Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: August 28, 2008

B. Natale, Ed.
MITRE
Y. Li, Ed.
Huawei Technologies

February 25, 2008

Expressing SNMP SMI Datatypes in XML Schema Definition Language
draft-ietf-opsawg-smi-datatypes-in-xsd-01.txt

Status of This Memo

Copyright Notice

Abstract

This memo (when approved as a standards-track RFC) defines the IETF standard expression of Structure of Management Information (SMI) base datatypes in Extensible Markup Language (XML) Schema Definition (XSD) language. The primary objective of this memo is to enable production of XML documents that are as faithful to the SMI as possible, using XSD as the validation mechanism.

Table of Contents

1. Introduction

Numerous uses exist -- both within and outside the traditional IETF network management community -- for the expression of management information described in and accessible via SMI Management Information Base (MIB) modules as XML documents [ref.XML]. For example, XML-based management applications which want to incorporate MIB modules as data models and/or to access MIB module instrumentation via gateways to SNMP agents will benefit from an IETF standard mapping of SMI datatypes and structures to XML documents via XSD.

MIB data models are described using SMIv2 [RFC2578] and, for legacy MIBs, SMIv1 [RFC1155]. MIB data is conveyed via SNMP using the base datatypes defined in the SMI. The SMI allows for creation of derivative datatypes, termed "textual conventions" ("TCs"), each of which has a unique name, a syntax based on a core SMI datatype, and relatively precise application-level semantics. TCs are used principally to facilitate correct application-level handling of MIB data and for the convenience of humans reading MIB modules and appropriately rendered MIB data output.

Various independent schemes have been devised for expressing the SMI datatypes and TCs in XSD [ref.XMLSchema]. These schemes have exhibited a degree of commonality (especially concerning the numeric SMI datatypes), but also sufficient differences (especially concerning the non-numeric SMI datatypes) to preclude general interoperability.

The primary purpose of this memo is to define a standard expression of SMI base datatypes in XSD to ensure uniformity and general interoperability in this respect. Internet operators, management tool developers, and users will benefit from the wider selection of management tools and the greater degree of unified management -- with attendant improvements in timeliness and accuracy of management information -- which such a standard will facilitate.

This memo is the first in a set of three related and (logically) ordered specifications:

1. SMI Base Datatypes [RFC2578] in XSD
2. SMI MIB Structure [RFC2578] in XSD
3. SNMP Textual Conventions [RFC2579] in XSD

As a set, these documents define the XSD equivalent of SMIv2 to encourage XML-based protocols to carry, and XML-based applications to use, the information modeled in SMIv2-compliant MIB modules.

This work defines XSD equivalents of the datatypes and data structures [RFC2578] and the textual conventions [RFC2579] defined in the SMIv2 standard (STD58) to encourage efficient reuse of existing (including future) MIB modules and instrumentation by XML-based management protocols and applications.

The goal of fidelity to the SMIv2 standard (STD58), as specified in the "Requirements" section below, is crucial to this effort to leverage the established "rough consensus" for the precise data modeling used in MIB modules, and to leverage existing "running code" for implemented SMIv2 data models. This effort does not include redesign of SMIv2 datatypes or data structures or textual conventions to overcome known limitations -- that work can be pursued in other efforts.

2. Conventions

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 [RFC2119].

Sections requiring further editing are identified by [TODO] markers in the text. Points requiring further WG research and discussion are identified by [DISCUSS] markers in the text.

3. Requirements

R1. All SMI datatypes MUST have a corresponding XSD datatype.

R2. SMIv2 is the normative SMI for this document -- SMIv1 modules, if encountered, MUST be converted (at least logically) in accordance with Section 2.1, inclusive, of the "Coexistence" RFC [RFC3584].

R3. The XSD datatype specified for a given SMI datatype MUST be able to represent all valid values for that SMI datatype.

R4. The XSD datatype specified for a given SMI datatype MUST represent any special encoding rules associated with that SMI datatype.

R5. The XSD datatype specified for a given SMI datatype MUST include any restrictions on values associated with the SMI datatype.

R6. The XSD datatype specified for a given SMI datatype MUST be the most direct XSD datatype, with the most parsimonious restrictions, which matches the foregoing requirements.

R7. The XML output produced as a result of meeting the foregoing requirements SHOULD be the most direct (i.e., avoiding superfluous "decoration") from the perspective of readability by humans.

[DISCUSS} Should any requirements be added, deleted, re-worded?

4. XSD for SMI Datatypes

This document concerns the SMI base datatypes. These are carried "on-the-wire" (and, therefore, have tag values defined in the SMI to identify them in varbinds) in SNMP PDUs between SNMP management applications and SNMP agents:

- INTEGER, Integer32
- Unsigned32, Gauge32
- Counter32
- TimeTicks
- Counter64
- OctetString
- Opague
- IpAddress
- ObjectIdentifier

The following should be considered a "notional" XSD file for now, pending agreement on the actual datatype specifications. An appropriate (official) targetNamespace must be designated (and approved) and agreement must be reached on whether any additional XSD content must be included (e.g., whether non-default values for elementFormDefault or attributeFormDefault or schemaLocation, etc., need to be specified).

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <xs:annotation>
    <xs:documentation>
        Mapping of SMIv2 datatypes from RFC 2578.
    </xs:documentation>
  </xs:annotation>

  <xs:simpleType name="INTEGER">
    <xs:restriction base="xs:int"/>
  </xs:simpleType>

  <xs:simpleType name="Integer32">
    <xs:restriction base="xs:int"/>
  </xs:simpleType>

  <xs:simpleType name="Unsigned32">
    <xs:restriction base="xs:unsignedInt"/>
  </xs:simpleType>

  <xs:simpleType name="Gauge32">
```

```
      <xs:restriction base="xs:unsignedInt"/>
    </xs:simpleType>

    <xs:simpleType name="Counter32">
      <xs:restriction base="xs:unsignedInt"/>
    </xs:simpleType>

    <xs:simpleType name="TimeTicks">
      <xs:restriction base="xs:unsignedInt"/>
    </xs:simpleType>

    <xs:simpleType name="Counter64">
      <xs:restriction base="xs:unsignedLong"/>
    </xs:simpleType>

    <xs:simpleType name="OctetString">
      <xs:restriction base="xs:hexBinary">
        <xs:maxLength value="65535"/>
      </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="Opaque">
      <xs:restriction base="xs:hexBinary"/>
    </xs:simpleType>

    <xs:simpleType name="IpAddress">
      <xs:restriction base="xs:string">
        <xs:pattern value=
        "((0|1[0-9]{0,2}|2([0-4][0-9]?|5[0-5]?|[6-9])?|[3-9][0-9]?)\.){3}
        (0|1[0-9]{0,2}|2([0-4][0-9]?|5[0-5]?|[6-9])?|[3-9][0-9]?)"/>
      </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="ObjectIdentifier">
      <xs:restriction base="xs:string">
        <xs:pattern value=
        "[0-2](\.[1-3]?[0-9])(\.(0|([1-9]\d*))){0,126}"/>
      </xs:restriction>
    </xs:simpleType>

  </xs:schema>
```

[TODO]Decisions needed on namespace issues [RFC3688]. One reviewer suggested (until we have an RFC):

- o "urn:ietf:params:xml:ns:opsawg:smi:v1.0" and
- o "urn:ietf:params:xml:schema:draft-ietf-opsawg-smi-datatypes-in-xsd-00.txt"

[DISCUSS] The "BITS" pseudo-type is treated as a Textual Convention for the purpose of this document and, therefore, will be defined in the associated "SNMP Textual Conventions in XSD" document.

[DISCUSS] Should we include value and pattern restriction language from the SMI specifications in the XSD as either "documentation" or "appInfo" "annotations" -- or keep the XSD as simple as possible and merely refer the reader to the relevant sections of those specifications?

5. Rationale

The XSD datatypes, including any specified restrictions, were chosen based on fit with the requirements specified earlier in this document, and with attention to simplicity while maintaining fidelity to the SMI. Also, the "canonical representations" (i.e., refinements of the "lexical representations") documented in the W3C XSD specifications are assumed.

[DISCUSS] The use of "canonical representations" in the XSD specs might merit review by others. This author's (Natale) understanding might not be complete or correct.

5.1. Numeric Datatypes

All of the numeric XSD datatypes specified in the previous section -- INTEGER, Integer32, Unsigned32, Gauge32, Counter32, TimeTicks, and Counter64 -- comply with the relevant requirements:

- o They cover all valid values for the corresponding SMI datatypes.
- o They comply with the standard encoding rules associated with the corresponding SMI datatypes.
- o They inherently match the range restrictions associated with the corresponding SMI datatypes.
- o They are the most direct XSD datatype which exhibit the foregoing characteristics relative to the corresponding SMI datatypes (which is why no "restriction" statements -- other than the "base" XSD type -- are required in the XSD).
- o The XML output produced from the canonical representation of these XSD datatypes is also the most direct from the perspective of readability by humans (i.e., no leading "+" sign and no leading zeros).

5.2. OctetString

This XSD datatype corresponds to the SMI "OCTET STRING" datatype.

Several independent schemes for mapping SMI datatypes to XSD have used the XSD "string" type to represent "OCTET STRING", but this mapping does not conform to the requirements specified in this document. Most notably, "string" cannot faithfully represent all valid values (0 thru 255) that each octet in an "OCTET STRING" can have -- or at least cannot do so in a way that provides for ready human readability of the resulting XML output.

Consequently, the XSD datatype "hexBinary" is specified as the standard mapping of the SMI "OCTET STRING" datatype. In hexBinary, each octet is encoded as two hexadecimal digits; the canonical representation limits the set of allowed hexadecimal digits to 0-9 and uppercase A-F.

The hexBinary representation of OCTET STRING complies with the relevant requirements:

- o It covers all valid values for the corresponding SMI datatype.
- o It complies with the standard encoding rules associated with the corresponding SMI datatype.
- o With the "maxLength" restriction to 65535 octets, the XSD datatype specification matches the restrictions associated with the corresponding SMI datatype.
- o It is the most direct XSD datatype which exhibits the foregoing characteristics relative to the corresponding SMI datatype (which must allow for any valid binary octet value).
- o The XML output produced from the canonical representation of this XSD datatype is not optimal with respect to readability by humans; however, that is a consequence of the SMI datatype itself. Where human readability is more of a concern, it is likely that the actual MIB objects in question will be represented by textual conventions which limit the set of values that will be included in the OctetStrings and will, thus, bypass the hexBinary typing.

5.3. Opaque

The "hexBinary" XSD datatype is specified as the representation of the SMI "Opague" datatype generally for the same reasons as "hexBinary" is specified for the "OctetString" datatype.

- o It covers all valid values for the corresponding SMI datatype.
- o It complies with the standard encoding rules associated with the corresponding SMI datatype.

- o There are no restriction issues associated with using "hexBinary" for "Opague".
- o It is the most direct XSD datatype which exhibits the foregoing characteristics relative to the corresponding SMI datatype (which must allow for any valid binary octet value).
- o The XML output produced from the canonical representation of this XSD datatype is not optimal with respect to readability by humans; however, that is a consequence of the SMI datatype itself. Unmediated "Opague" data is intended for consumption by applications, not humans.

[DISCUSS] Does the "Double-wrapping" aspect of "Opague" in the SMI need to be accommodated in the XSD syntax?

5.4. IpAddress

The XSD "string" datatype is the natural choice to represent an IpAddress as XML output. The "pattern" restriction applied in this case results in a "dotted-decimal string of four values between "0" and "255" separated by a period (".") character. This pattern also precludes leading zeros.

[DISCUSS] Is the leading-zeros restriction appropriate? It is specified here for the following reasons: Enhances human readability, conforms to the most common way of representing IpAddress values, and conforms to other selections in this document to avoid leading-zeros on numerical output values.

[DISCUSS] Irrespective of the previous discussion topic, can the pattern for IpAddress be simplified further (while still satisfying the core requirements for allowable value sequences)?

5.5. ObjectIdentifier

This XSD datatype corresponds to the SMI "OBJECT IDENTIFIER" datatype.

The XSD "string" datatype is also the natural choice to represent an ObjectIdentifier as XML output, for the same reasons as for the IpAddress choice. The "pattern" restriction applied in this case results in a dotted-decimal string of up to 128 elements (referred to as "sub-ids"), holding "Unsigned32" integer values.

Note that, while not mentioned in Sec. 7.1.3 of RFC 2578, due to the use of Abstract Syntax Notation One (ASN.1) Basic Encoding Rules (BER) the first two sub-ids of an "OBJECT IDENTIFIER" have limited value ranges ([0-2] and [0-39], respectively) and are packed into a single octet [Steedman], and the minimum length of an "OBJECT

IDENTIFIER" is two sub-ids (with a zero-valued "OBJECT IDENTIFIER" represented as "0.0"). No explicit "minLength" restriction (which would be "3" to allow for the minimum of two sub-ids and a single separating dot) is required, since the pattern itself enforces this restriction.

[DISCUSS] The pattern specified for ObjectIdentifier attempts to faithfully capture the restrictions mentioned above. Does it do so correctly and is there a more efficient way of doing so?

6. Security Considerations

Security considerations for any given SMI MIB module are likely to be relevant to any XSD/XML mapping of that MIB module; however, this mapping itself does not introduce any new security considerations.

If and when proxies or gateways are developed to convey SNMP management information from SNMP agents to XML-based management applications via XSD/XML mapping of MIB modules based on this specification and its planned siblings, special care will need to be taken to ensure that all applicable SNMP security mechanisms are supported in an appropriate manner yet to be determined.

7. IANA Considerations

[DISCUSS] We will likely need namespace and location resources from IANA...?.

8. Acknowledgements

Dave Harrington provided strategic and technical leadership to the team which developed this particular specification. Yan Li did much of the research into existing approaches that was used as a baseline for the recommendations in this particular specification.

This document owes much to draft-romascanu-netconf-datatypes-xx and to many other sources (including libsmi and group discussions on the NETCONF mailing lists) developed by those who have researched and published candidate mappings of SMI datatypes and textual conventions to XSD.

Individuals who participated in various discussions of this topic at IETF meetings and on IETF mailing lists include: Sharon Chisholm, David Harrington, Ray Atarashi, Yoshifumi Atarashi, Bert Wijnen, Andy Bierman, Randy Presuhn, Chris Lonvick, Eliot Lear, Avri Doria, Juergen Schoenwaelder, Rob Ennes, Faye Ly and Andre Westerinen.

[TODO] Expand list of participants as appropriate.

9. References

9.1. Normative References

[RFC1155] Rose, M. and K. McCloghrie, "Structure and identification of management information for TCP/IP-based internets", STD 16, RFC 1155, May 1990.

[RFC2119] Bradner, s., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2578] McCloghrie, K., Ed., Perkins, D., Ed., and J. Schoenwaelder, Ed., "Structure of Management Information Version 2 (SMIv2)", STD 58, RFC 2578, April 1999.

[RFC2579] McCloghrie, K., Perkins, D., and J. Schoenwaelder, "Textual Conventions for SMIv2", STD 58, RFC 2579, April 1999.

[RFC3584] Frye, R., Levi, D., Routhier, S., and B. Wijnen, "Coexistence between Version 1, Version 2, and Version 3 of the Internet-standard Network Management Framework", BCP 74, RFC 3584, August 2003.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, January 2004.

[ref.XML] World Wide Web Consortium, "Extensible Markup Language (XML) 1.0", W3C XML, February 1998, <http://www.w3.org/TR/1998/REC-xml-19980210>.

[ref.XMLSchema] World Wide Web Consortium, "XML Schema Part 1: Structures Second Edition", W3C XML Schema, October 2004, <http://www.w3.org/TR/xmlschema-1/>.

[ref.XSDDatatype] World Wide Web Consortium, "XML Schema Part 2: Datatypes Second Edition", W3C XML Schema, October 2004, <http://www.w3.org/TR/xmlschema-2/>.

9.2. Informational References

[Steedman] Steedman, D., "ASN.1: The Tutorial and Reference".

Appendix A. Open Issues

- Resolve all [TODO] items.
- Resolve all [DISCUSS] items.

Appendix B. Change Log

- -00 Initial version
- -01 version:
-
    * Incorporated mailing list comments on -00 version from Juergen Schoenwaelder
    * Incorporated mailing list comments on -00 version from David Harrington

Authors' Addresses

Bob Natale (editor)
MITRE
7515 Colshire Dr
MS H405
McLean, VA 22102
USA

Phone: +1 703-983-2505
EMail: rnatale@mitre.org

Yan Li (editor)
Huawei Technologies
No.3 Xinxi Road, Shangdi Information Industry Base
Beijing, HaiDian District 100085
P.R.China

Phone: +86 10 8288 2008
EMail: liyan_77@huawei.com

Acknowledgement

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).