NETMOD Working Group
Internet-Draft
Updates: 6110 (if approved)
Intended status: Standards Track
Expires: September 22, 2016

L. Lhotka
CZ.NIC

March 21, 2016

Defining and Using Metadata with YANG
draft-ietf-netmod-yang-metadata-07

Abstract

This document defines a YANG extension statement that allows for defining metadata annotations in YANG modules. The document also specifies XML and JSON encoding of annotations and other rules for annotating instances of YANG data nodes.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on September 22, 2016.

Copyright Notice

Copyright (c) 2016 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction 2
2. Terminology 4
2.1. Keywords 5
2.2. Terms Defined in Other Documents 5
2.3. Namespaces and Prefixes 6
2.4. Definitions of New Terms 7
3. Defining Annotations in YANG 7
3.1. Example Definition 8
4. Using Annotations 8
5. The Encoding of Annotations 9
5.1. XML Encoding 9
5.2. JSON Encoding 10
5.2.1. Metadata Object and Annotations 10
5.2.2. Adding Annotations to Anydata, Container and List Entries 11
5.2.3. Adding Annotations to Anyxml and Leaf Instances . . . 11
5.2.4. Adding Annotations to Leaf-list Entries 12
6. Representing Annotations in DSDL Schemas 13
7. Metadata YANG Module 14
8. IANA Considerations 16
9. Security Considerations 17
10. Acknowledgments 17
11. References 17
11.1. Normative References 17
11.2. Informative References 18
Appendix A. Change Log 19
A.1. Changes Between Revisions -06 and -07 19
A.2. Changes Between Revisions -05 and -06 19
A.3. Changes Between Revisions -04 and -05 19
A.4. Changes Between Revisions -03 and -04 19
A.5. Changes Between Revisions -02 and -03 19
A.6. Changes Between Revisions -01 and -02 19
A.7. Changes Between Revisions -00 and -01 20
A.8. Changes Between draft-lhotka-netmod-yang-metadata-01 and draft-ietf-netmod-yang-metadata-00 20
A.9. Changes Between draft-lhotka-netmod-yang-metadata-00 and -01 20
Author's Address 20

1. Introduction

There is a need to be able to annotate instances of YANG [I-D.ietf-netmod-rfc6020bis] data nodes with metadata. Typical use cases are:

- Complementing regular data model information with instance-specific metadata, comments etc.

- Providing information about data rendering in user interfaces.

- Deactivating a subtree in a configuration datastore while keeping the data in place.

- Network management protocols often use metadata annotations for various purposes in both operation requests and responses. For example, the <edit-config> operation in the NETCONF protocol (see section 7.2 of [RFC6241]) uses annotations in the form of XML attributes for identifying the location in a configuration datastore and the type of the operation.

However, metadata annotations could potentially lead to interoperability problems if they are used in an ad hoc fashion by different parties and/or without proper documentation. A sound metadata framework for YANG should therefore satisfy these requirements:

1. The set of annotations must be extensible in a decentralised manner so as to allow for defining new annotations without running into the risk of collisions with annotations defined and used by others.

2. Syntax and semantics of annotations must be documented and the documentation must be easily accessible.

3. Clients of network management protocols such as NETCONF [RFC6241] or RESTCONF [I-D.ietf-netconf-restconf] must be able to discover all annotations supported by a given server and identify each of them correctly.

4. Annotations sent by a server should not break clients that don't support them.

This document proposes a systematic way for defining metadata annotations. For this purpose, YANG extension statement "annotation" is defined in the module "ietf-yang-metadata" (Section 7). Other YANG modules importing this module can use the "annotation" statement for defining one or more annotations.

The benefits of defining the metadata annotations in a YANG module are the following:

- Each annotation is bound to a YANG module name and namespace URI. This makes its encoding in instance documents (both XML and JSON)

straightforward and consistent with the encoding of YANG data node instances.

- Annotations defined in IETF standard-track documents are indirectly registered through IANA in the "YANG Module Names" registry [RFC6020].

- Annotations are included in the data model. YANG compilers and tools supporting a certain annotation can thus take them into account and modify their behavior accordingly.

- Semantics of an annotation are defined in the "description" and "reference" statements.

- An annotation can be declared as conditional by using the "if-feature" statement.

- The type of each annotation is explicitly specified; any YANG built-in or derived type that is available for leaf or leaf-list data nodes may be specified for annotations as well.

In the XML encoding, XML attributes are a natural instrument for attaching annotations to data node instances. This document deliberately adopts some restrictions in order to remain compatible with the XML encoding of YANG data node instances and limitations of XML attributes. Specifically,

- annotations can only be scalar values;

- annotations cannot be attached to a whole list or leaf-list instance, only to individual list or leaf-list entries.

Due to the rules for YANG extensions (see sec. 6.3.1 in [I-D.ietf-netmod-rfc6020bis]), annotation definitions posit relatively weak conformance requirements. The alternative of introducing a new built-in YANG statement for defining annotations was considered, but it was seen as a major change to the language that is inappropriate for YANG 1.1, which was chartered as a maintenance revision. After evaluating real-life usage of metadata annotations, it is conceivable that such a new built-in statement might be added in a future revision of YANG.

2. Terminology

2.1. Keywords

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 [RFC2119].

2.2. Terms Defined in Other Documents

The following terms are defined in [RFC6241]:

- capability,
- client,
- datastore,
- message,
- protocol operation,
- server.

The following terms are defined in [I-D.ietf-netmod-rfc6020bis]:

- action,
- anydata,
- anyxml,
- built-in type,
- container,
- data model,
- data node,
- data tree,
- derived type,
- extension,
- leaf,
- leaf-list,

- list,
- module,
- RPC input and output.

The following terms are defined in [W3C.REC-xml-infoset-20040204]:

- attribute,
- document,
- element.

The following terms are defined in [W3C.REC-xml-names11-20060816]:

- local name,
- namespace name,
- prefix,
- qualified name.

The following terms are defined in [RFC7159]:

- array,
- member,
- object,
- primitive type.

2.3. Namespaces and Prefixes

In the following text, XML element names and YANG extension statements are always written with explicit namespace prefixes that are assumed to be bound to URI references as shown in Table 1.

Prefix	URI Reference
elm	http://example.org/example-last-modified
md	urn:ietf:params:xml:ns:yang:ietf-yang-metadata
rng	http://relaxng.org/ns/structure/1.0

Table 1: Used namespace prefixes and corresponding URI references

2.4. Definitions of New Terms

- annotation: a single item of metadata that is attached to YANG data node instances.
- metadata: additional information that complements a data tree.
- metadata object: an object in JSON encoding that contains all annotations attached to a given data node instance.

3. Defining Annotations in YANG

Metadata annotations are defined by YANG extension statement "md:annotation". This YANG language extension is defined in the module "ietf-yang-metadata" (Section 7).

Substatements of "md:annotation" are shown in Table 2. They are all core YANG statements, and the numbers in the second column refer to the corresponding section in [I-D.ietf-netmod-rfc6020bis] where each statement is described.

substatement	RFC 6020bis section	cardinality
description	7.21.3	0..1
if-feature	7.20.2	0..n
reference	7.21.4	0..1
status	7.21.2	0..1
type	7.6.3	1
units	7.3.3	0..1

Table 2: Substatements of "md:annotation".

An annotation carries a single value. The type substatement, which MUST be present, takes as an argument the name of an existing built-in or derived type, and the value of the annotation MUST match this type. See Section 7.4 of [I-D.ietf-netmod-rfc6020bis] for details.

An annotation can be made conditional by using one or more "if-feature" statements; the annotation is then supported only by servers that advertise the corresponding feature.

The semantics and usage rules for an annotation SHOULD be fully specified in "description", "reference" and "units" statements.

An annotation MUST NOT change the data tree semantics defined by YANG. For example, it is illegal to define and use an annotation that allows for overriding uniqueness of leaf-list entries.

The "status" statement can be used exactly as for YANG data nodes.

A YANG module containing one or more "md:annotation" extension statements SHOULD NOT be used for defining data nodes or groupings. Also, derived types, identities and features SHOULD NOT be defined in such a module unless they are used by the definitions of annotations in that module.

3.1. Example Definition

The following module defines the "last-modified" annotation:

```
module example-last-modified {
  namespace "http://example.org/example-last-modified";
  prefix "elm";
  import ietf-yang-types {
    prefix "yang";
  }
  import ietf-yang-metadata {
    prefix "md";
  }
  md:annotation last-modified {
    type yang:date-and-time;
    description
      "This annotation contains date and time when the
       annotated instance was last modified (or created).";
  }
}
```

4. Using Annotations

By advertising a YANG module in which a metadata annotation is defined using the "md:annotation" statement, a server indicates that it is prepared to handle that annotation according to the annotation's definition. That is, an annotation advertised by the server may be attached to an instance of a data node defined in any YANG module that is implemented by the server.

Depending on its semantics, an annotation may have an effect only in certain data trees and/or on instances of specific data nodes types.

A client MUST NOT add a specific annotation to data node instances if the server didn't advertise it.

Due care has to be exercised when introducing annotations in network management systems in order to avoid interoperability problems and software failures caused by a client that does not understand the annotations' semantics. Generally, it is safe for a server to use annotations in the following cases:

- An annotation is an integral part of a built-in or negotiated protocol capability.
- An annotation contains auxiliary information that is not critical for protocol operation.
- The client explicitly asks the server, e.g., via a parameter of a protocol operation request, for including an annotation in the response.

5. The Encoding of Annotations

XML attributes are a natural choice for encoding metadata in XML instance documents. For JSON [RFC7159], there is no generally established method for encoding metadata. This document thus introduces a special encoding method that is consistent with the JSON encoding of YANG data node instances as defined in [I-D.ietf-netmod-yang-json].

5.1. XML Encoding

Metadata annotations are added to XML-encoded instances of YANG data nodes as XML attributes according to these rules:

- The local name of the attribute SHALL be the same as the name of the annotation specified in the argument of the corresponding "md:annotation" statement.
- The namespace of the attribute SHALL be identified by the URI that appears as the argument of the "namespace" statement in the YANG module where the annotation is defined. It is RECOMMENDED that the prefix specified by the "prefix" statement in the same module is used in the qualified name of the attribute.

- The attribute value SHALL be encoded in the same way as the value of a YANG leaf instance having the same type, see [I-D.ietf-netmod-rfc6020bis], sec. 9.

For example, the "last-modified" annotation defined in Section 3.1 may be encoded as follows:

```
<foo xmlns:elm="http://example.org/example-last-modified"
     elm:last-modified="2015-09-16T10:27:35+02:00">
  ...
</foo>
```

5.2. JSON Encoding

The JSON metadata encoding defined in this section has the following properties:

1. The encoding of YANG data node instances as defined in [I-D.ietf-netmod-yang-json] does not change.

2. Namespaces of metadata annotations are encoded in the same way as namespaces of YANG data node instances, see [I-D.ietf-netmod-yang-json].

5.2.1. Metadata Object and Annotations

All metadata annotations assigned to a YANG data node instance are encoded as members (name/value pairs) of a single JSON object, henceforth denoted as the metadata object. The placement and name of this object depends on the type of the data node as specified in the following subsections.

The name of a metadata annotation (as a member of the metadata object) has the following ABNF syntax [RFC5234], where the production for "identifier" is defined in sec. 13 of [I-D.ietf-netmod-rfc6020bis]

```
annotation-name = identifier ":" identifier
```

where the left identifier is the name of the YANG module in which the annotation is defined, and the identifier on the right is the name of the annotation specified in the argument of the corresponding "md:annotation" statement.

Note that unlike member names of YANG data node instances in JSON encoding (see sec. 4 in [I-D.ietf-netmod-yang-json]), for annotations the explicit namespace identifier (module name) must always be present.

The value of a metadata annotation SHALL be encoded in exactly the same way as the value of a YANG leaf node having the same type as the annotation, see [I-D.ietf-netmod-yang-json], sec. 6.

5.2.2. Adding Annotations to Anydata, Container and List Entries

For a data node instance that is encoded as a JSON object (i.e., a container, list entry, or anydata node), the metadata object is added as a new member of that object with the name "@".

Examples:

o "cask" is a container or anydata node:

```
"cask": {
  "@": {
    "example-last-modified:last-modified":
      "2015-09-16T10:27:35+02:00"
  },
  ...
}
```

o "seq" is a list whose key is "name", annotation "last-modified" is added only to the first entry:

```
"seq": [
  {
    "@": {
      "example-last-modified:last-modified":
          "2015-09-16T10:27:35+02:00"
    },
    "name": "one",
    ...
  },
  {
    "name": "two",
    ...
  }
]
```

5.2.3. Adding Annotations to Anyxml and Leaf Instances

For an anyxml or leaf instance, the metadata object is added as a sibling name/value pair whose name is the symbol "@" concatenated with the name of the leaf or anyxml member that is being annotated. The namespace part (module name) is included if and only if it is in the name of the annotated member.

Examples:

o "flag" is a leaf node of the "boolean" type defined in module "foo", and we assume the namespace name has to be expressed in its JSON encoding:

```
"foo:flag": true,
"@foo:flag": {
  "example-last-modified:last-modified":
    "2015-09-16T10:27:35+02:00"
}
```

o "stuff" is an anyxml node:

```
"stuff": [1, null, "three"],
"@stuff": {
  "example-last-modified:last-modified":
    "2015-09-16T10:27:35+02:00"
}
```

5.2.4. Adding Annotations to Leaf-list Entries

For a leaf-list entry, which is represented as a JSON array with values of a primitive type, annotations may be assigned to one or more entries by adding a name/array pair as a sibling of the leaf-list entry, where the name is symbol "@" concatenated with the name of the leaf-list that is being annotated, and the value is a JSON array whose i-th element is the metadata object with annotations assigned to the i-th entry of the leaf-list entry, or null if the i-th entry has no annotations.

Trailing null values in that array, i.e., those following the last non-null metadata object, MAY be omitted.

For example, in the following leaf-list instance with four entries, the "last-modified" annotation is added to the second and third entry in the following way:

```
"bibliomod:folio": [6, 3, 7, 8],
"@bibliomod:folio": [
  null,
  { "example-last-modified:last-modified":
      "2015-06-18T17:01:14+02:00"
  },
  { "example-last-modified:last-modified":
      "2015-09-16T10:27:35+02:00"
  }
]
```

6. Representing Annotations in DSDL Schemas

[RFC6110] defines the standard mapping of YANG data models to Document Schema Definition Languages (DSDL) [ISO.19757-1]. This section specifies the mapping for the extension statement "md:annotation" (Section 7), which enables validation of XML instance documents containing metadata annotations.

The first step of the DSDL mapping procedure, i.e., the transformation of the YANG data model to the hybrid schema (see sec. 6 in [RFC6110]), is modified as follows:

1. If the data model contains at least one "md:annotation" statement, then a RELAX NG named pattern definition MUST be added as a child of the root <rng:grammar> element in the hybrid schema. It is RECOMMENDED to use the name "__yang_metadata__" for this named pattern.

2. A reference to the named pattern described in item 1 MUST be included as a child of every <rng:element> pattern that corresponds to an anydata, container, leaf, leaf-list or list data node.

3. Every metadata annotation definition in the form

   ```
   md:annotation ARGUMENT {
     ...
   }
   ```

   is mapped to the following RELAX NG pattern:

   ```
   <rng:optional>
     <rng:attribute name="PREFIX:ARGUMENT">
       ...
     </rng:attribute>
   </rng:optional>
   ```

   where PREFIX is the prefix bound to the namespace URI of the YANG module that contains the "md:annotation" statement. The above pattern SHALL be inserted as a child of the named pattern described in item 1.

4. Substatements of "md:annotation" SHALL be mapped to children of the "rng:attribute" pattern exactly as described in sec. 10 of [RFC6110].

For example, the named pattern (item 1), when constructed only for the "last-modified" annotation, will have the following definition:

```
<rng:define name="__yang_metadata__">
  <rng:optional>
    <rng:attribute name="elm:last-modified">
      <rng:ref name="ietf-yang-types__date-and-time"/>
    </rng:attribute>
  </rng:optional>
</rng:define>
```

Every "rng:element" pattern that corresponds to an anydata, container, leaf, list or leaf-list data node will then contain a reference to the above named pattern, for example

```
<rng:element name="foo:bar">
  <rng:ref name="__yang_metadata__"/>
  ...
</rng:element>
```

Note that it is not necessary to use such a reference for "rng:element" patterns corresponding to anyxml data nodes because they already permit any XML attributes to be attached to their instances.

The second step of the DSDL mapping procedure, i.e., the transformation of the hybrid schema to RELAX NG, Schematron and DSRL schemas, is unaffected by the inclusion of "md:annotation".

7. Metadata YANG Module

RFC Editor: In this section, replace all occurrences of 'XXXX' with the actual RFC number and all occurrences of the revision date below with the date of RFC publication (and remove this note).

RFC Editor: Also please replace all occurrences of 'RFC 6020bis' with the actual RFC number that will be assigned to [I-D.ietf-netmod-rfc6020bis].

<CODE BEGINS> file "ietf-yang-metadata@2016-03-21.yang"

```
module ietf-yang-metadata {

  namespace "urn:ietf:params:xml:ns:yang:ietf-yang-metadata";

  prefix "md";

  organization
    "IETF NETMOD (NETCONF Data Modeling Language) Working Group";

  contact
```

```
       "WG Web:   <http://tools.ietf.org/wg/netmod/>
        WG List:  <mailto:netmod@ietf.org>

        WG Chair: Lou Berger
                  <mailto:lberger@labn.net>

        WG Chair: Juergen Schoenwaelder
                  <mailto:j.schoenwaelder@jacobs-university.de>

        WG Chair: Kent Watsen
                  <mailto:kwatsen@juniper.net>

        Editor:   Ladislav Lhotka
                  <mailto:lhotka@nic.cz>";

    description
      "This YANG module defines an extension statement that allows for
       defining metadata annotations.

       Copyright (c) 2016 IETF Trust and the persons identified as
       authors of the code. All rights reserved.

       Redistribution and use in source and binary forms, with or
       without modification, is permitted pursuant to, and subject to
       the license terms contained in, the Simplified BSD License set
       forth in Section 4.c of the IETF Trust's Legal Provisions
       Relating to IETF Documents
       (http://trustee.ietf.org/license-info).

       This version of this YANG module is part of RFC XXXX
       (http://tools.ietf.org/html/rfcXXXX); see the RFC itself for
       full legal notices.";

    revision 2016-03-21 {
      description
        "Initial revision.";
      reference
        "RFC XXXX: Defining and Using Metadata with YANG";
    }

    extension annotation {
      argument name;
      description
        "This extension allows for defining metadata annotations in
         YANG modules. The 'md:annotation' statement can appear only at
         the top level of a YANG module or submodule, i.e. it becomes a
         new alternative in the ABNF production rule for 'body-stmts'
         (sec. 14 in RFC 6020bis).
```

```
        The argument of the 'md:annotation' statement defines the name
        of the annotation. Syntactically it is a YANG identifier as
        defined in RFC 6020bis, sec. 6.2.

        An annotation defined with this extension statement inherits
        the namespace and other context from the YANG module in which
        it is defined.

        Data type of the annotation value is specified in the same way
        as for a leaf data node using the 'type' statement.

        Semantics of the annotation and other documentation can be
        specified using the following standard YANG substatements (all
        are optional): 'description', 'if-feature', 'reference',
        'status', and 'units'.

        A server announces support for a particular annotation by
        including the module in which the annotation is defined among
        the advertised YANG modules (e.g., in NETCONF hello message or
        yang-library). The annotation then can be attached to any
        instance of data node defined in any YANG module that is
        advertised by the server.

        XML and JSON encoding of annotations is defined in
        RFC XXXX.";
    }
}

<CODE ENDS>
```

8. IANA Considerations

RFC Editor: In this section, replace all occurrences of 'XXXX' with the actual RFC number and all occurrences of the revision date below with the date of RFC publication (and remove this note).

This document registers a URI in the "IETF XML registry" [RFC3688]. Following the format in RFC 3688, the following registration has been made.

```
---------------------------------------------------------------------
URI: urn:ietf:params:xml:ns:yang:ietf-yang-metadata

Registrant Contact: The NETMOD WG of the IETF.

XML: N/A, the requested URI is an XML namespace.
---------------------------------------------------------------------
```

This document registers a YANG module in the "YANG Module Names" registry [RFC6020].

```
---------------------------------------------------------------------
name:         ietf-yang-metadata
namespace:    urn:ietf:params:xml:ns:yang:ietf-yang-metadata
prefix:       md
reference:    RFC XXXX
---------------------------------------------------------------------
```

9. Security Considerations

This document introduces a mechanism for defining metadata annotations in YANG modules and attaching them to instances of YANG data nodes. By itself, this mechanism represents no security threat. Security implications of a particular annotation defined using this mechanism MUST be duly considered and documented in the the annotation's definition.

An annotation SHOULD be subject to the same or stricter access control rules as the data node instance to which the annotation is attached. It is RECOMMENDED that security-sensitive or privacy-sensitive data be modeled as regular YANG data nodes rather than annotations.

10. Acknowledgments

The author wishes to thank Andy Bierman, Martin Bjorklund, Benoit Claise, Juergen Schoenwaelder, and Kent Watsen for their helpful comments and suggestions.

11. References

11.1. Normative References

[I-D.ietf-netmod-rfc6020bis]
Bjorklund, M., "The YANG 1.1 Data Modeling Language", draft-ietf-netmod-rfc6020bis-11 (work in progress), February 2016.

[I-D.ietf-netmod-yang-json]
Lhotka, L., "JSON Encoding of Data Modeled with YANG", draft-ietf-netmod-yang-json-09 (work in progress), March 2016.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <http://www.rfc-editor.org/info/rfc3688>.

[RFC5234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, DOI 10.17487/RFC5234, January 2008, <http://www.rfc-editor.org/info/rfc5234>.

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <http://www.rfc-editor.org/info/rfc6020>.

[RFC6110] Lhotka, L., Ed., "Mapping YANG to Document Schema Definition Languages and Validating NETCONF Content", RFC 6110, DOI 10.17487/RFC6110, February 2011, <http://www.rfc-editor.org/info/rfc6110>.

[RFC7159] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", RFC 7159, DOI 10.17487/RFC7159, March 2014, <http://www.rfc-editor.org/info/rfc7159>.

[W3C.REC-xml-infoset-20040204] Cowan, J. and R. Tobin, "XML Information Set (Second Edition)", World Wide Web Consortium Recommendation REC-xml-infoset-20040204, February 2004, <http://www.w3.org/TR/2004/REC-xml-infoset-20040204>.

[W3C.REC-xml-names11-20060816] Bray, T., Hollander, D., Layman, A., and R. Tobin, "Namespaces in XML 1.1 (Second Edition)", World Wide Web Consortium Recommendation REC-xml-names11-20060816, August 2006, <http://www.w3.org/TR/2006/REC-xml-names11-20060816>.

11.2. Informative References

[I-D.ietf-netconf-restconf] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", draft-ietf-netconf-restconf-10 (work in progress), March 2016.

[ISO.19757-1]
International Organization for Standardization, "Document Schema Definition Languages (DSDL) - Part 1: Overview", ISO/IEC 19757-1, November 2004.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <http://www.rfc-editor.org/info/rfc6241>.

Appendix A. Change Log

RFC Editor: Remove this section upon publication as an RFC.

A.1. Changes Between Revisions -06 and -07

- Added sentence in Sec. 9 (Stephen Farrell's suggestion).

A.2. Changes Between Revisions -05 and -06

- Added explanation of why a YANG extension is used rather than a built-in statement.

A.3. Changes Between Revisions -04 and -05

- Clarification regarding the type of an annotation.

A.4. Changes Between Revisions -03 and -04

- Added explanation of what "top level of a module" means.

A.5. Changes Between Revisions -02 and -03

- Section 4 was considerably simplified, also because member names starting with "@" are now permitted by [I-D.ietf-netmod-yang-json].

A.6. Changes Between Revisions -01 and -02

- The "type" statement became mandatory.

- Terminology section was extended.

- The annotation "inactive" defined in the example module was replaced with "last-modified" that is supposedly less controversial.

- Introduction now states limitation due to XML attribute properties.
- A recommendation was added to define annotations in a module by themselves.
- Section "Using Annotations" was added.
- An example for "anyxml" was added.
- RFC 6241 was moved to informative references.

A.7. Changes Between Revisions -00 and -01

- Define JSON encoding for annotations attached to 'anydata' nodes.

A.8. Changes Between draft-lhotka-netmod-yang-metadata-01 and draft-ietf-netmod-yang-metadata-00

- References to RFC 6020 were changed to the 6020bis I-D.
- Text about RFC 2119 key words was added to "ietf-yang-metadata" module description.

A.9. Changes Between draft-lhotka-netmod-yang-metadata-00 and -01

- Encoding of annotations for anyxml nodes was changed to be the same as for leafs. This was necessary because anyxml value now needn't be an object.
- It is stated that "md:annotation" statement defines only the syntax of an annotation.
- Allowed "if-feature" as a substatement of "md:annotation".

Author's Address

Ladislav Lhotka
CZ.NIC

Email: lhotka@nic.cz