Network Working Group
Request for Comments: 5076
Category: Standards Track

B. Hoeneisen
SWITCH

December 2007

ENUM Validation Information Mapping for the Extensible Provisioning Protocol

Status of This Memo

Abstract

This document describes an Extensible Provisioning Protocol (EPP) extension framework for mapping information about the validation process that has been applied for the E.164 number (or number range) that the E.164 Number Mapping (ENUM) domain name is based on. Specified in the Extensible Markup Language (XML), this mapping extends the EPP domain name mapping to provide an additional feature required for the provisioning of ENUM Domain Names.

Table of Contents

1. Introduction

This document describes a framework for an ENUM [2] validation information mapping for version 1.0 of EPP [3]. This mapping, an extension of the EPP domain name mapping described in [4], is specified using XML 1.0, as described in [5], and XML Schema notation, as described in [6] and [7].

The EPP core protocol specification [3] provides a complete description of EPP command and response structures. A thorough understanding of the base protocol specification is necessary to understand the mapping described in this document.

ENUM [2] describes how the Domain Name System (DNS) can be used to identify services associated with an E.164 number.

As described in RFC 4725 [9], usually only the Assignee of the E.164 number (or number range) has the right to register the corresponding ENUM domain name. Therefore, an ENUM validation process has to be applied before the ENUM domain name can be inserted into the DNS. The validation process shall ensure that the holder of the ENUM domain name coincides with the Assignee of the corresponding E.164 number (or number range). However, the details of the ENUM validation methods are beyond the scope of this document.

The EPP extension described in this document specifies a framework for the mapping of information about the ENUM validation process. As the local legislation or the validation procedures may vary, the content of the validation information itself is not part of this specification.

However, this document contains a working example (including XML schema) to show how the validation information could look. This example could even be used for a lightweight validation process. In fact, it has been an integral part of the Swiss ENUM trial.

Using this extension framework, the content of the validation information can be specified according to the local requirements. Such an extension is specified in [10].

More background information concerning the validation can be found in RFC 4725 [9], which also describes a typical basic role model for the ENUM registration process.

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [1].

In examples, "C:" represents lines sent by a protocol client and "S:" represents lines returned by a protocol server. Indentation and white space in examples are provided only to illustrate element relationships and are not REQUIRED features of this specification.

XML is case sensitive. Unless stated otherwise, XML specifications and examples provided in this document MUST be interpreted in the character case presented to develop a conforming implementation.

3. Requirements

The following requirements are the basis for this work:

1. The design shall allow multiple policies and validation procedures.

2. It shall be possible to transmit validation information with EPP domain object requests and responses.

3. It shall be possible to add, modify, and remove validation information.

4. It shall be possible to retrieve validation information stored in the ENUM Registry.

4. Object Attributes

This extension adds additional elements to the EPP domain name mapping [4]. Only new element descriptions are listed here.

4.1. ENUM Domain Names

An ENUM Domain Name is a representation of an E.164 number that has been translated to conform to domain name syntax as described in the ENUM specification [2].

4.2. Validation Information Commands

The following commands are defined for handling validation information at the registry:

- add: to add new validation information
- rem: to revoke validation information
- chg: to change stored validation information
- inf: to get information about stored validation information

4.3. Id

The "id" attribute, used to identify the validation, is represented in this mapping using a character string. It MUST be unique at least within the same ENUM Domain Name. To ensure uniqueness even after a transfer of an ENUM Domain Name, it is RECOMMENDED that the "id" attribute be unique per ENUM Registry.

The "id" attribute, usually assigned by the ENUM Registrar, is required for revoking or changing stored validation information and appears in the Validation Information Command elements (see Section 4.2).

4.4. Validation Information

The <validationInfo> element can contain any element containing validation information that is documented adequately. It is represented in this mapping using the XML schema <any> element and therefore, is extensible.

The number of <validationInfo> elements permitted per domain object is subject to local policy.

4.5. Validation Elements in the Example

As described above, this document includes an example for a possible content of validation information that is used in the EPP examples throughout this document.

This example is an optional part of this specification, i.e., a fully compliant RFC 5076 implementation does not need to implement this example.

4.5.1. Method Identifier

The <methodID> element is represented in this mapping using a character string with a maximum length of 63 characters. It contains an identifier for the method used for the validation. As stated in Section 1, the details of the ENUM validation methods are beyond the scope of this document.

4.5.2. Validation Entity Identifier

The <validationEntityID> element is represented in this mapping using a character string with a length of 3 to 16 characters. It contains an identifier assigned to the ENUM Validation Entity, e.g., by the ENUM Registry.

4.5.3. Registrar Identifier

The <registrarID> element is represented in this mapping using a character string with a length of 3 to 16 characters. It contains an identifier assigned to the ENUM Registrar by the ENUM Registry.

4.5.4. Execution Date

The <executionDate> element, the execution date of the validation, is represented in this mapping using the XML Schema 'date' data type.

4.5.5. Expiration Date

The <expirationDate> element, the expiration date of the validation, is represented in this mapping using the XML Schema 'date' data type.

5. EPP Command Mapping

A detailed description of the EPP syntax and semantics can be found in the EPP core protocol specification [3], and the EPP domain name mapping is described in [4]. The command mappings described here are specifically for use in implementing ENUM validation information provisioning processes via EPP.

Note: Whether or not this extension is included into an EPP request or response depends on local policy. For example, a local Registry policy might require the use of this extension for EPP <create>, <update>, and <info> commands, but not support it for EPP <transfer> and <renew> commands. Therefore, this is beyond the scope of this document.

5.1. EPP Query Commands

EPP provides three commands to retrieve object information: <check> to determine if an object is known to the server, <info> to retrieve detailed information associated with an object, and <transfer> to retrieve object transfer status information.

5.1.1. EPP <check> Command

This extension does not add any elements to the EPP <check> command or <check> response described in the EPP domain mapping [4].

5.1.2. EPP <info> Command

This extension does not add any elements to the EPP <info> command described in the EPP domain mapping [4]. Additional elements are defined for the <info> response.

When an <info> command has been processed successfully, the EPP <resData> element MUST contain child elements as described in the EPP domain mapping [4]. In addition, the EPP <extension> element MUST contain an <e164val:infData> element that identifies the extension namespace. The <e164val:infData> element contains one or more

<e164val:inf> elements, each with an "id" attribute identifying the validation. Each <e164val:inf> element contains an <e164val: validationInfo> element, which contains the validation information as child element.

In the example below, the validation information consists of a <valex:simpleVal> element that identifies the extension namespace. The <valex:simpleVal> element contains the following child elements:

- An <e164val:methodID> element that contains an identifier of the validation method.
- An OPTIONAL <e164val:validationEntityID> element that contains an identifier assigned to the ENUM Validation Entity.
- An OPTIONAL <e164val:registrarID> element that contains an identifier assigned to the ENUM Registrar by the ENUM Registry.
- An <e164val:executionDate> element that contains the date that the validation was performed.
- An OPTIONAL <e164val:expirationDate> element that contains the date that the validation expires.

Example for <info> response:

```
S:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
S:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0"
S:     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
S: <response>
S:  <result code="1000">
S:   <msg>Command completed successfully</msg>
S:  </result>
S:  <resData>
S:   <domain:infData
S:    xmlns:domain="urn:ietf:params:xml:ns:domain-1.0">
S:    <domain:name>5.1.5.1.8.6.2.4.4.1.4.e164.arpa</domain:name>
S:    <domain:roid>EXAMPLE1-REP</domain:roid>
S:    <domain:status s="ok"/>
S:    <domain:registrant>jd1234</domain:registrant>
S:    <domain:contact type="admin">sh8013</domain:contact>
S:    <domain:contact type="tech">sh8013</domain:contact>
S:    <domain:ns>
S:     <domain:hostObj>ns1.example.com</domain:hostObj>
S:     <domain:hostObj>ns2.example.com</domain:hostObj>
S:    </domain:ns>
S:    <domain:clID>ClientX</domain:clID>
S:    <domain:crID>ClientY</domain:crID>
```

```
S:    <domain:crDate>1999-04-03T22:00:00.0Z</domain:crDate>
S:    <domain:upID>ClientX</domain:upID>
S:    <domain:upDate>1999-12-03T09:00:00.0Z</domain:upDate>
S:    <domain:exDate>2005-04-03T22:00:00.0Z</domain:exDate>
S:    <domain:trDate>2000-04-08T09:00:00.0Z</domain:trDate>
S:    <domain:authInfo>
S:     <domain:pw>2fooBAR</domain:pw>
S:    </domain:authInfo>
S:   </domain:infData>
S:  </resData>
S:  <extension>
S:   <e164val:infData
S:    xmlns:e164val="urn:ietf:params:xml:ns:e164val-1.0">
S:    <e164val:inf id="EK77">
S:     <e164val:validationInfo>
S:      <valex:simpleVal
S:       xmlns:valex="urn:ietf:params:xml:ns:e164valex-1.1">
S:       <valex:methodID>Validation-X</valex:methodID>
S:       <valex:validationEntityID>VE-NMQ</valex:validationEntityID>
S:       <valex:registrarID>Client-X</valex:registrarID>
S:       <valex:executionDate>2004-04-08</valex:executionDate>
S:       <valex:expirationDate>2004-10-07</valex:expirationDate>
S:      </valex:simpleVal>
S:     </e164val:validationInfo>
S:    </e164val:inf>
S:   </e164val:infData>
S:  </extension>
S:  <trID>
S:   <clTRID>ABC-23456</clTRID>
S:   <svTRID>54321-XYZ</svTRID>
S:  </trID>
S: </response>
S:</epp>
```

Figure 1

5.1.3. EPP <transfer> Command

This extension does not add any elements to the EPP <transfer> command or <transfer> response described in the EPP domain mapping [4].

5.2. EPP Transform Commands

EPP provides five commands to transform objects: <create> to create an instance of an object, <delete> to delete an instance of an object, <renew> to extend the validity period of an object, <transfer> to manage object sponsorship changes, and <update> to change information associated with an object.

5.2.1. EPP <create> Command

This extension defines additional elements for the EPP <create> command described in the EPP domain mapping [4]. No additional elements are defined for the EPP <create> response.

The EPP <create> command provides a transform operation that allows a client to create a domain object. In addition to the EPP command elements described in the EPP domain mapping [4], the command MUST contain an <extension> element. The <extension> element MUST contain an <e164val:create> element that identifies the extension namespace. The <e164val:create> element contains one or more <e164val:add> elements, each with an "id" attribute identifying the validation. Each <e164val:add> element contains an <e164val:validationInfo> element, which contains the validation information as child element.

In the example below, the validation information consists of a <valex:simpleVal> element that identifies the extension namespace. The <valex:simpleVal> element contains the following child elements:

- An <e164val:methodID> element that contains an identifier of the validation method.
- An OPTIONAL <e164val:validationEntityID> element that contains an identifier assigned to the ENUM Validation Entity.
- An OPTIONAL <e164val:registrarID> element that contains an identifier assigned to the ENUM Registrar by the ENUM Registry.
- An <e164val:executionDate> element that contains the date that the validation was performed.
- An OPTIONAL <e164val:expirationDate> element that contains the date that the validation expires.

Example for <create> command:

```
C:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
C:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0"
C:     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
C: <command>
C:  <create>
C:   <domain:create
C:    xmlns:domain="urn:ietf:params:xml:ns:domain-1.0">
C:     <domain:name>5.1.5.1.8.6.2.4.4.1.4.e164.arpa</domain:name>
C:     <domain:period unit="y">1</domain:period>
C:     <domain:ns>
C:      <domain:hostObj>ns1.example.com</domain:hostObj>
C:      <domain:hostObj>ns2.example.com</domain:hostObj>
C:     </domain:ns>
C:     <domain:registrant>jd1234</domain:registrant>
C:     <domain:contact type="admin">sh8013</domain:contact>
C:     <domain:contact type="tech">sh8013</domain:contact>
C:     <domain:authInfo>
C:      <domain:pw>2fooBAR</domain:pw>
C:     </domain:authInfo>
C:   </domain:create>
C:  </create>
C:  <extension>
C:   <e164val:create
C:    xmlns:e164val="urn:ietf:params:xml:ns:e164val-1.0">
C:    <e164val:add id="EK77">
C:     <e164val:validationInfo>
C:      <valex:simpleVal
C:       xmlns:valex="urn:ietf:params:xml:ns:e164valex-1.1">
C:       <valex:methodID>Validation-X</valex:methodID>
C:       <valex:validationEntityID>VE-NMQ</valex:validationEntityID>
C:       <valex:registrarID>Client-X</valex:registrarID>
C:       <valex:executionDate>2004-04-08</valex:executionDate>
C:       <valex:expirationDate>2004-10-07</valex:expirationDate>
C:      </valex:simpleVal>
C:     </e164val:validationInfo>
C:    </e164val:add>
C:   </e164val:create>
C:  </extension>
C:  <clTRID>ABC-12345</clTRID>
C: </command>
C:</epp>
```

Figure 2

When an extended <create> command has been processed successfully, the EPP response is as described in the EPP domain mapping [4].

5.2.2. EPP <delete> Command

This extension does not add any elements to the EPP <delete> command or <delete> response described in the EPP domain mapping [4].

5.2.3. EPP <renew> Command

This extension defines additional elements for the EPP <renew> command described in the EPP domain mapping [4]. No additional elements are defined for the EPP <renew> response.

The EPP <renew> command provides a transform operation that allows a client to extend the validity period of a domain object. In addition to the EPP command elements described in the EPP domain mapping [4], the <renew> command MUST contain an <extension> element. The <extension> element MUST contain an <e164val:renew> element that identifies the extension namespace. The <e164val:renew> element contains one or more <e164val:add> elements, each with an "id" attribute identifying the validation. Each <e164val:add> element contains an <e164val:validationInfo> element, which contains the validation information as child element.

In the example below, the validation information consists of a <valex:simpleVal> element that identifies the extension namespace. The <valex:simpleVal> contains the following child elements:

- An <e164val:methodID> element that contains an identifier of the validation method.
- An OPTIONAL <e164val:validationEntityID> element that contains an identifier assigned to the ENUM Validation Entity.
- An OPTIONAL <e164val:registrarID> element that contains an identifier assigned to the ENUM Registrar by the ENUM Registry.
- An <e164val:executionDate> element that contains the date that the validation was performed.
- An OPTIONAL <e164val:expirationDate> element that contains the date that the validation expires.

Example for <renew> command:

```
C:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
C:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0"
C:     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
C: <command>
C:  <renew>
C:   <domain:renew
C:    xmlns:domain="urn:ietf:params:xml:ns:domain-1.0">
C:    <domain:name>5.1.5.1.8.6.2.4.4.1.4.e164.arpa</domain:name>
C:    <domain:curExpDate>2005-04-09</domain:curExpDate>
C:    <domain:period unit="y">1</domain:period>
C:   </domain:renew>
C:  </renew>
C:  <extension>
C:   <e164val:renew
C:    xmlns:e164val="urn:ietf:params:xml:ns:e164val-1.0">
C:    <e164val:add id="CAB176">
C:     <e164val:validationInfo>
C:      <valex:simpleVal
C:       xmlns:valex="urn:ietf:params:xml:ns:e164valex-1.1">
C:       <valex:methodID>Validation-X</valex:methodID>
C:       <valex:validationEntityID>VE-NMQ</valex:validationEntityID>
C:       <valex:registrarID>Client-X</valex:registrarID>
C:       <valex:executionDate>2005-03-30</valex:executionDate>
C:       <valex:expirationDate>2005-09-29</valex:expirationDate>
C:      </valex:simpleVal>
C:     </e164val:validationInfo>
C:    </e164val:add>
C:   </e164val:renew>
C:  </extension>
C:  <clTRID>ABC-45678</clTRID>
C: </command>
C:</epp>
```

Figure 3

When an extended <renew> command has been processed successfully, the EPP response is as described in the EPP domain mapping [4].

5.2.4. EPP <transfer> Command

This extension defines additional elements for the EPP <transfer> command described in the EPP domain mapping [4]. No additional elements are defined for the EPP <transfer> response.

The EPP <transfer> command provides a transform operation that allows a client to manage requests to transfer the sponsorship of a domain object. Clients can initiate, cancel, approve, and reject a transfer request.

In case of a transfer request, in addition to the EPP command elements described in the EPP domain mapping [4], the command MUST contain an <extension> element. The <extension> element MUST contain an <e164val:transfer> element that identifies the extension namespace. The <e164val:transfer> element contains one or more <e164val:add> elements, each with an "id" attribute identifying the validation. Each <e164val:add> element contains an <e164val: validationInfo> element, which contains the validation information as child element.

In the example below, the validation information consists of a <valex:simpleVal> element that identifies the extension namespace. The <valex:simpleVal> contains the following child elements:

- An <e164val:methodID> element that contains an identifier of the validation method.
- An OPTIONAL <e164val:validationEntityID> element that contains an identifier assigned to the ENUM Validation Entity.
- An OPTIONAL <e164val:registrarID> element that contains an identifier assigned to the ENUM Registrar by the ENUM Registry.
- An <e164val:executionDate> element that contains the date that the validation was performed.
- An OPTIONAL <e164val:expirationDate> element that contains the date that the validation expires.

Example for <transfer> command:

```
C:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
C:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0"
C:     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
C: <command>
C:  <transfer op="request">
C:   <domain:transfer
C:    xmlns:domain="urn:ietf:params:xml:ns:domain-1.0">
C:    <domain:name>5.1.5.1.8.6.2.4.4.1.4.e164.arpa</domain:name>
C:    <domain:authInfo>
C:     <domain:pw roid="HB1973-ZUE">2fooBAR</domain:pw>
C:    </domain:authInfo>
C:   </domain:transfer>
C:  </transfer>
C:  <extension>
C:   <e164val:transfer
C:    xmlns:e164val="urn:ietf:params:xml:ns:e164val-1.0">
C:    <e164val:add id="LJ1126">
C:     <e164val:validationInfo>
C:      <valex:simpleVal
C:       xmlns:valex="urn:ietf:params:xml:ns:e164valex-1.1">
C:       <valex:methodID>Validation-Y</valex:methodID>
C:       <valex:validationEntityID>VE2-LMQ</valex:validationEntityID>
C:       <valex:registrarID>Client-Y</valex:registrarID>
C:       <valex:executionDate>2005-01-22</valex:executionDate>
C:       <valex:expirationDate>2005-07-21</valex:expirationDate>
C:      </valex:simpleVal>
C:     </e164val:validationInfo>
C:    </e164val:add>
C:   </e164val:transfer>
C:  </extension>
C:  <clTRID>XYZ-54789</clTRID>
C: </command>
C:</epp>
```

Figure 4

When an extended <transfer> command has been processed successfully, the EPP response is as described in the EPP domain mapping [4].

5.2.5. EPP <update> Command

This extension defines additional elements for the EPP <update> command described in the EPP domain mapping [4]. No additional elements are defined for the EPP <update> response. The EPP <update> command provides a transform operation that allows a client to change the state of a domain object. In addition to the EPP command elements described in the EPP domain mapping [4], the <update> command MUST contain an <extension> element. The <extension> element MUST contain an <e164val:update> element that identifies the extension namespace. The <e164val:update> element contains one or more <e164val:add>, <e164val:rem>, or <e164val:chg> elements, each with an "id" attribute identifying the validation. Each <e164val:add> and <e164val:chg> element contains an <e164val: validationInfo> element, which contains the validation information as child element. <e164val:rem> elements do not have child elements.

In the example below, the validation information consists of a <valex:simpleVal> element that identifies the extension namespace. The <valex:simpleVal> contains the following child elements:

- An <e164val:methodID> element that contains an identifier of the validation method.
- An OPTIONAL <e164val:validationEntityID> element that contains an identifier assigned to the ENUM Validation Entity.
- An OPTIONAL <e164val:registrarID> element that contains an identifier assigned to the ENUM Registrar by the ENUM Registry.
- An <e164val:executionDate> element that contains the date that the validation was performed.
- An OPTIONAL <e164val:expirationDate> element that contains the date that the validation expires.

Example for <update> command:

```
C:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
C:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0"
C:     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
C: <command>
C:  <update>
C:   <domain:update
C:    xmlns:domain="urn:ietf:params:xml:ns:domain-1.0">
C:    <domain:name>5.1.5.1.8.6.2.4.4.1.4.e164.arpa</domain:name>
C:   </domain:update>
C:  </update>
C:  <extension>
C:   <e164val:update
C:    xmlns:e164val="urn:ietf:params:xml:ns:e164val-1.0">
C:    <e164val:add id="EK2510">
C:     <e164val:validationInfo>
C:      <valex:simpleVal
C:       xmlns:valex="urn:ietf:params:xml:ns:e164valex-1.1">
C:       <valex:methodID>Validation-X</valex:methodID>
C:       <valex:validationEntityID>VE-NMQ</valex:validationEntityID>
C:       <valex:registrarID>Client-X</valex:registrarID>
C:       <valex:executionDate>2004-10-02</valex:executionDate>
C:       <valex:expirationDate>2005-04-01</valex:expirationDate>
C:      </valex:simpleVal>
C:     </e164val:validationInfo>
C:    </e164val:add>
C:    <e164val:rem id="EK77"/>
C:   </e164val:update>
C:  </extension>
C:  <clTRID>ABC-34567</clTRID>
C: </command>
C:</epp>
```

Figure 5

When an extended <update> command has been processed successfully, the EPP response is as described in the EPP domain mapping [4].

6. Formal Syntax

An EPP object mapping is specified in XML Schema notation. The formal syntax presented here is a complete schema representation of the object mapping suitable for automated validation of EPP XML instances. The BEGIN and END tags are not part of the schemas; they are used to note the beginning and ending of the schema for URI registration purposes.

Formal syntax for Framework:

```
BEGIN
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="urn:ietf:params:xml:ns:e164val-1.0"
        xmlns:e164val="urn:ietf:params:xml:ns:e164val-1.0"
        xmlns:eppcom="urn:ietf:params:xml:ns:eppcom-1.0"
        xmlns="http://www.w3.org/2001/XMLSchema"
        elementFormDefault="qualified">

<!--
Import common element types.
-->
  <import namespace="urn:ietf:params:xml:ns:eppcom-1.0"
          schemaLocation="eppcom-1.0.xsd"/>

  <annotation>
    <documentation>
      Extensible Provisioning Protocol v1.0
      domain name extension schema for framework for
      provisioning of E.164 number validation information.
    </documentation>
  </annotation>

<!--
Child elements found in EPP commands.
-->
  <element name="create" type="e164val:insertType"/>
  <element name="update" type="e164val:updateType"/>
  <element name="renew" type="e164val:insertType"/>
  <element name="transfer" type="e164val:insertType"/>

<!--
Child elements of the <create>, <renew>, and <update> commands.
-->
  <complexType name="insertType">
    <sequence>
      <element name="add" type="e164val:addType"
       maxOccurs="unbounded" />
    </sequence>
  </complexType>

<!--
Child elements of the <update> command.
-->
  <complexType name="updateType">
    <sequence>
      <element name="add" type="e164val:addType"
```

```
          minOccurs="0"
          maxOccurs="unbounded"/>
        <element name="rem" type="e164val:remType"
          minOccurs="0"
          maxOccurs="unbounded"/>
        <element name="chg" type="e164val:chgType"
          minOccurs="0"
          maxOccurs="unbounded"/>
      </sequence>
    </complexType>

  <!--
  Data elements for add, chg and rem.
  -->
    <complexType name="addType">
      <sequence>
        <element ref="e164val:validationInfo"/>
      </sequence>
      <attribute name="id" type="eppcom:minTokenType"
       use="required"/>
    </complexType>

    <complexType name="chgType">
      <sequence>
        <element ref="e164val:validationInfo"/>
      </sequence>
      <attribute name="id" type="eppcom:minTokenType"
       use="required"/>
    </complexType>

    <complexType name="remType">
      <attribute name="id" type="eppcom:minTokenType"
       use="required"/>
    </complexType>


  <!--
  Child elements found in EPP responses
  -->
    <element name="infData" type="e164val:infDataType"/>

  <!--
  child elements of the <info> response.
  -->
    <complexType name="infDataType">
      <sequence>
        <element name="inf" type="e164val:infType"
          minOccurs="0"
```

```
          maxOccurs="unbounded"/>
      </sequence>
    </complexType>

  <!--
  Data elements for inf
  -->
    <complexType name="infType">
      <sequence>
        <element ref="e164val:validationInfo"/>
      </sequence>
      <attribute name="id" type="eppcom:minTokenType"
       use="required"/>
    </complexType>

  <!--
  Global elements.
  -->
    <element name="validationInfo" type="e164val:ValidationInfoType" />

  <!--
  Extension framework types.
  -->
    <complexType name="ValidationInfoType">
      <sequence>
        <any namespace="##other"/>
      </sequence>
    </complexType>


  <!--
  End of schema.
  -->
  </schema>
  END
```

Figure 6

Formal syntax for a simple validation (example):

```
BEGIN
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="urn:ietf:params:xml:ns:e164valex-1.1"
        xmlns:e164valex="urn:ietf:params:xml:ns:e164valex-1.1"
        xmlns:eppcom="urn:ietf:params:xml:ns:eppcom-1.0"
        xmlns="http://www.w3.org/2001/XMLSchema"
        elementFormDefault="qualified">
```

```
<!--
Import common element types.
-->
  <import namespace="urn:ietf:params:xml:ns:eppcom-1.0"
          schemaLocation="eppcom-1.0.xsd"/>

  <annotation>
    <documentation>
      Example for E.164 number validation information.
    </documentation>
  </annotation>


  <element name="simpleVal" type="e164valex:simpleValType"/>

  <complexType name="simpleValType">
    <sequence>
      <element name="methodID" type="e164valex:methodIdType"/>
      <element name="validationEntityID" type="eppcom:clIDType"
       minOccurs="0"/>
      <element name="registrarID" type="eppcom:clIDType"
       minOccurs="0"/>
      <element name="executionDate" type="date"/>
      <element name="expirationDate" type="date"
       minOccurs="0"/>
    </sequence>
  </complexType>

  <simpleType name="methodIdType">
    <restriction base="token">
      <minLength value="1"/>
      <maxLength value="63"/>
    </restriction>
  </simpleType>

<!--
End of schema.
-->
</schema>
END
```

Figure 7

7. IANA Considerations

This document uses Uniform Resource Names (URNs) to describe XML namespaces and XML schemas conforming to the registry mechanism described in RFC 3688 [8]. Four URI assignments have been made:

1. Registration for the extension namespace:
   * URI: urn:ietf:params:xml:ns:e164val-1.0
   * Registrant Contact: See the "Author's Address" section of this document.
   * XML: None. Namespace URIs do not represent an XML specification.

2. Registration for the extension XML schema:
   * URI: urn:ietf:params:xml:schema:e164val-1.0
   * Registrant Contact: See the "Author's Address" section of this document.
   * XML: See Section 6, "Formal Syntax", of this document.

3. Registration for the extension namespace:
   * URI: urn:ietf:params:xml:ns:e164valex-1.1
   * Registrant Contact: See the "Author's Address" section of this document.
   * XML: None. Namespace URIs do not represent an XML specification.

4. Registration for the extension XML schema:
   * URI: urn:ietf:params:xml:schema:e164valex-1.1
   * Registrant Contact: See the "Author's Address" section of this document.
   * XML: See Section 6, "Formal Syntax", of this document.

8. Security Considerations

The mapping extensions described in this document do not provide any security services beyond those described by EPP [3], the EPP domain name mapping [4], and protocol layers used by EPP. Security considerations related to ENUM are described in the "Security Considerations" section of the ENUM specification [2]. The security considerations described in these other specifications apply to this specification as well.

Validation information often contains sensitive personal information. It is RECOMMENDED that validation information in the <info> response is only provided to the sponsoring client.

9. Acknowledgements

The author would like to thank the following people who have provided feedback or significant contributions to the development of this document: Alfred Hoenes, Helena Malmborg, Alexander Mayrhofer, Andrew Newton, Marcel Parodi, Patrik Schaefer, and Patrick Zenklusen.

RFC 4114 [11] has been used as a template for this document. The structure and those paragraphs that apply to both documents have been taken over from [11]. The author would like to thank Scott Hollenbeck for this great spadework.

10. References

10.1. Normative References

[1] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[2] Faltstrom, P. and M. Mealling, "The E.164 to Uniform Resource Identifiers (URI) Dynamic Delegation Discovery System (DDDS) Application (ENUM)", RFC 3761, April 2004.

[3] Hollenbeck, S., "Extensible Provisioning Protocol (EPP)", RFC 3730, March 2004.

[4] Hollenbeck, S., "Extensible Provisioning Protocol (EPP) Domain Name Mapping", RFC 3731, March 2004.

[5] Paoli, J., Maler, E., Bray, T., and C. Sperberg-McQueen, "Extensible Markup Language (XML) 1.0 (Second Edition)", World Wide Web Consortium FirstEdition REC-xml-20001006, October 2000, <http://www.w3.org/TR/2000/REC-xml-20001006>.

[6] Thompson, H., Maloney, M., Mendelsohn, N., and D. Beech, "XML Schema Part 1: Structures Second Edition", World Wide Web Consortium Recommendation REC-xmlschema-1-20041028, October 2004, <http://www.w3.org/TR/2004/REC-xmlschema-1-20041028>.

[7] Biron, P. and A. Malhotra, "XML Schema Part 2: Datatypes Second Edition", World Wide Web Consortium Recommendation REC-xmlschema-2-20041028, October 2004, <http://www.w3.org/TR/2004/REC-xmlschema-2-20041028>.

[8] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, January 2004.

10.2. Informative References

[9] Mayrhofer, A. and B. Hoeneisen, "ENUM Validation Architecture", RFC 4725, November 2006.

[10] Lendl, O., "ENUM Validation Token Format Definition", Work in Progress.

[11] Hollenbeck, S., "E.164 Number Mapping for the Extensible Provisioning Protocol (EPP)", RFC 4114, June 2005.

Author's Address

Bernie Hoeneisen
SWITCH
Werdstrasse 2
CH-8004 Zuerich
Switzerland

Phone: +41 44 268 1515
EMail: bernhard.hoeneisen@switch.ch, bernie@ietf.hoeneisen.ch
URI: http://www.switch.ch/

Full Copyright Statement

Intellectual Property