SMI

STRUCTURE OF MANAGEMENT INFORMATION

RFC 1155: SMIv1

RFC 1212: CONCISE MIB DEFINITIONS

RFC 2578: SMIv2

RFC 2579: TEXTUAL CONVENTIONS

MAKES THE DEFINITION OF (NEW) MIBs EASIER
MANAGEMENT INFORMATION WITHIN MANAGED SYSTEMS MUST BE REPRESENTED AS:

- SCALARS
- TABLES

(= TWO DIMENSIONAL ARRAYS OF SCALARS)

THE SNMP PROTOCOL CAN ONLY EXCHANGE (A LIST OF) SCALARS

DEFINED IN TERMS OF ASN.1 CONSTRUCTS
SMI: DATA TYPES FOR SCALARS

<table>
<thead>
<tr>
<th>SIMPLE TYPES:</th>
<th>SMIv1</th>
<th>SMIv2</th>
</tr>
</thead>
<tbody>
<tr>
<td>INTEGER</td>
<td></td>
<td>INTEGER</td>
</tr>
<tr>
<td>OCTET STRING</td>
<td></td>
<td>OCTET STRING</td>
</tr>
<tr>
<td>OBJECT IDENTIFIER</td>
<td></td>
<td>OBJECT IDENTIFIER</td>
</tr>
<tr>
<td></td>
<td>-</td>
<td>Integer32</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>APPLICATION-WIDE TYPES:</th>
<th>SMIv1</th>
<th>SMIv2</th>
</tr>
</thead>
<tbody>
<tr>
<td>-</td>
<td></td>
<td>Unsigned32</td>
</tr>
<tr>
<td>-</td>
<td></td>
<td>Gauge32</td>
</tr>
<tr>
<td>Gauge</td>
<td></td>
<td>Counter32</td>
</tr>
<tr>
<td>Counter</td>
<td></td>
<td>Counter64</td>
</tr>
<tr>
<td>-</td>
<td></td>
<td>TimeTicks</td>
</tr>
<tr>
<td>TimeTicks</td>
<td></td>
<td>TimeTicks</td>
</tr>
<tr>
<td>IpAddress</td>
<td></td>
<td>IpAddress</td>
</tr>
<tr>
<td>Opaque</td>
<td></td>
<td>Opaque</td>
</tr>
<tr>
<td>NetworkAddress</td>
<td></td>
<td>-</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PSEUDO TYPES:</th>
<th>SMIv1</th>
<th>SMIv2</th>
</tr>
</thead>
<tbody>
<tr>
<td>-</td>
<td></td>
<td>BITS</td>
</tr>
</tbody>
</table>
EXAMPLE OF SCALAR OBJECTS

MANAGER

SNMP

AGENT

MANAGED OBJECT INSTANCES

name

address

uptime
OBJECT NAMING

INTRODUCE NAMING TREE

NEW-MIB:

1

address (1) info (2)

130.89.16.2

name (1) uptime (2)

printer-1 123456

THE LEAVES OF THE TREE REPRESENT THE MANAGED OBJECTS

NODES ARE INTRODUCED FOR NAMING PURPOSES
OBJECT NAMING

• address
  Object ID = 1.1
  Object Instance = 1.1.0
  Value of Instance = 130.89.16.2

• info
  Object ID = 1.2

• name
  Object ID = 1.2.1
  Object Instance = 1.2.1.0
  Value of Instance = printer-1

• uptime
  Object ID = 1.2.2
  Object Instance = 1.2.2.0
  Value of Instance = 123456

ALTERNATIVE:
Object ID = NEW-MIB info uptime
OBJECT NAMING: MIBs

- root
  - ccitt (0)
  - iso (1)
    - stnd (0)
    - reg-auth (1)
    - mb (2)
  - joint-iso-ccitt (2)
  - org (3)
    - dod (6)
      - internet (1)
        - directory (1)
        - mngt (2)
        - experimental (3)
        - private (4)
        - security (5)
        - snmpV2 (6)
          - snmpDomains (1)
          - snmpProxys (2)
          - snmpModules (3)
          - enterprises (1)
<table>
<thead>
<tr>
<th>OBJECT-TYPE:</th>
<th>INTEGER</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>OCTET STRING</td>
</tr>
<tr>
<td></td>
<td>OBJECT IDENTIFIER</td>
</tr>
<tr>
<td></td>
<td>BITS</td>
</tr>
<tr>
<td>IpAddress</td>
<td>read-only</td>
</tr>
<tr>
<td>Integer32</td>
<td>read-write</td>
</tr>
<tr>
<td>Counter32</td>
<td>read-create</td>
</tr>
<tr>
<td>Counter64</td>
<td>accessible-for-notify</td>
</tr>
<tr>
<td>Gauge32</td>
<td>not-accessible</td>
</tr>
<tr>
<td>TimeTicks</td>
<td>current</td>
</tr>
<tr>
<td>Opaque</td>
<td>deprecated</td>
</tr>
<tr>
<td>New Type</td>
<td>obsolete</td>
</tr>
</tbody>
</table>

### MAX-ACCESS

- read-only
- read-write
- read-create
- accessible-for-notify
- not-accessible

### STATUS

- current
- deprecated
- obsolete

### DESCRIPTION

""
OBJECT TYPE DEFINITION - EXAMPLE

-- Definition of address

address OBJECT-TYPE
SYNTAX   IpAddress
MAX-ACCESS read-write
STATUS   current
DESCRIPTION "The Internet address of this system"
 ::= {NEW-MIB  1}
DEFINITION OF NON-LEAF 'OBJECTS'

Name OBJECT IDENTIFIER ::= {...}

EXAMPLE:
info OBJECT IDENTIFIER ::= {NEW-MIB 2}

ALTERNATIVE CONSTRUCT: OBJECT IDENTITY

EXAMPLE:
info OBJECT-IDENTITY
STATUS current
DESCRIPTION "The node under which future scalar objects should be registered"
::= {NEW-MIB 2}
DEFINITION OF A MIB

NEW-MIB DEFINITIONS ::= BEGIN

import statement(s)
module identity definition

definition of all node and leaf objects

definition of implementation requirements

END
newMibModule MODULE-IDENTITY
LAST-UPDATED "200104041200Z"
ORGANIZATION "UT-ARCH"
CONTACT-INFO 
  EWI-ARCH Group
  University of Twente
  POBox 217
  7500 AE Enschede
  The Netherlands
  Email: simpleweb@simpleweb.org 

DESCRIPTION
  "Experimental MIB for demo purposes"
 ::= { enterprises ut(785) 7 }
IMPORT STATEMENT - EXAMPLE

IMPORTS

MODULE-IDENTITY, OBJECT-TYPE, TimeTicks, enterprises

FROM SNMPv2-SMI;
EXAMPLE: ROUTING TABLE

<table>
<thead>
<tr>
<th>destination</th>
<th>next</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>7</td>
<td>2</td>
</tr>
<tr>
<td>8</td>
<td>3</td>
</tr>
<tr>
<td>9</td>
<td>3</td>
</tr>
</tbody>
</table>

TO RETRIEVE INDIVIDUAL TABLE ENTRIES
EACH ENTRY SHOULD GET AN IDENTIFIER
POSSIBILITY 1 (NOT BEING USED BY SNMP): USE ROW NUMBERS

NEW-MIB:

address (1)
130.89.16.2

info (2)

name (1)
printer-1

uptime (2)
123456

routeTable (3)

dest(1)

next(2)

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>7</td>
<td>2</td>
</tr>
<tr>
<td>8</td>
<td>3</td>
</tr>
<tr>
<td>9</td>
<td>3</td>
</tr>
</tbody>
</table>

EXAMPLE: THE VALUE OF NEW-MIB routeTable next 5 IS 3
NAMING OF TABLE ENTRIES - II

POSSIBILITY 2 (USED BY SNMP): INTRODUCE AN INDEX COLUMN

NEW-MIB:

1

- address (1) 130.89.16.2
- info (2)
  - name (1) printer-1
  - uptime (2) 123456
- routeTable (3)
  - dest(1)
  - next(2)
    |   |   |
---|---|---|
  2 | 2 |
  3 | 3 |
  5 | 2 |
  7 | 2 |
  8 | 3 |
  9 | 3 |

EXAMPLE: THE VALUE OF NEW-MIB routeTable next 5 IS 2
TABLE INDEXING

GENERAL SCHEME

X.C.I

OID of Table  Column number  Index value

EXAMPLES:

OID of Table = 1.3
1.3.1.5 => 5
1.3.2.5 => 2
1.3.1.9 => 9
1.3.2.9 => 3
1.3.2.7 => 2
1.3.1.1 => entry does not exist
1.3.2.1 => entry does not exist
TABLE INDEXING - NON-INTEGER INDEX

AN INDEX NEED NOT BE AN INTEGER

routeTable (3)

dest (1)  next (2)

| 130.89.16.1  | 130.89.16.1 |
| 130.89.16.4  | 130.89.16.4 |
| 130.89.16.23 | 130.89.16.1 |
| 130.89.19.121| 130.89.16.1 |
| 192.1.23.24  | 130.89.16.4 |
| 193.22.11.97 | 130.89.16.4 |

EXAMPLES:
OID of Table = 1.3
1.3.1.130.89.16.23 => 130.89.16.23
1.3.2.130.89.16.23 => 130.89.16.1
1.3.1.193.22.11.97 => 193.22.11.97
1.3.2.193.22.11.97 => 130.89.16.4
1.3.2.130.89.19.121 => 130.89.16.1
TABLE INDEXING - MULTIPLE INDEX FIELDS

USE OF MULTIPLE INDEX FIELDS

X.C.I1.I2

OID of Table   Column number   Index value 1   Index value 2
### TABLE INDEXING - MULTIPLE INDEX FIELDS: EXAMPLE

#### EXAMPLE:

```
routeTable (3)

<table>
<thead>
<tr>
<th>dest (1)</th>
<th>policy (2)</th>
<th>next (3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>130.89.16.23</td>
<td>1</td>
<td>130.89.16.23</td>
</tr>
<tr>
<td>130.89.16.23</td>
<td>2</td>
<td>130.89.16.23</td>
</tr>
<tr>
<td>130.89.19.121</td>
<td>1</td>
<td>130.89.16.1</td>
</tr>
<tr>
<td>192.1.23.24</td>
<td>1</td>
<td>130.89.16.1</td>
</tr>
<tr>
<td>192.1.23.24</td>
<td>2</td>
<td>130.89.16.4</td>
</tr>
<tr>
<td>193.22.11.97</td>
<td>1</td>
<td>130.89.16.1</td>
</tr>
</tbody>
</table>
```

1.3.3.192.1.23.24.1 $\Rightarrow$ 130.89.16.1

1.3.3.192.1.23.24.2 $\Rightarrow$ 130.89.16.4

1 = low costs
2 = high reliability
TABLE DEFINITION

-- Definition of the route table

routeTable OBJECT-TYPE
SYNTAX SEQUENCE OF RouteEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION "This entity’s routing table"
 ::= {NEW-MIB  3}

routeEntry OBJECT-TYPE
SYNTAX RouteEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION "A route to a particular destination"
INDEX {dest, policy}
 ::= {routeTable  1}
ROUTE DEFINITION (cont. 1)

RouteEntry ::= SEQUENCE{
  dest ipAddress,  
policy INTEGER,  
next ipAddress  
}
<table>
<thead>
<tr>
<th>this is the table</th>
</tr>
</thead>
<tbody>
<tr>
<td>routeTable</td>
</tr>
<tr>
<td>OBJECT-TYPE</td>
</tr>
<tr>
<td>SYNTAX</td>
</tr>
<tr>
<td>SEQUENCE OF</td>
</tr>
<tr>
<td>RouteEntry</td>
</tr>
<tr>
<td>...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>this is a row</th>
</tr>
</thead>
<tbody>
<tr>
<td>routeEntry</td>
</tr>
<tr>
<td>OBJECT-TYPE</td>
</tr>
<tr>
<td>SYNTAX</td>
</tr>
<tr>
<td>RouteEntry</td>
</tr>
<tr>
<td>...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>this is a new type</th>
</tr>
</thead>
<tbody>
<tr>
<td>RouteEntry::=</td>
</tr>
<tr>
<td>SEQUENCE</td>
</tr>
<tr>
<td>...</td>
</tr>
</tbody>
</table>
TABLE DEFINITION (cont. 3)

dest OBJECT-TYPE
SYNTAX  ipAddress
ACCESS  not-accesible
STATUS  current
DESCRIPTION "The address of a particular destination"
 ::= {routeEntry  1}

policy OBJECT-TYPE
SYNTAX  INTEGER {
            costs(1)  -- lowest delay
            reliability(2) }  -- highest reliability
ACCESS  not-accesible
STATUS  current
DESCRIPTION "The routing policy to reach that destination"
 ::= {routeEntry  2}

next OBJECT-TYPE
SYNTAX  ipAddress
ACCESS  read-write
STATUS  current
DESCRIPTION "The internet address of the next hop"
 ::= {routeEntry  3}
DEFINITION OF NEW TYPES

TEXTUAL CONVENTIONS

TO REFINE SEMANTICS OF EXISTING TYPES

EXAMPLE:

RunState ::= TEXTUAL CONVENTION
  STATUS current
  DESCRIPTION "..."
  SYNTAX INTEGER{
    running(1)
    runnable(2)
    waiting(3)
    exiting(4)
  }
TEXTUAL CONVENTIONS

- PhysAddress
- MacAddress
- TruthValue
- AutonomousType
- InstancePointer
- VariablePointer
  - RowPointer
  - RowStatus
  - TimeStamp
  - TimeInterval
- DateAndTime
- StorageType
  - TDomain
  - TAddress

- Inet-Address...
### ROW-STATUS TEXTUAL CONVENTION

Used to change table rows

<table>
<thead>
<tr>
<th>TO:</th>
<th>VIA:</th>
<th>STATUS:</th>
</tr>
</thead>
<tbody>
<tr>
<td>130.89.16.4</td>
<td>130.89.1.1</td>
<td>ACTIVE</td>
</tr>
<tr>
<td>130.89.17.6</td>
<td>130.89.1.1</td>
<td>NOT READY</td>
</tr>
<tr>
<td>130.89.18.2</td>
<td>130.89.1.4</td>
<td>ACTIVE</td>
</tr>
<tr>
<td>130.89.18.7</td>
<td>130.89.1.4</td>
<td>ACTIVE</td>
</tr>
</tbody>
</table>
ROW-STATUS - STATE DIAGRAM

1. set status column to createAndGo
2. set status column to createAndWait
3. set status column to active
4. set status column to notInService
5. set status column to destroy
6. set any other column to some value
NOTIFICATION TYPES

SMIv2:
• MIBs MAY NOW INCLUDE NOTIFICATION TYPE MACROS

EXAMPLE:
linkUp NOTIFICATION-TYPE
  OBJECTS   {ifIndex}
  STATUS    current
  DESCRIPTION
            "A linkUp trap signifies that the entity has detected that the ifOperStatus object has changed to Up"
::= {snmpTraps 4}
DEFINITION OF IMPLEMENTATION REQUIREMENTS

THE MODULE-COMPLIANCE CONSTRUCT DEFINES IMPLEMENTATION REQUIREMENTS FOR AGENTS

newMibCompliance MODULE-COMPLIANCE
STATUS ...
DESCRIPTION ...

MODULE 1

MODULE n

::= { ... }
OBJECT GROUP CONSTRUCT

TO DEFINE A SET OF RELATED OBJECT TYPES

EXAMPLE:

newMibScalarGroup OBJECT-GROUP
  OBJECTS { address, name, uptime }
  STATUS current
  DESCRIPTION "The collection of scalar objects."
 ::= { demoGroups 1 }