Overview

A NAFEM Data Protocol Management Information Base (MIB) module contains six parts each of which are covered in the sections following:

- Name
- Imports
- Identity
- Type Assignments
- Definitions
- End

Section One: Name

The MIB naming convention contains three distinct parts (see figure one below). The name is written in all capital letters\(^1\) and each of the three parts of the MIB name is separated by a hyphen (-).

By convention, the first part of the MIB name is the organization, which in this case is NAFEM. This is to differentiate it from similarly named MIBs from other organizations.

The second part of the MIB name is the subject of the module. The NAFEM Data Protocol contains thirteen separate MIBs (administration, asset management, bulk transfer, clock/calendar, inventory management, maintenance, monitor, notify, security, traps, utility, nafem textual convention and nafem registration).

The final part of the MIB name is the suffix: -MIB. This section also defines the beginning of the definitions section of the MIB.

\(^1\) ASN.1 (Specification of Basic Encoding Rules for Abstract) specifies that only the first letter be capitalized.
Section Two: Imports

The imports section is used to specify references of already existing MIBs. In nearly every MIB, there are portions that are defined in already existing MIB standards. By referencing already existing MIBs, it is not necessary to include the elements of the pre-existing MIB.

<table>
<thead>
<tr>
<th>IMPORTS</th>
<th>FROM RFC-1212</th>
<th>FROM RFC-1213-MIB</th>
<th>FROM NAFEM-TEXTUAL-CONVENTION-MIB</th>
<th>FROM NAFEM-REGISTRATION-MIB</th>
</tr>
</thead>
<tbody>
<tr>
<td>OBJECT-TYPE</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Display String</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NafemDateTime</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SecurityGroup</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure Two – Imports

In figure two, the definition of the macro OBJECT-TYPE is specified in RFC-1212 (1); the type definition for DisplayString is specified in RFC-1213-MIB (2); the NafemDateTime definition is found in NAFEM-TEXTUAL-CONVENTION-MIB (3); the SecurityGroup identifier and associated OID is found in NAFEM-REGISTRATION-MIB (4); and the last import is followed by a semi-colon (5).
Section Three: Identity

The information in this section is actually taken from (Structure and identification of Management Information for TCP/IP-based internets version 2 (SMIv2). It is included here because it contains useful information and permits future transition to SMIv2.

-- nafemSecurityModule MODULE-IDENTITY
--  LAST-UPDATED " 200301010000000 "
--  ORGANIZATION "NAFEM"
--  CONTACT-INFO
--      " Contact: NAFEM Technical Liaison Committee
--
-- Postal: North American Association of Food Equipment Manufacturers
-- (NAFEM)
-- 161 North Clark Street, Suite 2020
-- Chicago, IL 60611
-- Phone: +1 312.821.0201
-- Fax: +1 312.821.2020
-- Web: http://www.nafem.org
--
-- DESCRIPTION
-- " The Security Object Group module.
--
-- Copyright © 2003 North American Association of Food Equipment Manufacturers
-- (NAFEM). All rights reserved. Reproduction of this document is authorized on the
-- condition that the foregoing copyright notice is included.
--
-- The SNMP MIB module (Specification) embodies NAFEM’s proprietary intellectual
-- property. NAFEM retains all title and ownership in the Specification, including any
-- revisions.
--
-- NAFEM grants all interested parties a non-exclusive license to use and distribute an
-- unmodified copy of this Specification in connection with management of products
-- that incorporate the NAFEM Communications Protocol, and without fee, this copyright
-- notice and license appear on all copies.
--
-- This Specification is supplied ‘AS IS’ and NAFEM makes no warranty, either
-- expressed or implied, as to the use, operation, condition, or performance of the
-- Specification. “
--
-- REVISION " 2.0 "
-- DESCRIPTION " updated version of the MIB module NAFEM-SECURITY-MIB "
--
-- ::= { nafemMIBObjects 2 }

Since the NAFEM protocol uses SMIv1, this definition for DisplayString consists only of the first line. The remaining commented lines are the SMIv2 definition of DisplayString. The SMIv2 definition was again included for the additional information it provides and is present for the upgrade to SMIv2.
Section Four: Type Assignments

Type assignments follow the Imports Section (SMIv1) or Identity Section (SMIv2) and are usually textual conventions or macros that simplifies other definitions (see figure three below). Since NAFEM uses SMIv1, the definition for DisplayString consists only of the first line. The remaining lines are the SMIv2 definition of DisplayString. These lines are included for the additional information it provides and is present for an upgrade to SMIv2.

```
DisplayString ::=OCTET STRING (SIZE (0..255))
-- SMIv2 syntax follows
-- DisplayString ::= TEXTUAL-CONVENTION
-- DISPLAY-HINT " 255a "
-- STATUS current
-- DESCRIPTION " Values must conform to the NVE ASCII character set which consists of the 7-bit ASCII displayable characters and a few control characters as defined in RFC 854. "
-- SYNTAX OCTET STRING (SIZE (0..255))
```

Figure Three – Type Assignments

Section Five: Definitions

The definitions section contains the actual definitions for the object in the module.

Section Six - End

The keyword END indicates the end of the MIB.

Summary of Naming Conventions

<table>
<thead>
<tr>
<th>Item</th>
<th>Convention</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASN.1 Data Type</td>
<td>Initial Uppercase</td>
<td>DisplayString</td>
</tr>
<tr>
<td>Data Value</td>
<td>Initial Lowercase</td>
<td>copy-from-host (1)</td>
</tr>
<tr>
<td>Data Identifier</td>
<td>Initial Lowercase</td>
<td>invNotifyMssgTable</td>
</tr>
<tr>
<td>ASN.1 Keyword</td>
<td>All Uppercase</td>
<td>INTEGER</td>
</tr>
<tr>
<td>ASN.1 Macro</td>
<td>All Uppercase</td>
<td>OBJECT-TYPE</td>
</tr>
<tr>
<td>Module Name</td>
<td>All Uppercase with –MIB suffix</td>
<td>NAFEM-SECURITY-MIB</td>
</tr>
<tr>
<td>Data Identifier for Table Entry Lists</td>
<td>Initial Uppercase</td>
<td>InvNotifyMssgEntry</td>
</tr>
</tbody>
</table>
Scalar Variable Definition

adminHostIpAddress OBJECT-TYPE
SYNTAX IpAddress
ACCESS read-write
STATUS mandatory
DESCRIPTION "" : = { adminGroup 1 }

Table Definition

The following example is a table definition using SMIv1 taken from the NAFEM Inventory Management MIB.

-- Inventory STORAGE TABLE

invInfoStorageTable OBJECT-TYPE
SYNTAX SEQUENCE OF InvInfoStorageEntry
ACCESS not-accessible
STATUS mandatory
DESCRIPTION "Table of information on inventory items for inventory stocking."
: := { invStorageAlarm 1 }

invInfoStorageEntry OBJECT-TYPE
SYNTAX InvInfoStorageEntry
ACCESS not-accessible
STATUS mandatory
DESCRIPTION "An entry in the table of information on inventory items for inventory stocking."
INDEX { invInfoStorageEntryIndex }
: := { invInfoStorageTable 1 }

invInfoStorageEntry : := SEQUENCE {
  invInfoStorageEntryIndex INTEGER,
  invInfoStorageForecastVol INTEGER,
  invInfoStorageOnHandVol INTEGER,
  invInfoStorageStockLevel INTEGER,
}

invInfoStorageEntryIndex OBJECT-TYPE
SYNTAX INTEGER
ACCESS read-only
STATUS mandatory
DESCRIPTION "An index to an entry in the table of information on inventory items for inventory stocking."
: := { invInfoStorageEntry 1 }
invInfoStorageForecastVol OBJECT-TYPE
SYNTAX INTEGER
ACCESS read-write
STATUS mandatory
DESCRIPTION “Projected usage of an inventory item. “
: := { invInfoStorageEntry 2 }

invInfoStorageOnHandVol OBJECT-TYPE
SYNTAX INTEGER
ACCESS read-write
STATUS mandatory
DESCRIPTION “Current on hand volume of an inventory item. “
: := { invInfoStorageEntry 3 }

invInfoStorageStockLevel OBJECT-TYPE
SYNTAX INTEGER
ACCESS read-write
STATUS mandatory
DESCRIPTION “Minimum desired stocking level of an inventory item. “
: := { invInfoStorageEntry 4 }

Tables may have more than one index and may be indexed by the indexes of other tables. The following example is defines two tables that use multiple indexes but share a common index.

exampleTable1 OBJECT-TYPE
SYNTAX SEQUENCE OF ExampleEntry1
ACCESS not-accessible
STATUS mandatory
DESCRIPTION “A table of data values. “
: := { example 1 }

exampleEntry1 OBJECT-TYPE
SYNTAX ExampleEntry1
ACCESS not-accessible
STATUS mandatory
DESCRIPTION “An entry in the example Table1. “
: := { exampleTable1 1 }

exampleEntry1 : := SEQUENCE {
  exampleIndex1 INTEGER,
  exampleIndex2 IPAddress,
  column3 INTEGER
}
exampleIndex1 OBJECT-TYPE
SYNTAX INTEGER
ACCESS read-only
STATUS mandatory
DESCRIPTION "The first index."
: := { exampleEntry1 1 }

exampleIndex2 OBJECT-TYPE
SYNTAX INTEGER
ACCESS read-only
STATUS mandatory
DESCRIPTION "The second index."
: := { exampleEntry1 2 }

exampleTable2 OBJECT-TYPE
SYNTAX SEQUENCE OF ExampleEntry1
ACCESS not-accessible
STATUS mandatory
DESCRIPTION "A table of data values."
: := { example 2 }

exampleEntry2 OBJECT-TYPE
SYNTAX ExampleEntry1
ACCESS not-accessible
STATUS mandatory
DESCRIPTION "An entry in the example Table1."
: := { exampleTable2 1 }

e
c
e
e
e

exampleEntry1 : := {
  column2 INTEGER,
  column3 INTEGER
}

Enumerations

The following example is an enumerated integer. Under the requirements of SMIv1, enumerated values are always greater than zero.

bulkFileTransferCommand OBJECT-TYPE
SYNTAX INTEGER {
  copy-from-host (1),
  copy-to-host (2),
  erase-file (3)
}
ACCESS read-write
STATUS mandatory
DESCRIPTION ""
: := { bulkFileTransferEntry 4 }
Resources

For additional information on formatting and structure of SNMP management information bases, see Understanding SNMP MIBs by David Perkins and Evan McGinnis; published by Prentice Hall (ISBN 0-13-437708-7).