Network Working Group
Request for Comments: 4181
BCP: 111
Category: Best Current Practice

C. Heard, Ed.

September 2005

Guidelines for Authors and Reviewers of MIB Documents

Status of This Memo

Copyright Notice

Abstract

This memo provides guidelines for authors and reviewers of IETF standards-track specifications containing MIB modules. Applicable portions may be used as a basis for reviews of other MIB documents.

Table of Contents

1. Introduction

Some time ago, the IESG instituted a policy of requiring expert review of IETF standards-track specifications containing MIB modules. These reviews were established to ensure that such specifications follow established IETF documentation practices and that the MIB modules they contain meet certain generally accepted standards of quality, including (but not limited to) compliance with all syntactic and semantic requirements of SMIv2 (STD 58) [RFC2578] [RFC2579] [RFC2580] that are applicable to "standard" MIB modules. The purpose of this memo is to document the guidelines that are followed in such reviews.

Please note that the guidelines in this memo are not intended to alter requirements or prohibitions (in the sense of "MUST", "MUST NOT", "SHALL", or "SHALL NOT" as defined in RFC 2119 [RFC2119]) of existing BCPs or Internet Standards except where those requirements or prohibitions are ambiguous or contradictory. In the exceptional cases where ambiguities or contradictions exist, this memo documents the current generally accepted interpretation. In certain instances, the guidelines in this memo do alter recommendations (in the sense of "SHOULD", "SHOULD NOT", "RECOMMENDED", or "NOT RECOMMENDED" as defined in RFC 2119) of existing BCPs or Internet Standards. This has been done where practical experience has shown that the published recommendations are suboptimal. In addition, this memo provides guidelines for the selection of certain SMIv2 options (in the sense of "MAY" or "OPTIONAL" as defined in RFC 2119) in cases where there is a consensus on a preferred approach.

Although some of the guidelines in this memo are not applicable to non-standards track or non-IETF MIB documents, authors and reviewers of those documents should consider using the ones that do apply.

Reviewers and authors need to be aware that some of the guidelines in this memo do not apply to MIB modules that have been translated to SMIv2 from SMIv1 (STD 16) [RFC1155] [RFC1212] [RFC1215].

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL", when used in the guidelines in this memo, are to be interpreted as described in RFC 2119 [RFC2119].

The terms "MIB module" and "information module" are used interchangeably in this memo. As used here, both terms refer to any of the three types of information modules defined in Section 3 of RFC 2578 [RFC2578].

The term "standard", when it appears in quotes, is used in the same sense as in the SMIv2 documents [RFC2578] [RFC2579] [RFC2580]. In particular, it is used to refer to the requirements that those documents levy on "standard" modules or "standard" objects.

3. General Documentation Guidelines

In general, IETF standards-track specifications containing MIB modules are subject to the same requirements as IETF standards-track RFCs (see [RFC2223bis]), although there are some differences. In particular, since the version under review will be an Internet-Draft, the notices on the front page MUST comply with the requirements of http://www.ietf.org/ietf/1id-guidelines.txt and not with those of [RFC2223bis]. In addition, since the specification under review is expected to be submitted to the IESG, it MUST comply with certain requirements that do not necessarily apply to RFCs; see http://www.ietf.org/ID-Checklist.html for details.

Section 4 of [RFC2223bis] lists the sections that may exist in an RFC. Sections from the abstract onward may also be present in an Internet-Draft; see http://www.ietf.org/ID-Checklist.html. The "body of memo" is always required, and in a MIB document MUST contain at least the following:

- MIB boilerplate section
- Narrative sections
- Definitions section
- Security Considerations section
- IANA Considerations section
- References section

Section-by-section guidelines follow.

3.1. MIB Boilerplate Section

This section MUST contain a verbatim copy of the latest approved Internet-Standard Management Framework boilerplate, which is available on-line at http://www.ops.ietf.org/mib-boilerplate.html.

3.2. Narrative Sections

The narrative part MUST include an overview section that describes the scope and field of application of the MIB modules defined by the specification and that specifies the relationship (if any) of these MIB modules to other standards, particularly to standards containing other MIB modules. The narrative part SHOULD include one or more sections to briefly describe the structure of the MIB modules defined in the specification.

If the MIB modules defined by the specification import definitions from other MIB modules (except for those defined in the SMIv2 documents [RFC2578] [RFC2579] [RFC2580]) or are always implemented in conjunction with other MIB modules, then those facts MUST be noted in the overview section, as MUST any special interpretations of objects in other MIB modules. For instance, so-called media-specific MIB modules are always implemented in conjunction with the IF-MIB [RFC2863] and are REQUIRED to document how certain objects in the IF-MIB are used. In addition, media-specific MIB modules that rely on the ifStackTable [RFC2863] and the ifInvStackTable [RFC2864] to maintain information regarding configuration and multiplexing of interface sublayers MUST contain a description of the layering model.

3.3. Definitions Section

This section contains the MIB module(s) defined by the specification. These MIB modules MUST be written in SMIv2 [RFC2578] [RFC2579] [RFC2580]; SMIv1 [RFC1155] [RFC1212] [RFC1215] has "Not Recommended" status [RFC3410] and is no longer acceptable in IETF MIB modules.

See Section 4 for guidelines on SMIv2 usage.

3.4. Security Considerations Section

Each specification that defines one or more MIB modules MUST contain a section that discusses security considerations relevant to those modules. This section MUST be patterned after the latest approved template (available at http://www.ops.ietf.org/mib-security.html). In particular, writable MIB objects that could be especially disruptive if abused MUST be explicitly listed by name and the associated security risks MUST be spelled out; similarly, readable MIB objects that contain especially sensitive information or that raise significant privacy concerns MUST be explicitly listed by name and the reasons for the sensitivity/privacy concerns MUST be explained. If there are no risks/vulnerabilities for a specific category of MIB objects, then that fact MUST be explicitly stated. Failure to mention a particular category of MIB objects will not be equated to a claim of no risks/vulnerabilities in that category;

rather, it will be treated as an act of omission and will result in the document being returned to the author for correction. Remember that the objective is not to blindly copy text from the template, but rather to think and evaluate the risks/vulnerabilities and then state/document the result of this evaluation.

3.5. IANA Considerations Section

In order to comply with IESG policy as set forth in http://www.ietf.org/ID-Checklist.html, every Internet-Draft that is submitted to the IESG for publication MUST contain an IANA Considerations section. The requirements for this section vary depending what actions are required of the IANA.

3.5.1. Documents that Create a New Name Space

If an Internet-Draft defines a new name space that is to be administered by the IANA, then the document MUST include an IANA Considerations section conforming to the guidelines set forth in RFC 2434 [RFC2434] that specifies how the name space is to be administered.

Name spaces defined by MIB documents generally consist of the range of values for some type (usually an enumerated INTEGER) defined by a TEXTUAL-CONVENTION (TC) or of a set of administratively-defined OBJECT IDENTIFIER (OID) values. In each case, the definitions are housed in stand-alone MIB modules that are maintained by the IANA. These IANA-maintained MIB modules are separate from the MIB modules defined in standards-track specifications so that new assignments can be made without having to republish a standards-track RFC. Examples of IANA-maintained MIB modules include the IANAifType-MIB (http://www.iana.org/assignments/ianaiftype-mib), which defines a name space used by the IF-MIB [RFC2863], and the IANA-RTPROTO-MIB (http://www.iana.org/assignments/ianaiprouteprotocol-mib), which defines a name space used by the IPMROUTE-STD-MIB [RFC2932].

The current practice for such cases is to include a detailed IANA Considerations section complying with RFC 2434 in the DESCRIPTION clause of the MODULE-IDENTITY invocation in each IANA-maintained MIB module and for the IANA Considerations section of the MIB document that defines the name spaces to refer to the URLs for the relevant modules. See RFC 2932 [RFC2932] for an example. This creates a chicken-and-egg problem for MIB documents that have not yet been published as RFCs because the relevant IANA-maintained MIB modules will not yet exist. The accepted way out of this dilemma is to include the initial content of each IANA-maintained MIB module in a non-normative section of the initial issue of the document that defines the name space; for an example, see [RFC1573], which

documents the initial version of the IANAifType-MIB. That material is usually omitted from subsequent updates to the document since the IANA-maintained modules are then available on-line (cf. [RFC2863]).

Reviewers of draft MIB documents to which these considerations apply MUST check that the IANA Considerations section proposed for publication in the RFC is present and contains pointers to the appropriate IANA-maintained MIB modules. Reviewers of Internet Drafts that contain the proposed initial content of IANA-maintained MIB modules MUST also verify that the DESCRIPTION clauses of the MODULE-IDENTITY invocations contain an IANA Considerations section compliant with the guidelines in RFC 2434.

3.5.2. Documents that Require Assignments in Existing Namespace(s)

If an Internet-Draft requires the IANA to update an existing registry prior to publication as an RFC, then the IANA Considerations section in the draft MUST document that fact. MIB documents that contain the initial version of a MIB module will generally require that the IANA assign an OBJECT IDENTIFIER value for the MIB module's MODULE-IDENTITY value and possibly to make other assignments as well. Internet-Drafts containing such MIB modules MUST contain an IANA Considerations section that specifies the registries that are to be updated, the descriptors to which OBJECT IDENTIFIER values are being assigned, and the subtrees under which the values are to be allocated. The text SHOULD be crafted so that after publication it will serve to document the OBJECT IDENTIFIER assignments. For example, something along the following lines would be appropriate for an Internet-Draft containing a single MIB module with MODULE-IDENTITY descriptor powerEthernetMIB that is to be assigned a value under the 'mib-2' subtree:

> The MIB module in this document uses the following IANA-assigned OBJECT IDENTIFIER values recorded in the SMI Numbers registry:
>
> | Descriptor | OBJECT IDENTIFIER value |
> |---|---|
> | powerEthernetMIB | { mib-2 XXX } |
>
> Editor's Note (to be removed prior to publication): the IANA is requested to assign a value for "XXX" under the 'mib-2' subtree and to record the assignment in the SMI Numbers registry. When the assignment has been made, the RFC Editor is asked to replace "XXX" (here and in the MIB module) with the assigned value and to remove this note.

Note well: prior to official assignment by the IANA, a draft document MUST use placeholders (such as "XXX" above) rather than actual numbers. See Section 4.5 for an example of how this is done in a draft MIB module.

3.5.3. Documents with No IANA Requests

If an Internet-Draft makes no requests of the IANA, then that fact MUST be documented in the IANA Considerations section. When an Internet-Draft contains an update of a previously published MIB module, it typically will not require any action on the part of the IANA, but it may inherit an IANA Considerations section documenting existing assignments from the RFC that contains the previous version of the MIB module. In such cases, the draft MUST contain a note (to be removed prior to publication) explicitly indicating that nothing is required from the IANA. For example, a draft that contains an updated version of the POWER-ETHERNET-MIB [RFC3621] might include an IANA Considerations section such as the following:

> The MIB module in this document uses the following IANA-assigned OBJECT IDENTIFIER values recorded in the SMI Numbers registry:
>
> | Descriptor | OBJECT IDENTIFIER value |
> |---|---|
> | powerEthernetMIB | { mib-2 105 } |
>
> Editor's Note (to be removed prior to publication): this draft makes no additional requests of the IANA.

If an Internet-Draft makes no requests of the IANA and there are no existing assignments to be documented, then suitable text for the draft would be something along the following lines:

> No IANA actions are required by this document.

3.6. References Sections

Section 4.7f of [RFC2223bis] specifies the requirements for the references sections in an RFC; http://www.ietf.org/ID-Checklist.html imposes the same requirements on Internet-Drafts. In particular, there MUST be separate lists of normative and informative references, each in a separate section. The style SHOULD follow that of recently published RFCs.

The standard MIB boilerplate available at http://www.ops.ietf.org/mib-boilerplate.html includes lists of normative and informative references that MUST appear in all IETF

specifications that contain MIB modules. If items from other MIB modules appear in an IMPORTS statement in the Definitions section, then the specifications containing those MIB modules MUST be included in the list of normative references. When items are imported from an IANA-maintained MIB module, the corresponding normative reference SHALL point to the on-line version of that MIB module. It is the policy of the RFC Editor that all references must be cited in the text; such citations MUST appear in the overview section where documents containing imported definitions (other than those already mentioned in the MIB boilerplate) are required to be mentioned (cf. Section 3.2).

In general, each normative reference SHOULD point to the most recent version of the specification in question.

3.7. Copyright Notices

IETF MIB documents MUST contain the copyright notices and disclaimer specified in Sections 5.4 and 5.5 of RFC 3978 [RFC3978]. Authors and reviewers MUST check to make sure that the correct year is inserted into these notices. In addition, the DESCRIPTION clause of the MODULE-IDENTITY invocation of each MIB module that will appear in the published RFC MUST contain a pointer to the copyright notices in the RFC. A template suitable for inclusion in an Internet-Draft, appropriate for MIB modules other than those that are to be maintained by the IANA, is as follows:

```
       DESCRIPTION
         " [ ... ]

         Copyright (C) The Internet Society (date). This version
         of this MIB module is part of RFC yyyy; see the RFC
         itself for full legal notices."
-- RFC Ed.: replace yyyy with actual RFC number & remove this note
```

A template suitable for MIB modules that are to be maintained by the IANA is as follows:

```
       DESCRIPTION
         " [ ... ]

         Copyright (C) The Internet Society (date). The initial
         version of this MIB module was published in RFC yyyy;
         for full legal notices see the RFC itself. Supplementary
         information may be available at:
         http://www.ietf.org/copyrights/ianamib.html."
-- RFC Ed.: replace yyyy with actual RFC number & remove this note
```

In each case, the current year is to be inserted in place of the word "date".

3.8. Intellectual Property Section

Section 5 of RFC 3979 [RFC3979] contains a notice regarding intellectual property rights or other rights that must appear in all IETF RFCs. A verbatim copy of that notice SHOULD appear in every IETF MIB document.

4. SMIv2 Usage Guidelines

In general, MIB modules in IETF standards-track specifications MUST comply with all syntactic and semantic requirements of SMIv2 [RFC2578] [RFC2579] [RFC2580] that apply to "standard" MIB modules and except as noted below SHOULD comply with SMIv2 recommendations. The guidelines in this section are intended to supplement the SMIv2 documents in the following ways:

- to document the current generally accepted interpretation when those documents contain ambiguities or contradictions;
- to update recommendations in those documents that have been shown by practical experience to be out-of-date or otherwise suboptimal;
- to provide guidance in selection of SMIv2 options in cases where there is a consensus on a preferred approach.

4.1. Module Names

RFC 2578 Section 3 specifies the rules for module names. Note in particular that names of "standard" modules MUST be unique, MUST follow the syntax rules in RFC 2578 Section 3, and MUST NOT be changed when a MIB module is revised (see also RFC 2578 Section 10).

It is RECOMMENDED that module names be mnemonic. See Appendix C for suggested naming conventions.

4.2. Descriptors, TC Names, and Labels

RFC 2578 Sections 3.1, 7.1.1, and 7.1.4 and RFC 2579 Section 3 recommend that descriptors and names associated with macro invocations and labels associated with enumerated INTEGER and BITS values be no longer than 32 characters, but require that they be no longer than 64 characters.

Restricting descriptors, TC names, and labels to 32 characters often conflicts with the recommendation that they be mnemonic and (for descriptors and TC names) with the requirement that they be unique (see RFC 2578 Section 3.1 and RFC 2579 Section 3). The consensus of the current pool of MIB reviewers is that the SMIv2 recommendation to limit descriptors, TC names, and labels to 32 characters SHOULD be set aside in favor of promoting clarity and uniqueness and that automated tools such as MIB compilers SHOULD NOT by default generate warnings for violating that recommendation.

Note that violations of the 64-character limit MUST NOT be ignored; they MUST be treated as errors.

See Appendix C for suggested descriptor and TC naming conventions.

4.3. Naming Hierarchy

RFC 2578 Section 4 describes the object identifier subtrees that are maintained by IANA and specifies the usages for those subtrees. In particular, the mgmt subtree { iso 3 6 1 2 } is used to identify IETF "standard" objects, while the experimental subtree { iso 3 6 1 3 } is used to identify objects that are under development in the IETF. It is REQUIRED that objects be moved from the experimental subtree to the mgmt subtree when a MIB module enters the IETF standards track.

Experience has shown that it is impractical to move objects from one subtree to another once those objects have seen large-scale use in an operational environment. Hence any object that is targeted for deployment in an operational environment MUST NOT be registered under the experimental subtree, irrespective of the standardization status of that object. The experimental subtree should be used only for objects that are intended for limited experimental deployment. Such objects typically are defined in Experimental RFCs.

Note: the term "object", as used here and in RFC 2578 Section 4, is to be broadly interpreted as any construct that results in an OBJECT IDENTIFIER registration. The list of such constructs is specified in RFC 2578 Section 3.6.

4.4. IMPORTS Statement

RFC 2578 Section 3.2 specifies which symbols must be imported and also lists certain predefined symbols that must not be imported.

The general requirement is that if an external symbol other than a predefined ASN.1 type or the BITS construct is used, then it MUST be mentioned in the module's IMPORTS statement. The words "external object" in the first paragraph of that section may give the

impression that such symbols are limited to those that refer to object definitions, but that is not the case, as subsequent paragraphs should make clear.

Note that exemptions to this general requirement are granted by RFC 2580 Sections 5.4.3 and 6.5.2 for descriptors of objects appearing in the OBJECT clause of a MODULE-COMPLIANCE statement or in the VARIATION clause of an AGENT-CAPABILITIES statement. Some MIB compilers also grant exemptions to descriptors of notifications appearing in a VARIATION clause and to descriptors of object groups and notification groups referenced by a MANDATORY-GROUPS clause, a GROUP clause, or an INCLUDES clause, although RFC 2580 (through apparent oversight) does not mention those cases. The exemptions are sometimes seen as unhelpful because they make IMPORTS rules more complicated and inter-module dependencies less obvious than they otherwise would be. External symbols referenced by compliance statements and capabilities statements MAY therefore be listed in the IMPORTS statement; if this is done, it SHOULD be done consistently.

Finally, even though it is not forbidden by the SMI, it is considered poor style to import symbols that are not used, and standards-track MIB modules SHOULD NOT do so.

4.5. MODULE-IDENTITY Invocation

RFC 2578 Section 3 requires that all SMIv2 MIB modules start with exactly one invocation of the MODULE-IDENTITY macro. This invocation MUST appear immediately after the IMPORTS statement.

RFC 2578 Section 5 describes how the various clauses are used. The following additional guidelines apply to all MIB modules over which the IETF has change control:

- If the module was developed by an IETF working group, then the ORGANIZATION clause MUST provide the full name of the working group, and the CONTACT-INFO clause MUST include working group mailing list information. The CONTACT-INFO clause SHOULD also provide a pointer to the working group's web page.

- A REVISION clause MUST be present for each revision of the MIB module, and the UTC time of the most recent REVISION clause MUST match that of the LAST-UPDATED clause. The DESCRIPTION clause associated with each revision MUST state in which RFC that revision appeared and SHOULD provide a list of all significant changes. When a MIB module is revised, UTC times in all REVISION clauses SHOULD be updated to use four-digit year notation.

- The value assigned to the MODULE-IDENTITY descriptor MUST be unique and (for IETF standards-track MIB modules) SHOULD reside under the mgmt subtree [RFC2578]. Most often it will be an IANA-assigned value directly under mib-2 [RFC2578], although for media-specific MIB modules that extend the IF-MIB [RFC2863] it is customary to use an IANA-assigned value under transmission [RFC2578]. In the past, some IETF working groups have made their own assignments from subtrees delegated to them by IANA, but that practice has proven problematic and is NOT RECOMMENDED.

While a MIB module is under development, the RFC number in which it will eventually be published is usually unknown and must be filled in by the RFC Editor prior to publication. An appropriate form for the REVISION clause applying to a version under development would be something along the following lines:

```
       REVISION    "200212132358Z"  -- December 13, 2002
       DESCRIPTION "Initial version, published as RFC yyyy."
-- RFC Ed.: replace yyyy with actual RFC number & remove this note
```

Note that after RFC publication, a REVISION clause is present only for published versions of a MIB module and not for interim versions that existed only as Internet-Drafts. Thus, a draft version of a MIB module MUST contain just one new REVISION clause that covers all changes since the last published version (if any).

When the initial version of a MIB module is under development, the value assigned to the MODULE-IDENTITY descriptor will be unknown if an IANA-assigned value is used, because the assignment is made just prior to publication as an RFC. The accepted form for the MODULE-IDENTITY statement in draft versions of such a module is something along the following lines:

```
   <descriptor> MODULE-IDENTITY

       [ ... ]

       ::= { <subtree> XXX }
-- RFC Ed.: replace XXX with IANA-assigned number & remove this note
```

where <descriptor> is whatever descriptor has been selected for the module and <subtree> is the subtree under which the module is to be registered (e.g., mib-2 or transmission). Note that XXX must be temporarily replaced by a number in order for the module to compile.

Note well: prior to official assignment by the IANA, a draft document MUST use a placeholder (such as "XXX" above) rather than an actual number. If trial implementations are desired during the development process, then an assignment under the 'experimental' subtree may be obtained from the IANA (cf. Section 4.3).

4.6. Textual Conventions and Object Definitions

4.6.1. Usage of Data Types

4.6.1.1. INTEGER, Integer32, Gauge32, and Unsigned32

The 32-bit integer data types INTEGER, Integer32, Gauge32, and Unsigned32 are described in RFC 2578 Section 2 and further elaborated in RFC 2578 Sections 7.1.1, 7.1.7, and 7.1.11. The following guidelines apply when selecting one of these data types for an object definition or a textual convention:

- For integer-valued enumerations:

  - INTEGER is REQUIRED; - Integer32, Unsigned32, and Gauge32 MUST NOT be used.

Note that RFC 2578 recommends (but does not require) that integer-valued enumerations start at 1 and be numbered contiguously. This recommendation SHOULD be followed unless there is a valid reason to do otherwise, e.g., to match values of external data or to indicate special cases, and any such special-case usage SHOULD be clearly documented. For an example, see the InetAddressType TC [RFC4001].

Although the SMI allows DEFVAL clauses for integer-valued enumerations to specify the default value either by label or by numeric value, the label form is preferred since all the examples in RFC 2578 are of that form and some tools do not accept the numeric form.

- If the value range is between -2147483648..2147483647 (inclusive) and negative values are possible, then:

  - Integer32 is RECOMMENDED;
  - INTEGER is acceptable;
  - Unsigned32 and Gauge32 MUST NOT be used.

- If the value range is between 0..4294967295 (inclusive) and the value of the information being modelled may increase above the maximum value or decrease below the minimum value, then:

- Gauge32 is RECOMMENDED;
- Unsigned32 is acceptable;
- INTEGER and Integer32 MUST NOT be used if values greater than 2147483647 are possible.

- If the value range is between 0..4294967295 (inclusive), and values greater than 2147483647 are possible, and the value of the information being modelled does not increase above the maximum value nor decrease below the minimum value, then:

  - Unsigned32 is RECOMMENDED;
  - Gauge32 is acceptable;
  - INTEGER and Integer32 MUST NOT be used.

- If the value range is between 0..2147483647 (inclusive), and the value of the information being modelled does not increase above the maximum value nor decrease below the minimum value, then:

  - Unsigned32 is RECOMMENDED;
  - INTEGER, Integer32, and Gauge32 are acceptable.

- For integer-valued objects that appear in an INDEX clause or for integer-valued TCs that are to be used in an index column:

  - Unsigned32 with a range that excludes zero is RECOMMENDED for most index objects. It is acceptable to include zero in the range when it is semantically significant or when it is used as the index value for a unique row with special properties. Such usage SHOULD be clearly documented in the DESCRIPTION clause.

  - Integer32 or INTEGER with a non-negative range is acceptable. Again, zero SHOULD be excluded from the range except when it is semantically significant or when it is used as the index value for a unique row with special properties, and in such cases the usage SHOULD be clearly documented in the DESCRIPTION clause.

  - Use of Gauge32 is acceptable for index objects that have gauge semantics.

The guidelines above combine both the usage rules for integer data types and the INDEX rules in RFC 2578 Section 7.7 up to and including bullet (1) plus the next-to-last paragraph on page 28.

Sometimes it will be necessary for external variables to represent values of an index object -- e.g., ifIndex [RFC2863]. In such cases, authors of the module containing that object SHOULD consider defining TCs such as InterfaceIndex and/or InterfaceIndexOrZero [RFC2863].

Note that INTEGER is a predefined ASN.1 type and MUST NOT be present in a module's IMPORTS statement, whereas Integer32, Gauge32, and Unsigned32 are defined by SNMPv2-SMI and MUST be imported from that module if used.

4.6.1.2. Counter32 and Counter64

Counter32 and Counter64 have special semantics as described in RFC 2578 Sections 7.1.6 and 7.1.10, respectively. Object definitions MUST (and textual conventions SHOULD) respect these semantics. That means:

- It is OK to use Counter32/64 for counters that may/will be reset when the management subsystem is re-initialized or when other unusual/irregular events occur (e.g., counters maintained on a line card may be reset when the line card is reset). However, if it is possible for such other unusual/irregular events to occur, the DESCRIPTION clause MUST state that this is so and MUST describe those other unusual/irregular events in sufficient detail that it is possible for a management application to determine whether a reset has occurred since the last time the counter was polled. The RECOMMENDED way to do this is to provide a discontinuity indicator as described in RFC 2578 Sections 7.1.6 and 7.1.10. For an example of such a discontinuity indicator, see the ifCounterDiscontinuityTime object in the IF-MIB [RFC2863].

- It is NOT OK to put in the DESCRIPTION clause of a Counter32/64 that there is a requirement that on a discontinuity the counter MUST reset to zero or to any other specific value.

- It is NOT OK to put in the DESCRIPTION clause of a Counter32/64 that there is a requirement that it MUST reset at any specific time/event (e.g., midnight).

- It is NOT OK for one manager to request the agent to reset the value(s) of counter(s) to zero, and Counter32/64 is the wrong syntax for "counters" that regularly reset themselves to zero. For the latter, it is better to define or use textual conventions such as those in RFC 3593 [RFC3593] or RFC 3705 [RFC3705].

RFC 2578 Section 7.1.10 places a requirement on "standard" MIB modules that the Counter64 type may be used only if the information being modelled would wrap in less than one hour if the Counter32 type was used instead. Now that SNMPv3 is an Internet Standard and SNMPv1 is Historic (see http://www.rfc-editor.org/rfcxx00.html for status and [RFC3410] for rationale), there is no reason to continue enforcing this restriction. Henceforth "standard" MIB modules MAY use the Counter64 type when it makes sense to do so, and MUST use

Counter64 if the information being modelled would wrap in less than one hour if the Counter32 type was used instead. Note also that there is no longer a requirement to define Counter32 counterparts for each Counter64 object, although one is still allowed to do so.

There also exist closely-related textual conventions ZeroBasedCounter32 and ZeroBasedCounter64 defined in RMON2-MIB [RFC2021] and HCNUM-TC [RFC2856], respectively.

The only difference between ZeroBasedCounter32/64 TCs and Counter32/64 is their starting value; at time=X, where X is their minimum-wrap-time after they were created, the behavior of ZeroBasedCounter32/64 becomes exactly the same as Counter32/64. Thus, the preceding paragraphs/rules apply not only to Counter32/64, but also to ZeroBasedCounter32/64 TCs.

4.6.1.3. CounterBasedGauge64

SMIv2 unfortunately does not provide 64-bit integer base types. In order to make up for this omission, the CounterBasedGauge64 textual convention is defined in HCNUM-TC [RFC2856]. This TC uses Counter64 as a base type, but discards the special counter semantics, which is allowed under the generally accepted interpretation of RFC 2579 Section 3.3. It does inherit all the syntactic restrictions of that type, which means that it MUST NOT be subtyped and that objects defined with it MUST NOT appear in an INDEX clause, MUST NOT have a DEFVAL clause, and MUST have a MAX-ACCESS of read-only or accessible-for-notify.

This TC SHOULD be used for object definitions that require a 64-bit unsigned data type with gauge semantics. If a 64-bit unsigned data type with different semantics is needed, then a different TC based on Counter64 MUST be used, since one TC cannot refine another (cf. RFC 2579 Section 3.5).

4.6.1.4. OCTET STRING

The OCTET STRING type is described in RFC 2578 Section 7.1.2. It represents arbitrary binary or textual data whose length is between 0 and 65535 octets inclusive. Objects and TCs whose SYNTAX is of this type SHOULD have a size constraint when the actual bounds are more restrictive than the SMI-imposed limits. This is particularly true for index objects. Note, however, that size constraints SHOULD NOT be imposed arbitrarily, as the SMI does not permit them to be changed afterward.

There exist a number of standard TCs that cater to some of the more common requirements for specialized OCTET STRING types. In particular, SNMPv2-TC [RFC2579] contains the DisplayString, PhysAddress, MacAddress, and DateAndTime TCs; the SNMP-FRAMEWORK-MIB [RFC3411] contains the SnmpAdminString TC; and the SYSAPPL-MIB [RFC2287] contains the Utf8String and LongUtf8String TCs. When a standard TC provides the desired semantics, it SHOULD be used in an object's SYNTAX clause instead of OCTET STRING or an equivalent locally-defined TC.

Note that OCTET STRING is a predefined ASN.1 type and MUST NOT be present in a module's IMPORTS statement.

4.6.1.5. OBJECT IDENTIFIER

The OBJECT IDENTIFIER type is described in RFC 2578 Section 7.1.3. Its instances represent administratively assigned names. Note that both the SMI and the SNMP protocol limit instances of this type to 128 sub-identifiers and require that each sub-identifier be within the range 0 to 4294967295 inclusive. Subtyping is not allowed.

The purpose of OBJECT IDENTIFIER values is to provide authoritative identification either for some type of item or for a specific instance of some type of item. Among the items that can be identified in this way are definitions in MIB modules created via the MODULE-IDENTITY, OBJECT-IDENTITY, OBJECT-TYPE, NOTIFICATION-TYPE, OBJECT-GROUP, NOTIFICATION-GROUP, MODULE-COMPLIANCE, and AGENT-CAPABILITIES constructs; and via instances of objects defined in MIB modules, protocols, languages, specifications, interface types, hardware, and software. For some of these uses other possibilities exist, e.g., OCTET STRING or enumerated INTEGER values. The OBJECT IDENTIFIER type SHOULD be used instead of the alternatives when the set of identification values needs to be independently extensible without the need for a registry to provide centralized coordination.

There exist a number of standard TCs that cater to some of the more common requirements for specialized OBJECT IDENTIFIER types. In particular, SNMPv2-TC [RFC2579] contains the AutonomousType, VariablePointer, and RowPointer TCs. When a standard TC provides the desired semantics, it SHOULD be used in an object's SYNTAX clause instead of OBJECT IDENTIFIER or an equivalent locally-defined TC.

Note that OBJECT IDENTIFIER is a predefined ASN.1 type and MUST NOT be present in a module's IMPORTS statement.

4.6.1.6. The BITS Construct

The BITS construct is described in RFC 2578 Section 7.1.4. It represents an enumeration of named bits. The bit positions in a TC or object definition whose SYNTAX is of this type MUST start at 0 and SHOULD be contiguous.

Note that the BITS construct is defined by the macros that use it and therefore MUST NOT be present in a module's IMPORTS statement.

4.6.1.7. IpAddress

The IpAddress type described in RFC 2578 Section 7.1.5 SHOULD NOT be used in new MIB modules. The InetAddress/InetAddressType textual conventions [RFC4001] SHOULD be used instead.

4.6.1.8. TimeTicks

The TimeTicks type is described in RFC 2578 Section 7.1.8. It represents the time in hundredths of a second between two epochs, reduced modulo 2^32. It MUST NOT be subtyped, and the DESCRIPTION clause of any object or TC whose SYNTAX is of this type MUST identify the reference epochs.

The TimeTicks type SHOULD NOT be used directly in definitions of objects that are snapshots of sysUpTime [RFC3418]. The TimeStamp TC [RFC2579] already conveys the desired semantics and SHOULD be used instead.

4.6.1.9. TruthValue

The TruthValue TC is defined in SNMPv2-TC [RFC2579]. It is an enumerated INTEGER type that assumes the values true(1) and false(2).

This TC SHOULD be used in the SYNTAX clause of object definitions that require a Boolean type. MIB modules SHOULD NOT use enumerated INTEGER types or define TCs that duplicate its semantics.

4.6.1.10. Other Data Types

There exist a number of standard TCs that cater to some of the more common requirements for specialized data types. Some have been mentioned above, and Appendix B contains a partial list that includes those plus some others that are a bit more specialized. An on-line version of that list, which is updated as new TCs are developed, can be found at http://www.ops.ietf.org/mib-common-tcs.html.

Whenever a standard TC already conveys the desired semantics, it SHOULD be used in an object definition instead of the corresponding base type or a locally-defined TC. This is especially true of the TCs defined in SNMPv2-TC [RFC2579] and SNMP-FRAMEWORK-MIB [RFC3411] because they are Internet Standards, and so modules that refer to them will not suffer delay in advancement on the standards track on account of such references.

MIB module authors need to be aware that enumerated INTEGER or BITS TCs may in some cases be extended with additional enumerated values or additional bit positions. When an imported TC that may be extended in this way is used to define an object that may be written or that serves as an index in a read-create table, then the set of values or bit positions that needs to be supported SHOULD be specified either in the object's DESCRIPTION clause or in an OBJECT clause in the MIB module's compliance statement(s). This may be done by explicitly listing the required values or bit positions, or it may be done by stating that an implementation may support a subset of values or bit positions of its choosing.

4.6.2. DESCRIPTION and REFERENCE Clauses

It is hard to overemphasize the importance of an accurate and unambiguous DESCRIPTION clause for all objects and TCs. The DESCRIPTION clause contains the instructions that implementors will use to implement an object, and if they are inadequate or ambiguous, then implementation quality will suffer. Probably the single most important job of a MIB reviewer is to ensure that DESCRIPTION clauses are sufficiently clear and unambiguous to allow interoperable implementations to be created.

A very common problem is to see an object definition for, say, 'stdMIBPoofpoofCounter' with a DESCRIPTION clause that just says "Number of poofpoofs" with no indication what a 'poofpoof' is. In such cases, it is strongly RECOMMENDED that there either be at least a minimal explanation or else a REFERENCE clause to point to the definition of a 'poofpoof'.

For read-write objects (other than columns in read-create tables that have well-defined persistence properties), it is RECOMMENDED that the DESCRIPTION clause specify what happens to the value after an agent reboot. Among the possibilities are that the value remains unchanged, that it reverts to a well-defined default value, or that the result is implementation-dependent.

4.6.3. DISPLAY-HINT Clause

The DISPLAY-HINT clause is used in a TC to provide a nonbinding hint to a management application as to how the value of an instance of an object defined with the syntax in the TC might be displayed. Its presence is optional.

Although management applications typically default to decimal format ("d") for integer TCs that are not enumerations and to a hexadecimal format ("1x:" or "1x " or "1x_") for octet string TCs when the DISPLAY-HINT clause is absent, it should be noted that SMIv2 does not actually specify any defaults. MIB authors should be aware that a clear hint is provided to applications only when the DISPLAY-HINT clause is present.

4.6.4. Conceptual Table Definitions

RFC 2578 Sections 7.1.12 and 7.1.12.1 specify the rules for defining conceptual tables, and RFC 2578 Sections 7.7, 7.8, and 7.8.1 specify conceptual table indexing rules. The following guidelines apply to such definitions:

- For conceptual rows:

  - If the row is an extension of a row in some other table, then an AUGMENTS clause MUST be used if the relationship is one-to-one, and an INDEX clause MUST be used if the relationship is sparse. In the latter case, the INDEX clause SHOULD be identical to that of the original table.

  - If the row is an element of an expansion table -- that is, if multiple row instances correspond to a single row instance in some other table -- then an INDEX clause MUST be used, and the first-mentioned elements SHOULD be the indices of that other table, listed in the same order.

  - If objects external to the row are present in the INDEX clause, then the conceptual row's DESCRIPTION clause MUST specify how those objects are used in identifying instances of its columnar objects, and in particular MUST specify for which values of those index objects the conceptual row may exist.

  - Use of the IMPLIED keyword is NOT RECOMMENDED for any index object that may appear in the INDEX clause of an expansion table. Since this keyword may be associated only with the last object in an INDEX clause, it cannot be associated with the same index object in a primary table and an expansion table. This will cause the sort order to be different in the primary table and any

expansion tables. As a consequence, an implementation will be unable to reuse indexing code from the primary table in expansion tables, and data structures meant to be extended might actually have to be replicated. Designers who are tempted to use IMPLIED should consider that the resulting sort order rarely meets user expectations, particularly for strings that include both uppercase and lowercase letters, and it does not take the user language or locale into account.

- If dynamic row creation and/or deletion by management applications is supported, then:

  - There SHOULD be one columnar object with a SYNTAX value of RowStatus [RFC2579] and a MAX-ACCESS value of read-create. This object is called the status column for the conceptual row. All other columnar objects MUST have a MAX-ACCESS value of read-create, read-only, accessible-for-notify, or not-accessible; a MAX-ACCESS value of read-write is not allowed.

  - There either MUST be one columnar object with a SYNTAX value of StorageType [RFC2579] and a MAX-ACCESS value of read-create, or else the row object (table entry) DESCRIPTION clause MUST specify what happens to dynamically-created rows after an agent restart.

  - If the agent itself may also create and/or delete rows, then the conditions under which this can occur MUST be clearly documented in the row object DESCRIPTION clause.

- For conceptual rows that include a status column:

  - The DESCRIPTION clause of the status column MUST specify which columnar objects (if any) have to be set to valid values before the row can be activated. If any objects in cascading tables have to be populated with related data before the row can be activated, then this MUST also be specified.

  - The DESCRIPTION clause of the status column MUST specify whether or not it is possible to modify other columns in the same conceptual row when the status value is active(1). Note that in many cases it will be possible to modify some writable columns when the row is active but not others. In such cases, the DESCRIPTION clause for each writable column SHOULD state whether or not that column can be modified when the row is active, and the DESCRIPTION clause for the status column SHOULD state that modifiability of other columns when the status value is active(1) is specified in the DESCRIPTION clauses for those columns (rather than listing the modifiable columns individually).

- For conceptual rows that include a StorageType column:

  - The DESCRIPTION clause of the StorageType column MUST specify which read-write or read-create columnar objects in permanent(4) rows an agent must, at a minimum, allow to be writable.

Note that RFC 2578 Section 7.8 requires that the lifetime of an instance of a conceptual row that AUGMENTS a base row must be the same as the corresponding instance of the base row. It follows that there is no need for a RowStatus or StorageType column in an augmenting row if one is already present in the base row.

Complete requirements for the RowStatus and StorageType TCs can be found in RFC 2579, in the DESCRIPTION clauses for those TCs.

4.6.5. OID Values Assigned to Objects

RFC 2578 Section 7.10 specifies the rules for assigning OBJECT IDENTIFIER (OID) values to OBJECT-TYPE definitions. In particular:

- A conceptual table MUST have exactly one subordinate object, which is a conceptual row. The OID assigned to the conceptual row MUST be derived by appending a sub-identifier of "1" to the OID assigned to the conceptual table.

- A conceptual row has as many subordinate objects as there are columns in the row; there MUST be at least one. The OID assigned to each columnar object MUST be derived by appending a non-zero sub-identifier, unique within the row, to the OID assigned to the conceptual row.

- A columnar or scalar object MUST NOT have any subordinate objects.

- The last sub-identifier of an OID assigned to any object (be it table, row, column, or scalar) MUST NOT be equal to zero. Note that sub-identifiers of intermediate nodes MAY be equal to zero.

- The OID assigned to an object definition MUST NOT also be assigned to another definition that results in OID registration. RFC 2578 Section 3.6 lists the constructs that create OID registrations.

Although it is not specifically required by the SMI, it is customary (and strongly RECOMMENDED) that object definitions not be registered beneath group definitions, compliance statements, capabilities statements, or notification definitions. It is also customary (and strongly RECOMMENDED) that group definitions, compliance statements,

capabilities statements, and notification definitions not be registered beneath object definitions. See Appendix D for a RECOMMENDED OID assignment scheme.

4.6.6. OID Length Limitations and Table Indexing

As specified in RFC 2578 Section 3.5, all OIDs are limited to 128 sub-identifiers. While this is not likely to cause problems with administrative assignments, it does place some limitations on table indexing. That is true because the length limitation also applies to OIDs for object instances, and these consist of the concatenation of the "base" OID assigned in the object definition plus the index components. When a table has multiple indices of types such as OCTET STRING or OBJECT IDENTIFIER that resolve to multiple sub-identifiers, then the 128-sub-identifier limit can be quickly reached.

Despite its inconvenience, the 128-sub-identifier limit is not something that can be ignored. In addition to being imposed by the SMI, it is also imposed by the SNMP (see the last paragraph in Section 4.1 of RFC 3416 [RFC3416]). It follows that any table with enough indexing components to violate this limit cannot be read or written using the SNMP and so is unusable. Hence table design MUST take the 128-sub-identifier limit into account. It is RECOMMENDED that all MIB documents make explicit any limitations on index component lengths that management software must observe. This may be done either by including SIZE constraints on the index components or by specifying applicable constraints in the conceptual row DESCRIPTION clause or in the surrounding documentation.

4.7. Notification Definitions

RFC 2578 Section 8 specifies the rules for notification definitions. In particular:

- Inaccessible objects MUST NOT appear in the OBJECTS clause.

- For each object type mentioned in the OBJECTS clause, the DESCRIPTION clause MUST specify which object instance is to be present in the transmitted notification and MUST specify the information/meaning conveyed.

- The OBJECT IDENTIFIER (OID) value assigned to each notification type MUST have a next-to-last sub-identifier of zero, so that it is possible to convert an SMIv2 notification definition into an SMIv1 trap definition and back again without information loss (see [RFC3584] Section 2.1.2) and possible for a multilingual proxy chain to translate an SNMPv2 trap into an SNMPv1 trap and back again without information loss (see [RFC3584] Section 3). In

addition, the OID assigned to a notification definition MUST NOT also be assigned to another definition that results in OID registration. RFC 2578 Section 3.6 lists the constructs that create OID registrations.

Although it is not specifically required by the SMI, it is customary (and strongly RECOMMENDED) that notification definitions not be registered beneath group definitions, compliance statements, capabilities statements, or object definitions (this last is especially unwise, as it may result in an object instance and a notification definition sharing the same OID). It is also customary (and strongly RECOMMENDED) that the OIDs assigned to notification types be leaf OIDs (i.e., that there be no OID registrations subordinate to a notification definition). See Appendix D for a RECOMMENDED OID assignment scheme.

In many cases, notifications will be triggered by external events, and sometimes it will be possible for those external events to occur at a sufficiently rapid rate that sending a notification for each occurrence would overwhelm the network. In such cases, a mechanism MUST be provided for limiting the rate at which the notification can be generated. A common technique is to require that the notification generator use throttling -- that is, to require that it generate no more than one notification for each event source in any given time interval of duration T. The throttling period T MAY be configurable, in which case it is specified in a MIB object, or it MAY be fixed, in which case it is specified in the notification definition. Examples of the fixed time interval technique can be found in the SNMP-REPEATER-MIB [RFC2108] and in the ENTITY-MIB [RFC4133].

4.8. Compliance Statements

RFC 2580 Sections 3, 4, and 5 specify the rules for conformance groups and compliance statements. In particular:

- Every object with a MAX-ACCESS value other than "not-accessible" MUST be contained in at least one object group.

- Every notification MUST be contained in at least one notification group.

- There MUST be at least one compliance statement defined for each "standard" MIB module. It may reside either within that MIB module or within a companion MIB module.

In writing compliance statements, there are several points that are easily overlooked:

- An object group or notification group that is not mentioned either in the MANDATORY-GROUPS clause or in any GROUP clause of a MODULE-COMPLIANCE statement is unconditionally optional with respect to that compliance statement. An alternate way to indicate that an object group or notification group is optional is to mention it in a GROUP clause whose DESCRIPTION clause states that the group is optional. The latter method is RECOMMENDED (for optional groups that are relevant to the compliance statement) in order to make it clear that the optional status is intended rather than being the result of an act of omission.

- If there are any objects with a MAX-ACCESS value of read-write or read-create for which there is no OBJECT clause that specifies a MIN-ACCESS of read-only, then implementations must support write access to those objects in order to be compliant with that MODULE-COMPLIANCE statement. This fact sometimes catches MIB module authors by surprise. When confronted with such cases, reviewers SHOULD verify that this is indeed what the authors intended, since it often is not.

- On the other side of the coin, MIB module authors need to be aware that while a read-only compliance statement is sufficient to support interoperable monitoring applications, it is not sufficient to support interoperable configuration applications. A technique commonly used in MIB modules that are intended to support both monitoring and configuration is to provide both a read-only compliance statement and a full compliance statement. A good example is provided by the DIFFSERV-MIB [RFC3289]. Authors SHOULD consider using this technique when it is applicable.

Sometimes MIB module authors will want to specify that a compliant implementation needs to support only a subset of the values allowed by an object's SYNTAX clause. For accessible objects, this may be done either by specifying the required values in an object's DESCRIPTION clause or by providing an OBJECT clause with a refined SYNTAX in a compliance statement. The latter method is RECOMMENDED for most cases, and is REQUIRED if there are multiple compliance statements with different value subsets required. The DIFFSERV-MIB [RFC3289] illustrates this point. The diffServMIBFullCompliance statement contains the following OBJECT clause. (See Section 4.8.1, "Note Regarding These Examples and RFC 2578".)

```
 OBJECT       diffServDataPathStatus
 SYNTAX       RowStatus { active(1) }
 WRITE-SYNTAX RowStatus { createAndGo(4), destroy(6) }
 DESCRIPTION
    "Support for createAndWait and notInService is not required."
```

whereas the diffServMIBReadOnlyCompliance statement contains this:

```
 OBJECT       diffServDataPathStatus
 SYNTAX       RowStatus { active(1) }
 MIN-ACCESS   read-only
 DESCRIPTION
    "Write access is not required, and active is the only status that
    needs to be supported."
```

One cannot do this for inaccessible index objects because they cannot be present in object groups and cannot be mentioned in OBJECT clauses. There are situations, however, in which one might wish to indicate that an implementation is required to support only a subset of the possible values of some index in a read-create table. In such cases, the requirements MUST be specified either in the index object's DESCRIPTION clause (RECOMMENDED if there is only one value subset) or in the DESCRIPTION clause of a MODULE-COMPLIANCE statement (REQUIRED if the value subset is unique to the compliance statement).

In many cases, a MIB module is always implemented in conjunction with one or more other MIB modules. That fact is REQUIRED to be noted in the surrounding documentation (see Section 3.2 above), and it SHOULD also be noted in the relevant compliance statements. In cases where a particular compliance statement in (say) MIB module A requires the complete implementation of some other MIB module B, then the RECOMMENDED approach is to include a statement to that effect in the DESCRIPTION clause of the compliance statement(s) in MIB module A. It is also possible, however, that MIB module A might have requirements that are different from those that are expressed by any compliance statement of module B -- for example, module A might not require any of the unconditionally mandatory object groups from module B but might require mandatory implementation of an object group from module B that is only conditionally mandatory with respect to the compliance statement(s) in module B. In such cases, the RECOMMENDED approach is for the compliance statement(s) in module A to formally specify requirements with respect to module B via appropriate MODULE, MANDATORY-GROUPS, GROUP, and OBJECT clauses. An example is provided by the compliance statements in the DIFFSERV-MIB [RFC3289], which list the ifCounterDiscontinuityGroup from IF-MIB [RFC2863] as a mandatory group. That group is not sufficient to satisfy any IF-MIB compliance statement, and it is conditionally mandatory in the IF-MIB's current compliance statement ifCompliance3.

4.8.1. Note Regarding These Examples and RFC 2578

There has been some dispute as to whether syntax refinements that restrict enumerations (RFC 2578 Section 9) are permitted with TCs, as shown in the examples above, or are allowed only with the base types

INTEGER and BITS, as suggested by a strict reading of RFC 2578. The rough consensus of the editors of the SMIv2 documents and the current pool of MIB reviewers is that they should be allowed with TCs. MIB module authors should be aware that some MIB compilers follow the strict reading of RFC 2578 and require that the TC be replaced by its base type (INTEGER or BITS) when enumerations are refined. That usage is legal, and it can be found in some older MIB modules such as the IF-MIB [RFC2863].

4.9. Revisions to MIB Modules

RFC 2578 Section 10 specifies general rules that apply any time a MIB module is revised. Specifically:

- The MODULE-IDENTITY invocation MUST be updated to include information about the revision. In particular, the LAST-UPDATED clause value MUST be set to the revision time, a REVISION clause with the same UTC time and an associated DESCRIPTION clause describing the changes MUST be added, and any obsolete information in the existing DESCRIPTION, ORGANIZATION, and CONTACT-INFO clauses MUST be replaced with up-to-date information. See Section 4.5 above for additional requirements that apply to MIB modules that are under IETF change control.

- On the other hand, the module name MUST NOT be changed (except to correct typographical errors), existing definitions (even obsolete ones) MUST NOT be removed from the MIB module, and descriptors and OBJECT IDENTIFIER values associated with existing definitions MUST NOT be changed or re-assigned.

It is important to note that the purpose in forbidding certain kinds of changes is to ensure that a revised MIB module is compatible with fielded implementations based on previous versions of the module. There are two distinct aspects of this backward-compatibility requirement. One is "over the wire" compatibility of agent and manager implementations that are based on different revisions of the MIB module. The other is "compilation" compatibility with MIB modules that import definitions from the revised MIB module. The rules forbidding changing or re-assigning OBJECT IDENTIFIER values are necessary to ensure "over the wire" compatibility; the rules against changing module names or descriptors or removing obsolete definitions are necessary to ensure compilation compatibility.

RFC 2578 Section 10.2 specifies rules that apply to revisions of object definitions. The following guidelines correct some errors in these rules and provide some clarifications:

- Bullet (1) allows the labels of named numbers and named bits in SYNTAX clauses of type enumerated INTEGER or BITS to be changed. This can break compilation compatibility, since those labels may be used by DEFVAL clauses in modules that import the definitions of the affected objects. Therefore, labels of named numbers and named bits MUST NOT be changed when revising IETF MIB modules (except to correct typographical errors), and they SHOULD NOT be changed when revising enterprise MIB modules.

- Although not specifically permitted in bullets (1) through (8), it is generally considered acceptable to add range constraints to the SYNTAX clause of an integer-valued object, provided that the constraints simply make explicit some value restrictions that were implicit in the definition of the object. The most common example is an auxiliary object with a SYNTAX of INTEGER or Integer32 with no range constraint. Since an auxiliary object is not permitted to assume negative values, adding the range constraint (0..2147483647) cannot possibly result in any "over the wire" change, nor will it cause any compilation compatibility problems with a correctly written MIB module. Such a change SHOULD be treated by a reviewer as an editorial change, not as a semantic change. Similarly, removal of a range or size constraint from an object definition when that range or size constraint is enforced by the underlying data type SHOULD be treated by a reviewer as an editorial change.

RFC 2578 Section 10.3 specifies rules that apply to revisions of notification definitions. No clarifications or corrections are required.

RFC 2579 Section 5 specifies rules that apply to revisions of textual convention definitions. The following guideline corrects an error in these rules:

- Bullet (1) allows the labels of named numbers and named bits in SYNTAX clauses of type enumerated INTEGER or BITS to be changed. This can break compilation compatibility, since those labels may be used by DEFVAL clauses in modules that import the definitions of the affected TCs. Therefore, labels of named numbers and named bits MUST NOT be changed when revising IETF MIB modules (except to correct typographical errors), and they SHOULD NOT be changed when revising enterprise MIB modules.

RFC 2580 Section 7.1 specifies rules that apply to revisions of conformance groups. Two point are worth reiterating:

- Objects and notifications MUST NOT be added to or removed from an existing object group or notification group. Doing so could cause a compilation failure or (worse) a silent change in the meaning of a compliance statement or capabilities statement that refers to that group.

- The status of a conformance group is independent of the status of its members. Thus, a current group MAY refer to deprecated objects or notifications. This may be desirable in certain cases, e.g., a set of widely-deployed objects or notifications may be deprecated when they are replaced by a more up-to-date set of definitions, but the conformance groups that contain them may remain current in order to encourage continued implementation of the deprecated objects and notifications.

RFC 2580 Section 7.2 specifies rules that apply to revisions of compliance statements. The following guidelines correct an omission from these rules and emphasize one important point:

- RFC 2580 should (but does not) recommend that an OBJECT clause specifying support for the original set of values be added to a compliance statement when an enumerated INTEGER object or a BITS object referenced by the compliance statement has enumerations or named bits added, assuming that no such clause is already present and that the effective MIN-ACCESS value is read-write or read-create. This is necessary in order to avoid a silent change to the meaning of the compliance statement. MIB module authors and reviewers SHOULD watch for this to ensure that such OBJECT clauses are added when needed. Note that this may not always be possible to do, since affected compliance statements may reside in modules other than the one that contains the revised definition(s).

- The status of a compliance statement is independent of the status of its members. Thus, a current compliance statement MAY refer to deprecated object groups or notification groups. This may be desirable in certain cases, e.g., a set of widely-deployed object or notification groups may be deprecated when they are replaced by a more up-to-date set of definitions, but compliance statements that refer to them may remain current in order to encourage continued implementation of the deprecated groups.

RFC 2580 Section 7.3 specifies rules that apply to revisions of capabilities statements. The following guideline corrects an omission from these rules:

- RFC 2580 should (but does not) recommend that VARIATION clauses specifying support for the original set of values be added to a capabilities statement when enumerated INTEGER objects or BITS

objects referenced by the capabilities statement have enumerations added, assuming that no such clauses are already present. This is necessary in order to avoid a silent change to the meaning of the capabilities statement.

In certain exceptional situations, the cost of strictly following the SMIv2 rules governing MIB module revisions may exceed the benefit. In such cases, the rules can be waived, but when that is done both the change and the justification for it MUST be thoroughly documented. One example is provided by Section 3.1.5 of RFC 2863, which documents the semantic change that was made to ifIndex in the transition from MIB-II [RFC1213] to the IF-MIB [RFC2863] and provides a detailed justification for that change. Another example is provided by the REVISION clause of the SONET-MIB [RFC2558] that documents raising the MAX-ACCESS of several objects to read-write while adding MIN-ACCESS of read-only for compatibility with the previous version [RFC1595].

Authors and reviewers may find it helpful to use tools that can list the differences between two revisions of a MIB module. Please see http://www.ops.ietf.org/mib-review-tools.html for more information.

5. Acknowledgments

Most of the material on usage of data types was based on input provided by Bert Wijnen with assistance from Keith McCloghrie, David T. Perkins, and Juergen Schoenwaelder. Much of the other material on SMIv2 usage was taken from an unpublished guide for MIB authors and reviewers by Juergen Schoenwaelder. Some of the recommendations in these guidelines are based on material drawn from the on-line SMIv2 errata list at http://www.ibr.cs.tu-bs.de/ietf/smi-errata/. Thanks to Frank Strauss and Juergen Schoenwaelder for maintaining that list and to the contributors who supplied the material for that list. Finally, thanks are due to the following individuals whose comments on earlier versions of this memo contained many valuable suggestions for additions, clarifications, and corrections: Andy Bierman, Bob Braden, Michelle Cotton, David Harrington, Harrie Hazewinkel, Dinakaran Joseph, Michael Kirkham, Keith McCloghrie, David T. Perkins, Randy Presuhn, Dan Romascanu, Juergen Schoenwaelder, Frank Strauss, Dave Thaler, and Bert Wijnen.

6. Security Considerations

Implementation and deployment of a MIB module in a system may result in security risks that would not otherwise exist. It is important for authors and reviewers of documents that define MIB modules to ensure that those documents fully comply with the guidelines in http://www.ops.ietf.org/mib-security.html so that all such risks are adequately disclosed.

7. IANA Considerations

This document affects the IANA to the extent that it describes what is required to be present in the IANA Considerations section of a MIB document, but it does not require that the IANA update any existing registry or create any new registry.

Appendix A: MIB Review Checklist

The purpose of a MIB review is to review the MIB module both for technical correctness and for adherence to IETF documentation requirements. The following checklist may be helpful when reviewing a draft document:

1.) I-D Boilerplate -- verify that the draft contains the required Internet-Draft boilerplate (see http://www.ietf.org/ietf/1id-guidelines.txt), including the appropriate statement to permit publication as an RFC, and that I-D boilerplate does not contain references or section numbers.

2.) Abstract -- verify that the abstract does not contain references, that it does not have a section number, and that its content follows the guidelines in http://www.ietf.org/ietf/1id-guidelines.txt.

3.) MIB Boilerplate -- verify that the draft contains the latest approved SNMP Network Management Framework boilerplate from the OPS area web site (http://www.ops.ietf.org/mib-boilerplate.html).

4.) Security Considerations Section -- verify that the draft uses the latest approved template from the OPS area web site (http://www.ops.ietf.org/mib-security.html) and that the guidelines therein have been followed.

5.) IANA Considerations Section -- this section must always be present. If the draft requires no action from the IANA, ensure that this is explicitly noted. If the draft requires OID values to be assigned, ensure that the IANA Considerations section contains the information specified in Section 3.5 of these guidelines. If the draft contains the initial version of an IANA-maintained module, verify that the MODULE-IDENTITY invocation contains maintenance instructions that comply with the requirements in RFC 2434. In the latter case, the IANA Considerations section that will appear in the RFC MUST contain a pointer to the actual IANA-maintained module.

6.) References -- verify that the references are properly divided between normative and informative references, that RFC 2119 is included as a normative reference if the terminology defined therein is used in the document, that all references required by the boilerplate are present, that all MIB modules containing imported items are cited as normative references, and that all citations point to the most current RFCs unless there is a valid reason to do otherwise (for example, it is OK to include an informative reference to a previous version of a specification to help explain a feature included for backward compatibility).

7.) Copyright Notices -- verify that the draft contains an abbreviated copyright notice in the DESCRIPTION clause of each MODULE-IDENTITY invocation and that it contains the full copyright notice and disclaimer specified in Sections 5.4 and 5.5 of RFC 3978 at the end of the document. Make sure that the correct year is used in all copyright dates.

8.) IPR Notice -- if the draft does not contains a verbatim copy of the IPR notice specified in Section 5 of RFC 3979, recommend that the IPR notice be included.

9.) Other Issues -- check for any issues mentioned in http://www.ietf.org/ID-Checklist.html that are not covered elsewhere.

10.) Technical Content -- review the actual technical content for compliance with the guidelines in this document. The use of a MIB compiler is recommended when checking for syntax errors; see http://www.ops.ietf.org/mib-review-tools.html for more information. Checking for correct syntax, however, is only part of the job. It is just as important to actually read the MIB document from the point of view of a potential implementor. It is particularly important to check that DESCRIPTION clauses are sufficiently clear and unambiguous to allow interoperable implementations to be created.

Appendix B: Commonly Used Textual Conventions

The following TCs are defined in SNMPv2-TC [RFC2579]:

```
DisplayString               OCTET STRING (SIZE (0..255))
PhysAddress                 OCTET STRING
MacAddress                  OCTET STRING (SIZE (6))
TruthValue                  enumerated INTEGER
TestAndIncr                 INTEGER (0..2147483647)
AutonomousType              OBJECT IDENTIFIER
VariablePointer             OBJECT IDENTIFIER
RowPointer                  OBJECT IDENTIFIER
RowStatus                   enumerated INTEGER
TimeStamp                   TimeTicks
TimeInterval                INTEGER (0..2147483647)
DateAndTime                 OCTET STRING (SIZE (8 | 11))
StorageType                 enumerated INTEGER
TDomain                     OBJECT IDENTIFIER
TAddress                    OCTET STRING (SIZE (1..255))
```

Note 1. InstancePointer is obsolete and MUST NOT be used.

Note 2. DisplayString does not support internationalized text. It MUST NOT be used for objects that are required to hold

internationalized text (which is always the case if the object is intended for use by humans [RFC2277]). Designers SHOULD consider using SnmpAdminString, Utf8String, or LongUtf8String for such objects.

Note 3. TDomain and TAddress SHOULD NOT be used in new MIB modules. The TransportDomain, TransportAddressType, and TransportAddress TCs (defined in TRANSPORT-ADDRESS-MIB [RFC3419]) SHOULD be used instead.

The following TC is defined in SNMP-FRAMEWORK-MIB [RFC3411]:

```
SnmpAdminString             OCTET STRING (SIZE (0..255))
```

The following TCs are defined in SYSAPPL-MIB [RFC2287]:

```
Utf8String                  OCTET STRING (SIZE (0..255))
LongUtf8String              OCTET STRING (SIZE (0..1024))
```

The following TCs are defined in INET-ADDRESS-MIB [RFC4001]:

```
InetAddressType             enumerated INTEGER
InetAddress                 OCTET STRING (SIZE (0..255))
InetAddressPrefixLength     Unsigned32 (0..2040)
InetPortNumber              Unsigned32 (0..65535))
InetAutonomousSystemNumber  Unsigned32
InetScopeType               enumerated INTEGER
InetZoneIndex               Unsigned32
InetVersion                 enumerated INTEGER
```

The following TCs are defined in TRANSPORT-ADDRESS-MIB [RFC3419]:

```
TransportDomain             OBJECT IDENTIFIER
TransportAddressType        enumerated INTEGER
TransportAddress            OCTET STRING (SIZE (0..255))
```

The following TC is defined in RMON2-MIB [RFC2021]:

```
ZeroBasedCounter32          Gauge32
```

The following TCs are defined in HCNUM-TC [RFC2856]:

```
ZeroBasedCounter64          Counter64
CounterBasedGauge64         Counter64
```

The following TCs are defined in IF-MIB [RFC2863]:

```
InterfaceIndex              Integer32 (1..2147483647)
```

```
   InterfaceIndexOrZero        Integer32 (0..2147483647)

   The following TCs are defined in ENTITY-MIB [RFC4133]:

   PhysicalIndex               Integer32 (1..2147483647)
   PhysicalIndexOrZero         Integer32 (0..2147483647)

   The following TCs are defined in PerfHist-TC-MIB [RFC3593]:

   PerfCurrentCount            Gauge32
   PerfIntervalCount           Gauge32
   PerfTotalCount              Gauge32

   The following TCs are defined in HC-PerfHist-TC-MIB [RFC3705]:

   HCPerfValidIntervals        Integer32 (0..96)
   HCPerfInvalidIntervals      Integer32 (0..96)
   HCPerfTimeElapsed           Integer32 (0..86399)
   HCPerfIntervalThreshold     Unsigned32 (0..900)
   HCPerfCurrentCount          Counter64
   HCPerfIntervalCount         Counter64
   HCPerfTotalCount            Counter64
```

Appendix C: Suggested Naming Conventions

Authors and reviewers of IETF MIB modules have often found the following naming conventions to be helpful in the past, and authors of new IETF MIB modules are urged to consider following them.

- The module name should be of the form XXX-MIB (or XXX-TC-MIB for a module with TCs only), where XXX is a unique prefix (usually all caps with hyphens for separators) that is not used by any existing module. For example, the module for managing optical interfaces [RFC3591] uses the prefix OPT-IF and has module name OPT-IF-MIB.

- The descriptor associated with the MODULE-IDENTITY invocation should be of the form xxxMIB, xxxMib, or xxxMibModule, where xxx is a mixed-case version of XXX starting with a lowercase letter and without any hyphens. For example, the OPT-IF-MIB uses the prefix optIf, and the descriptor associated with its MODULE-IDENTITY invocation is optIfMibModule.

- Other descriptors within the MIB module should start with the same prefix xxx. OPT-IF-MIB uses the prefix optIf for all descriptors.

- Names of TCs that are specific to the MIB module and names of SEQUENCE types that are used in conceptual table definitions should start with a prefix Xxx that is the same as xxx but with the initial letter changed to uppercase. OPT-IF-MIB uses the prefix OptIf on the names of TCs and SEQUENCE types.

- The descriptor associated with a conceptual table should be of the form xxxZzzTable; the descriptor associated with the corresponding conceptual row should be of the form xxxZzzEntry; the name of the associated SEQUENCE type should be of the form XxxZzzEntry; and the descriptors associated with the subordinate columnar objects should be of the form xxxZzzSomeotherName. An example from the OPT-IF-MIB is the OTMn table. The descriptor of the table object is optIfOTMnTable, the descriptor of the row object is optIfOTMnEntry, the name of the associated SEQUENCE type is OptIfOTMnEntry, and the descriptors of the columnar objects are optIfOTMnOrder, optIfOTMnReduced, optIfOTMnBitRates, optIfOTMnInterfaceType, optIfOTMnTcmMax, and optIfOTMnOpticalReach.

- When abbreviations are used, then they should be used consistently. Inconsistent usage such as

  xxxYyyDestAddr
  xxxZzzDstAddr

  should be avoided.

Appendix D: Suggested OID Layout

As noted in RFC 2578 Section 5.6, it is common practice to use the value of the MODULE-IDENTITY invocation as a subtree under which other OBJECT IDENTIFIER values assigned within the module are defined. That, of course, leaves open the question of how OIDs are assigned within that subtree. One assignment scheme that has gained favor -- and that is RECOMMENDED unless there is a specific reason not use it -- is to have three separate branches immediately below the MODULE-IDENTITY value dedicated (respectively) to notification definitions, object definitions, and conformance definitions, and to further subdivide the conformance branch into separate sub-branches for compliance statements and object/notification groups. This structure is illustrated below, with naming conventions following those outlined in Appendix C. The numbers in parentheses are the sub-identifiers assigned to the branches.

```
xxxMIB
|
+-- xxxNotifications(0)
+-- xxxObjects(1)
+-- xxxConformance(2)
    |
    +-- xxxCompliances(1)
    +-- xxxGroups(2)
```

When using this scheme, notification definition values are assigned immediately below the xxxNotifications node. This ensures that each OID assigned to a notification definition has a next-to-last sub-identifier of zero, which is REQUIRED by Section 4.7 above. The other sub-branches may have additional sub-structure, but none beyond that specified in Section 4.6.5 above is actually required.

One example of a MIB module whose OID assignments follow this scheme is the POWER-ETHERNET-MIB [RFC3621].

Normative References

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

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

[RFC2580] McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M., and S. Waldbusser, "Conformance Statements for SMIv2", STD 58, RFC 2580, April 1999.

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

[RFC2863] McCloghrie, K. and F. Kastenholz, "The Interfaces Group MIB", RFC 2863, June 2000.

[RFC2864] McCloghrie, K. and G. Hanson, "The Inverted Stack Table Extension to the Interfaces Group MIB", RFC 2864, June 2000.

[RFC2434] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 2434, October 1998.

[RFC3978] Bradner, S., "IETF Rights in Contributions", BCP 78, RFC 3978, March 2005.

[RFC3979] Bradner, S., "Intellectual Property Rights in IETF Technology", BCP 79, RFC 3979, March 2005.

[RFC4001] Daniele, M., Haberman, B., Routhier, S., and J. Schoenwaelder, "Textual Conventions for Internet Network Addresses", RFC 4001, February 2005.

[RFC3593] Tesink, K., "Textual Conventions for MIB Modules Using Performance History Based on 15 Minute Intervals", RFC 3593, September 2003.

[RFC3705] Ray, B. and R. Abbi, "High Capacity Textual Conventions for MIB Modules Using Performance History Based on 15 Minute Intervals", RFC 3705, February 2004.

[RFC2021] Waldbusser, S., "Remote Network Monitoring Management Information Base Version 2 using SMIv2", RFC 2021, January 1997.

[RFC2856] Bierman, A., McCloghrie, K., and R. Presuhn, "Textual Conventions for Additional High Capacity Data Types", RFC 2856, June 2000.

[RFC3411] Harrington, D., Presuhn, R., and B. Wijnen, "An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks", STD 62, RFC 3411, December 2002.

[RFC2287] Krupczak, C. and J. Saperia, "Definitions of System-Level Managed Objects for Applications", RFC 2287, February 1998.

[RFC3418] Presuhn, R., Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Management Information Base (MIB) for the Simple Network Management Protocol (SNMP)", STD 62, RFC 3418, December 2002.

[RFC3416] Presuhn, R., Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Protocol Operations for the Simple Network Management Protocol (SNMP)", STD 62, RFC 3416, December 2002.

[RFC4133] Bierman, A. and K. McCloghrie, "Entity MIB (Version 3)", RFC 4133, August 2005.

[RFC2277] Alvestrand, H., "IETF Policy on Character Sets and Languages", BCP 18, RFC 2277, January 1998.

[RFC3419] Daniele, M. and J. Schoenwaelder, "Textual Conventions for Transport Addresses", RFC 3419, December 2002.

Informative References

[RFC1155] Rose, M. and K. McCloghrie, "Structure and Identification of Management Information for TCP/IP-based Internets", STD 16, RFC 1155, May 1990.

[RFC1212] Rose, M. and K. McCloghrie, "Concise MIB Definitions", STD 16, RFC 1212, March 1991.

[RFC1215] Rose, M., "A Convention for Defining Traps for use with the SNMP", RFC 1215, March 1991.

[RFC2223bis] Reynolds, J. and R. Braden, "Instructions to Request for Comments (RFC) Authors", Work in Progress, August 2004.

[RFC3410] Case, J., Mundy, R., Partain, D., and B. Stewart, "Introduction and Applicability Statements for Internet-Standard Management Framework", RFC 3410, December 2002.

[RFC2932] McCloghrie, K., Farinacci, D., and D. Thaler, "IPv4 Multicast Routing MIB", RFC 2932, October 2000.

[RFC1573] McCloghrie, K. and F. Kastenholz, "Evolution of the Interfaces Group of MIB-II", RFC 1573, January 1994.

[RFC3621] Berger, A. and D. Romascanu, "Power Ethernet MIB", RFC 3621, December 2003.

[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.

[RFC2108] de Graaf, K., Romascanu, D., McMaster, D., and K. McCloghrie, "Definitions of Managed Objects for IEEE 802.3 Repeater Devices using SMIv2", RFC 2108, February 1997.

[RFC3289] Baker, F., Chan, K., and A. Smith, "Management Information Base for the Differentiated Services Architecture", RFC 3289, May 2002.

[RFC1213] McCloghrie, K. and M. Rose, "Management Information Base for Network Management of TCP/IP-based internets - MIB-II", STD 17, RFC 1213, March 1991.

[RFC1595] Brown, T. and K. Tesink, "Definitions of Managed Objects for the SONET/SDH Interface Type", RFC 1595, March 1994.

[RFC2558] Tesink, K., "Definitions of Managed Objects for the SONET/SDH Interface Type", RFC 2558, March 1999.

[RFC3591] Lam, H-K., Stewart, M., and A. Huynh, "Definitions of Managed Objects for the Optical Interface Type", RFC 3591, September 2003.

Editor's Address

C. M. Heard
158 South Madison Ave. #207
Pasadena, CA 91101-2569
USA

Phone: +1 626 795 5311
EMail: heard@pobox.com

Full Copyright Statement

Intellectual Property

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.