Tutorial Exercises on the Structure of Management Information (SMI)

This exercise includes a number of questions in section 3 on page 7. You are required to write your answer to each of these questions and submit them next week.

1 Background

The SNMP protocol is called “simple” because the protocol itself is quite simple. However, the difficulty is in applying it to actually managing systems and networks.

There are many terms and standards involved; it is necessary to understand enough of them to make sense of the MIBs that define the objects that you want to monitor and manage. If you can make sense of the MIB files, you can identify the objects that you want to monitor.

1.1 Management Information Base (MIB)

The MIBs define the objects that you can manage.

When you installed the Net SNMP software package, you installed some MIB files into the directory /usr/share/snmp/mibs/. You can list them all with:

$ rpm -ql ucd-snmp | grep snmp/mibs/*.txt

There are many other MIBs that are not included here; you can download others from somewhere such as [http://www.simpleweb.org/ietf/](http://www.simpleweb.org/ietf/) and include them into your Net SNMP clients as explained at [http://net-snmp.sourceforge.net/FAQ.html#How_do_I_add_a_MIB](http://net-snmp.sourceforge.net/FAQ.html#How_do_I_add_a_MIB) and at [http://net-snmp.sourceforge.net/tutorial/commands/mib-options.html](http://net-snmp.sourceforge.net/tutorial/commands/mib-options.html).

1.2 Management Database (MDB)

The MIBs define what actual information the MDB may contain. The management database is a real database, and holds the actual data, whose format is defined by the MIB, stored in the agent or manager. It contains the measured or administratively configured values of the elements of the network.

1.3 Structure of Management Information

SMI is a definition of the structure of the MIBs, how they are connected together into a tree, as shown in figure 1 on the next page. See the RFCs below in section 3.2 on page 8. It specifies which part of ASN.1 will be used to define MIBs.
Abstract Syntax Notation One (ASN.1)

ASN.1 is widely used for many things other than SNMP. See [http://asn1.elibel.tm.fr/en/uses/](http://asn1.elibel.tm.fr/en/uses/) for a list of some of the applications of ASN.1. There is a web site dedicated to providing information about it at [http://asn1.elibel.tm.fr/](http://asn1.elibel.tm.fr/).

Basic Encoding Rules (BER)

The basic encoding rules is an ISO standard. It describes a method for encoding values of each ASN.1 type as a string of octets.

1.5.1 ASN.1 Keywords used in SNMP

Table 1 on page 9 lists some frequently used ASN.1 keywords.

1.5.2 ASN.1 Symbols and Operators

Table 2 on page 9 lists the ASN.1 symbols.

1.5.3 ASN.1 Data Types used in SNMP

There are three “base types” of data defined in ASN.1 used in SMI: INTEGER, OCTET STRING, and OBJECT IDENTIFIER.
1.6 Syntax of a Managed Object Definition

Every object definition in SMI has the format:

```plaintext
name OBJECT-TYPE
    SYNTAX datatype
    ACCESS either read-only, read-write, write-only, or not-accessible
    DESCRIPTION "Some text that describes this managed object."
    ::= { unique object ID that defines this object }
```

We will refer to this later in our activities.

2 The MIB-II Definition


```plaintext
RFC1213-MIB DEFINITIONS ::= BEGIN

IMPORTS
    mgmt, NetworkAddress, IpAddress, Counter, Gauge,
        TimeTicks
    FROM RFC1155-SMI

OBJECT-TYPE
    FROM RFC-1212;
```

The first line defines the name of the MIB, here RFC1213-MIB. The format of this definition is always the same.

The IMPORTS section of the MIB is sometimes called the linkage section. It lets you import definitions of datatypes and OIDs from other MIBs. Here we get the definition of:

- mgmt
- NetworkAddress
- IpAddress
- Counter
- Gauge
- TimeTicks

from RFC1155-SMI, the MIB from the RFC that defines SMIV1.

It also imports OBJECT-TYPE from RFC-1212, the Concise MIB Definition, which defines how MIB files are written.

```plaintext
mib-2 OBJECT IDENTIFIER ::= { mgmt 1 }
```

The line above says that the OID of mib-2 is 1.3.6.1.2.1. RFC1155-SMI defines mgmt as the OID 1.3.6.1.2.
-- groups in MIB-II

system OBJECT IDENTIFIER ::= { mib-2 1 }
interfaces OBJECT IDENTIFIER ::= { mib-2 2 }
at OBJECT IDENTIFIER ::= { mib-2 3 }
ip OBJECT IDENTIFIER ::= { mib-2 4 }
tcp OBJECT IDENTIFIER ::= { mib-2 5 }
udp OBJECT IDENTIFIER ::= { mib-2 6 }
egp OBJECT IDENTIFIER ::= { mib-2 7 }
transmission OBJECT IDENTIFIER ::= { mib-2 10 }
snmp OBJECT IDENTIFIER ::= { mib-2 11 }

So here the system group is defines as the OID 1.3.6.1.2.1.1, and so on. A comment is a line starting with --.

-- the Interfaces table

-- Implementation of the Interfaces group is mandatory for
-- all systems.

ifNumber OBJECT-TYPE
SYNTAX INTEGER
ACCESS read-only
STATUS mandatory
DESCRIPTION
"The number of network interfaces (regardless of
their current state) present on this system."
::= { interfaces 1 }

The ifNumber above tells how many entries there are in the table.

-- The Interfaces table contains information on the entity’s
-- interfaces. Each interface is thought of as being
-- attached to a ‘subnetwork’. Note that this term should
-- not be confused with ‘subnet’ which refers to an
-- addressing partitioning scheme used in the Internet suite
-- of protocols.

ifTable OBJECT-TYPE
SYNTAX SEQUENCE OF IfEntry
ACCESS not-accessible
STATUS mandatory
DESCRIPTION
"A list of interface entries. The number of
entries is given by the value of ifNumber."
::= { interfaces 2 }

This is the first managed object shown here. ifTable represents a table of network interfaces on a managed device. Notice that object names are defined with mixed case, the first letter is lowercase.
Notice that this follows the format of an OBJECT-TYPE in section 1.6 on page 3.

The SYNTAX of ifTable is SEQUENCE OF IfEntry. The object is not-accessible, which means that you cannot query the agent for the value of this object. It has a STATUS of mandatory, which means that if an agent complies with the MIB-II specification, then it must implement this object. The DESCRIPTION tells you what this object is. The unique OID is 1.3.6.1.2.1.2.2, or iso.org.dod.internet.mgmt.interfaces.2.

Next, let’s look at the SEQUENCE definition, which is used with the SEQUENCE OF type in the ifTable definition.

IfEntry ::= SEQUENCE {
    ifIndex INTEGER,
    ifDescr DisplayString,
    ifType INTEGER,
    ifMtu INTEGER,
    ifSpeed Gauge,
    ifPhysAddress PhysAddress,
    ifAdminStatus INTEGER,
    ifOperStatus INTEGER,
    ifLastChange TimeTicks,
    ifInOctets Counter,
    ifInUcastPkts Counter,
    ifInNUcastPkts Counter,
    ifInDiscards Counter,
    ifInErrors Counter,
    ifInUnknownProtos Counter,
    ifOutOctets Counter,
    ifOutUcastPkts Counter,
    ifOutNUcastPkts Counter,
    ifOutDiscards Counter,
    ifOutErrors
Counter,
ifOutQLen
Gauge,
ifSpecific

OBJECT IDENTIFIER
}

The name of the SEQUENCE (IfEntry) is mixed-case, but the first letter is capitalised, which is different from the object definition for ifTable. A SEQUENCE is a list of objects that go into one row of a table. After this, we must have OBJECT-TYPE definitions that define each of these variables. A table can have any number of rows. The agent manages the number of rows. An NMS can also add rows to a table using a set operation.

IfEntry is the data type; rather like a struct definition in the C language.

Let’s look at ifEntry, the definition of what we find in the table, the actual rows of the table themselves. It looks almost the same as the definition for ifTable, except that it has a new clause, INDEX. The index is a unique value that identifies a single row in the table, like an array index. A table is rather like an array of structs in C. The agent assigns these index values. If a router has eight interfaces, then ifTable will contain eight rows.

ifEntry OBJECT-TYPE
SYNTAX IfEntry
ACCESS not-accessible
STATUS mandatory
DESCRIPTION
"An interface entry containing objects at the subnetwork layer and below for a particular interface."
INDEX { ifIndex }
::= { ifTable 1 }

Here we now look at the definition for ifIndex, the first item in IfEntry. Notice that indexes start from 1.

ifIndex OBJECT-TYPE
SYNTAX INTEGER
ACCESS read-only
STATUS mandatory
DESCRIPTION
"A unique value for each interface. Its value ranges between 1 and the value of ifNumber. The value for each interface must remain constant at least from one re-initialization of the entity’s network management system to the next re-initialization."
::= { ifEntry 1 }

This object is read-only, which means that you can see the value, but not change it.

Here is the last object we look at from this table:

ifDescr OBJECT-TYPE
SYNTAX DisplayString (SIZE (0..255))
ACCESS read-only
STATUS mandatory
DESCRIPTION
"A textual string containing information about the interface. This string should include the name of the manufacturer, the product name and the version of the hardware interface."
::= { ifEntry 2 }

END

ifDescr is just a textual description of the interface.

The MIB definition finishes with END.

3 Questions

Write your answer to these questions on a piece of paper. Hand the paper in next week. These questions are designed to help you become familiar with ASN.1 and MIB files. I expect you to refer to the MIB file itself to work out the answers.

1. Draw the tree representing the interfaces group. To make the diagram simpler, only include in the table the two entries shown: ifIndex and ifDescr.

2. What is the difference between the syntax “SEQUENCE” and the syntax “SEQUENCE OF”?

3. State in one short sentence the difference between the Management Information Base (MIB) and the management database (MDB).

4. Use the information in section 1.1 on page 1 to download a copy of the Printer-MIB.

5. How many tables are there in the Printer-MIB? (Hint: it should take less than two minutes to get an accurate answer.)

6. The pagecount of pages printed by a printer is given by prtMarkerLifeCount.1.1, an entry in a table
   
   (a) What is the name of the table that this is part of?

   (b) What is the OID of the managed object that gives the pagecount?

7. Write an ASN.1 module that defines DaysOfWeek as a SEQUENCE type with each day of the week as the type VisibleString. Define the type VisibleString in any appropriate way. Assign OIDs to each object using the ICT departmental enterprise number.

3.1 Additional Practical Exercises

1. Add the Printer-MIB to your SNMP clients. Refer to the documentation for adding a MIB to your clients; see section 1.1 on page 1.
2. Determine the IP address of the printer in the laboratory, and also determine its community string.

3. Determine the number of PrtMarkerEntries in this printer’s MIB. Write the command that you used.

4. Use SNMP to determine the number of pages that this printer has printed so far. Write the command that you used, and the result.

5. Determine the guaranteed printable area of the printer on an A4 page, using SNMP. Write the query and the result.

3.2 Where can I get the standards documents from?

Keyword | Brief Description
--- | ---
BEGIN | Start of an ASN.1 module
CHOICE | List of alternatives; used in defining SMIV1 and SMIV2 (RFC1155-SMI and SNMPv2-SMI) to define classes of datatypes (SimpleSyntax and ApplicationSyntax), and in SMIV2.
DEFINITIONS | Definition of a data type or managed object
END | End of an ASN.1 module
EXPORTS | Data types that can be exported to other modules
IDENTIFIER | A sequence of non-negative numbers
IMPORTS | Data types defined in external modules that are used in this module
INTEGER | A 32-bit integer (i.e., in the range \(-2^{31}\) to \(2^{31} - 1\)).
MACRO | Required for defining macros, such as the OBJECT-TYPE macro defined in RFC1155-SMI
OBJECT IDENTIFIER | Used to uniquely identify an object with an OID
OCTET | An eight-bit binary value, used with STRING
OCTET STRING | A string of bytes
OF | Used with SEQUENCE
SEQUENCE | An ordered list of data, somewhat like a struct in the C language, usually used to represent a row in a table
SEQUENCE OF | A table of data. Somewhat like an array of struct in C
STRING | Used with OCTET for strings of binary bytes
TYPE NOTATION | used in MACRO definitions to define the syntax of the new types
VALUE NOTATION | used in MACRO definitions to define the syntax of the new values

Table 1: ASN.1 Keywords.

Symbol | Meaning
--- | ---
::= | “defined as”, or assignment
| | or, alternatives, options of a list
- | signed number
-- | introduces a comment
{ } | start and end of a list
[ ] | start and end of a tag
( ) | start and end of a subtype
.. | range

Table 2: The ASN.1 symbols.