:element name="short" type="xs:short"/>
27                         <xs:element name="integer" type="xs:integer"/>
28                         <xs:element name="long" type="xs:long"/>
29                         <xs:element name="float" type="xs:float"/>
30                         <xs:element name="double" type="xs:double"/>
31                         <xs:element name="datetime" type="xs:dateTime"/>
32                         <xs:element name="anyURI" type="xs:anyURI"/>
33                         <xs:element name="boolean" type="xs:boolean"/>
34                         <xs:element name="string" type="xs:string"/>
35                       </xs:choice>
```

2.3 Example Test

A test may ask for several arguments. The one specified below takes in one argument, an OGC filter.

```
<test>
    <name>RoadsInbox</name>
    <description>Checks to see if there are any roads in a given
       box</description>
    <plugin>Constraint</plugin>
    <argument>
      <name>passFilter</name>
      <filter>
        <ogc:Not>
          <ogc:Disjoint>
            <ogc:PropertyName></ogc:PropertyName>
            <gml:Box>
              <gml:coordinates>-57.9118,46.2023
                    -46.6873,51.8145</gml:coordinates>
            </gml:Box>
          </ogc:Disjoint>
        </ogc:Not>
      </filter>
    </argument>
</test>
```

The first part of the test is its name:

```
<name>RoadsInbox</name>
```

Second is the optional description:

```
      <description> Checks to see if there are any roads in a given
         box  </description>
```

Third is the name of the plug-in that will be used. Each plug-in requires a certain amount of arguments that are declared later.

```
            <plugin>Constraint</plugin>
```

Last is the argument. The Constraint Plug-In requires a filter argument.

```
       <filter>
        <ogc:Not>
          <ogc:Disjoint>
            <ogc:PropertyName></ogc:PropertyName>
            <gml:Box>
              <gml:coordinates>-57.9118,46.2023
                    -46.6873,51.8145</gml:coordinates>
            </gml:Box>
          </ogc:Disjoint>
        </ogc:Not>
      </filter>
```

3 PLUG-IN CONFIGURATION LANGUAGE

A Validation Plug-In is a component that performs tests on data for the Validation Processor.

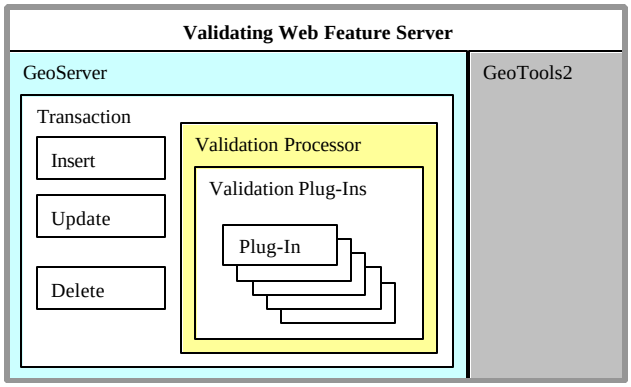

Figure 3 - Validation Plug-Ins

To configure a Plug-In the Validating Web Feature Server will need

- Identification by which Validation Tests can refer to the Plug-In
- Provide a Java Class capable of performing Validation Tests
- Configuration Information for the Java Class

This information allows us to switch which between alternate Plug-In implementations with out rewriting Test Suites.

More documentation on how a validation plug-in is used can be found in the Plug-in API design document.

3.1 Plug-In XML Schema

The Plug-In Configuration XML Schema contains the following tags:

- name: used by Validation Tests to identify Plug-In
- description: optional user supplied description
- class: full class name for the Java class implementing this Plug-In
- arguments: configuration information

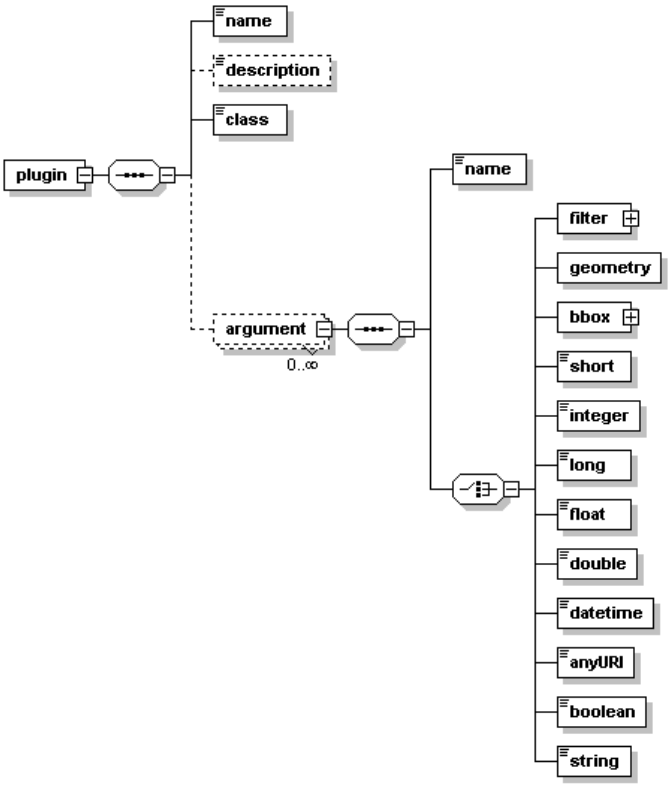

Figure 4 - Plug-In Schema

Plug-In arguments are defined in the same manner as the Test Suite Validation Language.

3.2 Sample Plug-In Configuration File

Sample XML for a validation plug-in:

```
 1 <?xml version="1.0" encoding="UTF-8"?>
 2 <plugin xmlns="pluginSchema"
 3 xmlns:gml="http://www.opengis.net/gml"
 4 xmlns:ogc="http://www.opengis.net/ogc"
 5 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 6 xsi:schemaLocation="pluginSchema C:\example\plugin.xsd">
 7   <name>Constraint</name>
 8   <description>All features must pass the provided filter</description>
 9   <class>org.geoserver.validation.plugins.filter.OGCFilter</class>
10   <argument>
11     <name>tempDirectory</name>
12     <anyURI>file:///c:/temp</anyURI>
13   </argument>
14 </plugin>
```

3.3 Plug-In XML Schema

The following code is the schema file for the plug-in validation files:

```
 1 <xs:schema targetNamespace="pluginSchema"
 2 xmlns:ogc="http://www.opengis.net/ogc"
 3 xmlns:gml="http://www.opengis.net/gml"
 4 xmlns="pluginSchema"
 5 xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
 6   <xs:import namespace="http://www.opengis.net/gml"
       schemaLocation="geometry.xsd"/>
 7   <xs:import namespace="http://www.opengis.net/ogc"
       schemaLocation="filter.xsd"/>
 8   <xs:element name="plugin">
 9     <xs:complexType>
10       <xs:sequence>
11         <xs:element name="name" type="xs:string"/>
12         <xs:element name="description" type="xs:string" minOccurs="0"/>
13         <xs:element name="class" type="xs:string"/>
14         <xs:element name="argument" minOccurs="0" maxOccurs="unbounded">
15           <xs:complexType>
16             <xs:sequence>
17               <xs:element name="name" type="xs:string"/>
18               <xs:choice>
19                 <xs:element name="filter" type="ogc:FilterType"/>
20                 <xs:element name="geometry"
                        type="gml:AbstractGeometryType"/>
21                 <xs:element name="bbox" type="ogc:BBOXType"/>
22                 <xs:element name="short" type="xs:short"/>
23                 <xs:element name="integer" type="xs:integer"/>
24                 <xs:element name="long" type="xs:long"/>
25                 <xs:element name="float" type="xs:float"/>
26                 <xs:element name="double" type="xs:double"/>
27                 <xs:element name="datetime" type="xs:dateTime"/>
28                 <xs:element name="anyURI" type="xs:anyURI"/>
29                 <xs:element name="boolean" type="xs:boolean"/>
30                 <xs:element name="string" type="xs:string"/>
31               </xs:choice>
32             </xs:sequence>
33           </xs:complexType>
34         </xs:element>
35       </xs:sequence>
36     </xs:complexType>
37   </xs:element>
38 </xs:schema>
```

3.3.1 Plug-in Definition

```
 8   <xs:element name="plugin">
 9     <xs:complexType>
10       <xs:sequence>
```

The plug-in contains several elements and is therefore declared a complex type.

3.3.2 Plug-in name

```
11         <xs:element name="name" type="xs:string"/>
```

Here is where the name of the plug-in is defined.

3.3.3 Plug-in description

```
12          <xs:element name="description" type="xs:string" minOccurs="0"/>
```

This information is strictly for debugging purposes and does not effect the outcome of the plug-in. Therefore it is optional.

3.3.4 Class name

```
13          <xs:element name="class" type="xs:string"/>
```

This tag defines the Java class that will be used to perform the test.

3.3.5 Arguments

```
14          <xs:element name="argument" minOccurs="0" maxOccurs="unbounded">
15            <xs:complexType>
16              <xs:sequence>
17                <xs:element name="name" type="xs:string"/>
```

Every plug-in can have zero or more arguments passed into it. Each argument contains a name and a value.

The possible values for the argument are listed below:

```
18                <xs:choice>
19                  <xs:element name="filter" type="ogc:FilterType"/>
20                  <xs:element name="geometry"
                         type="gml:AbstractGeometryType"/>
21                  <xs:element name="bbox" type="ogc:BBOXType"/>
22                  <xs:element name="short" type="xs:short"/>
23                  <xs:element name="integer" type="xs:integer"/>
24                  <xs:element name="long" type="xs:long"/>
25                  <xs:element name="float" type="xs:float"/>
26                  <xs:element name="double" type="xs:double"/>
27                  <xs:element name="datetime" type="xs:dateTime"/>
28                  <xs:element name="anyURI" type="xs:anyURI"/>
29                  <xs:element name="boolean" type="xs:boolean"/>
30                  <xs:element name="string" type="xs:string"/>
31                </xs:choice>
```

4 REFERENCES

O'Reilly XML.com
http://www.xml.com

Extensible Markup Language (XML) 1.0 (Second Edition)
http://www.w3.org/TR/REC-xml

XML Schema Part 1: Structures
http://www.w3.org/TR/xmlschema-1/

Apache SOAP ("Simple Object Access Protocol")
http://ws.apache.org/soap/