Chapter 6

SNMP Management:
SNMPv2

Objectives

- Community-based security
- SNMPv2 enhancements
  - Additional messages
  - Formalization of SMI
- Get-bulk request and information-request
- SNMP MIB modifications
- Incompatibility with SNMPv1
- Proxy server
- Bilingual manager

Major Changes with SNMPv2

- Two significant messages were added.

  – Bulk data transfer: It allows to request and receive bulk data using the get-bulk message. This speeds up the get-next-request process and is especially useful to retrieve data from tables.

  – Manager-to-manager message: This extends the communication of management messages between management systems and thus makes network management systems interoperable.

-

Notes

- **Security features, originally to be in SNMPv2 moved to SNMPv3 due to lack of consensus**
- **SNMPv2, like SNMPv1, is community-based administrative framework**

Major Changes with SNMPv2

•Enhancements to SMI: SMIv2 is divided into three parts

- **Module definitions**: MODULE-IDENTITY macro is used to define an information module. It concisely conveys the semantics of the information module.

- **Object definitions**: OBJECT-TYPE macro defines the syntax and semantics of a managed object.

- **Trap definitions** is also termed notification and is defined by a NOTIFICATION-TYPE macro.

Major Changes with SNMPv2

•Textual conventions ((collectively, Type Definitions)) are designed to help define new data types.

• Conformance statements : help the customer objectively compare features of various products. It also keeps vendors honest in claiming their product as being compatible with a given SNMP version

• Row creation and deletion in table

• MIB enhancements : In SNMPv2, the Internet node in the MIB has two new subgroups: security and snmpV2, as shown in Figure 6.1. There are significant changes to System and SNMP groups of version 1.

• Transport mappings: There are several changes to the communication model in SNMPv2. Although use of UDP is the preferred transport protocol mechanism for SNMP management, other transport protocols could be used with SNMPv2. The mappings needed to define other protocols on to UDP are the subject of RFC 1906

SNMPv2 Internet Group

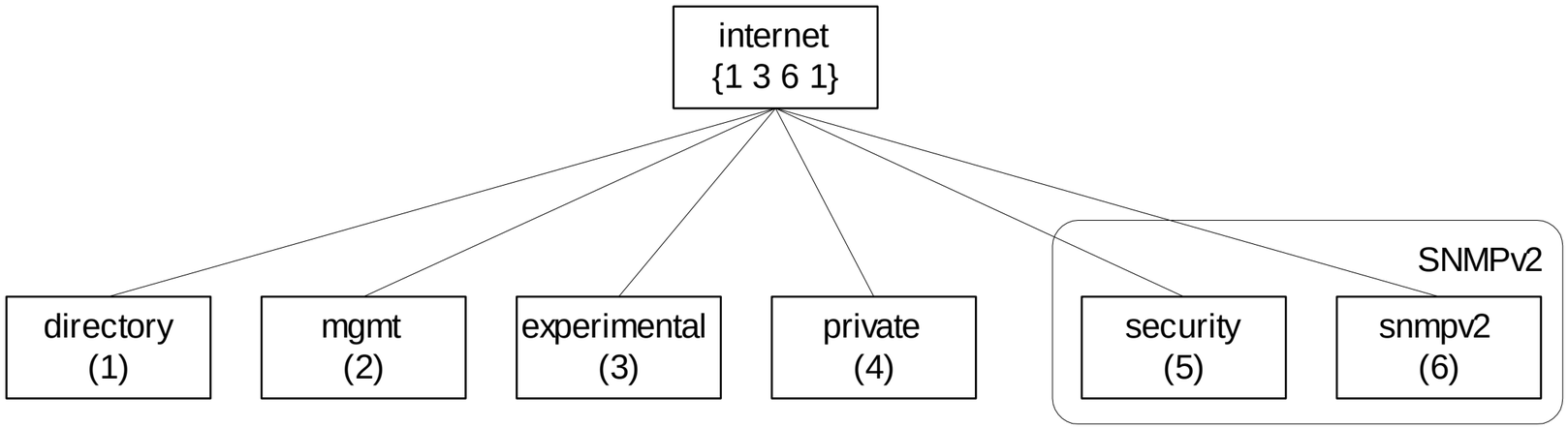


Figure 6.1 SNMPv2 Internet Group

Notes

- Objects added to System group
- Extensive modification of the SNMP group
- Additional SNMPv2 group added
- Security group is a placeholder

SNMPv2 NM Architecture

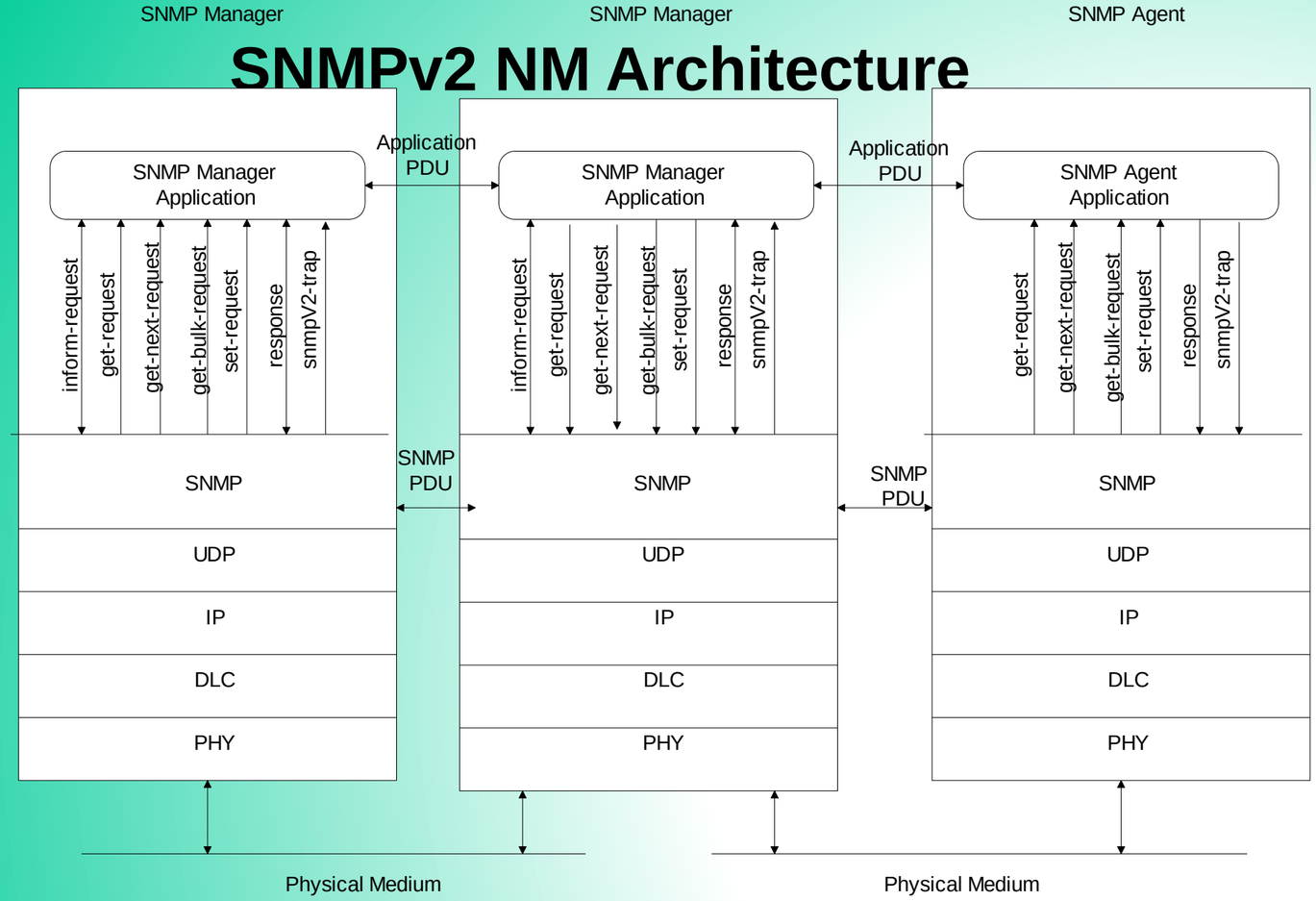


Figure 6.2 SNMPv2 Network Management Architecture

SNMPv2 New Messages

- inform-request
  - manager-to-manager message
- get-bulk-request
  - transfer of large data

SNMPv2 system architecture looks essentially the same as that of version 1, as shown in Figure 4.9. However, there are two significant enhancements in SNMPv2 architecture, which are shown in Figure 6.2. First, there are seven messages instead of five as in Figure 4.9. Second, two manager applications can communicate with each other at the peer level. Another message, report message, is missing from Figure 6.2. This is because even though it has been defined as a message, SNMPv2 Working Group did not specify its details. It is left for the implementers to generate the specifications. It is not currently being used and is hence omitted from the figure.

Module definition

Module Identity Macro

```
MODULE-IDENTITY MACRO ::=
BEGIN
        TYPE NOTATION ::=
                                "LAST-UPDATED" value (Update UTCTime)
                                "ORGANIZATION" Text
                                "CONTACT-INFO" Text
                                "DESCRIPTION" Text
                                RevisionPart
        VALUE NOTATION ::=
                                value (VALUE OBJECT IDENTIFIER)
        RevisionPart ::=        Revisions | empty
        Revisions ::= Revision | Revisions Revision
        Revision ::=
                "REVISION" value (UTCTime)
                "DESCRIPTION" Text
        -- uses the NVT ASCII character set
        Text ::= """" string """"
END
```

Figure 6.7 MODULE-IDENTITY Macro

- Module is a group of related assignments
- MODULE-IDENTITY macro defines the module definitions

The MODULE-IDENTITY macro is added to SMIv2 specifying an informational module. It provides administrative information regarding the informational module as well as revision history. SMIv2 MODULE-IDENTITY macro is presented in Figure 6.7.

Module Identity Macro

```
MODULE-IDENTITY MACRO ::=
BEGIN
        TYPE NOTATION ::=
                        "LAST-UPDATED" value (Update UTCTime)
                        "ORGANIZATION" Text
                        "CONTACT-INFO" Text
                        "DESCRIPTION" Text
                        RevisionPart
        VALUE NOTATION ::=
                        value (VALUE OBJECT IDENTIFIER)
        RevisionPart ::=    Revisions | empty
        Revisions ::= Revision | Revisions Revision
        Revision ::=
                "REVISION" value (UTCTime)
                "DESCRIPTION" Text
        -- uses the NVT ASCII character set
        Text ::= """" string """"
END
```

Figure 6.7 MODULE-IDENTITY Macro

- Module is a group of related assignments
- MODULE-IDENTITY macro defines the module definitions

```
isiMIBModule        MODULE-IDENTITY
        LAST-UPDATED    "9802101100Z"
        ORGANIZATION    "InfoTech Services Inc."
        CONTACT-INFO    "Mani Subramanian
                        Tele: 770-111-1111
                        Fax: 770-111-2222
                        email: manis@bellsouth.net"
        DESCRIPTION     " Version 1.1 of the InfoTech Services MIB module"
        Revision        "9709021500Z"
        DESCRIPTION     "Revision 1.0 on September 2, 1997 was a draft
                        version"
```

Figure 6.8 Example of MODULE-IDENTITY Macro

Figure 6.8 shows an example of a MODULE-IDENTITY macro (a real-world example of a non-existent module) for a network component vendor, InfoTech Services, Inc. (isi), which is updating their private-enterprises-isi MIB module {private.enterprises.isi}.

OBJECT

- OBJECT IDENTIFIER **defines the *administrative identification* of a node in the MIB**
- OBJECT-IDENTITY macro *assigns* an object identifier VALUE to an object in the MIB
- OBJECT-TYPE macro defines the *type* of a managed object

```
OBJECT-IDENTITY MACRO ::=
BEGIN
        TYPE NOTATION ::=
                        "STATUS"            Status
                        "DESCRIPTION"       Text
                        ReferPart

        VALUE NOTATION ::=
                        value (VALUE OBJECT IDENTIFIER)

        Status ::=          "current" | "deprecated" | "obsolete"
        ReferPart ::=       "REFERENCE" Text | empty
        Text ::=            """string """
END
```

Figure 6.9 OBJECT-IDENTITY Macro

```
isiRouter  OBJECT-IDENTITY
        STATUS          current
        DESCRIPTION     "An 8-slot IP router in the IP router
                         family."
        REFERENCE   "ISI Memorandum No. ISI-R123 dated
                         January. 20, 1997"
        ::= {private.enterprises.isi 1}
```

Figure 6.10(a) Example of OBJECT-IDENTITY Macro

OBJECT-IDENTITY / OBJECT-TYPE

- OBJECT-IDENTITY is high level description
- OBJECT-TYPE details description needed for implementation

```
isiRouter  OBJECT-IDENTITY
        STATUS          current
        DESCRIPTION     "An 8-slot IP router in the IP router
                         family."
        REFERENCE   "ISI Memorandum No. ISI-R123 dated
                     January. 20, 1997"
        ::= {private.enterprises.isi 1}
```

Figure 6.10(a) Example of OBJECT-IDENTITY Macro

```
routerIsi123     OBJECT-TYPE
        SYNTAX          DisplayString
        MAX-ACCESS      read-only
        STATUS          current
        DESCRIPTION     "An 8-slot IP router that can
                         switch up to 100 million packets
                         per second."
        ::= {isiRouter 1}
```

Figure 6.10(b) Example of OBJECT-TYPE Macro

Table Expansion

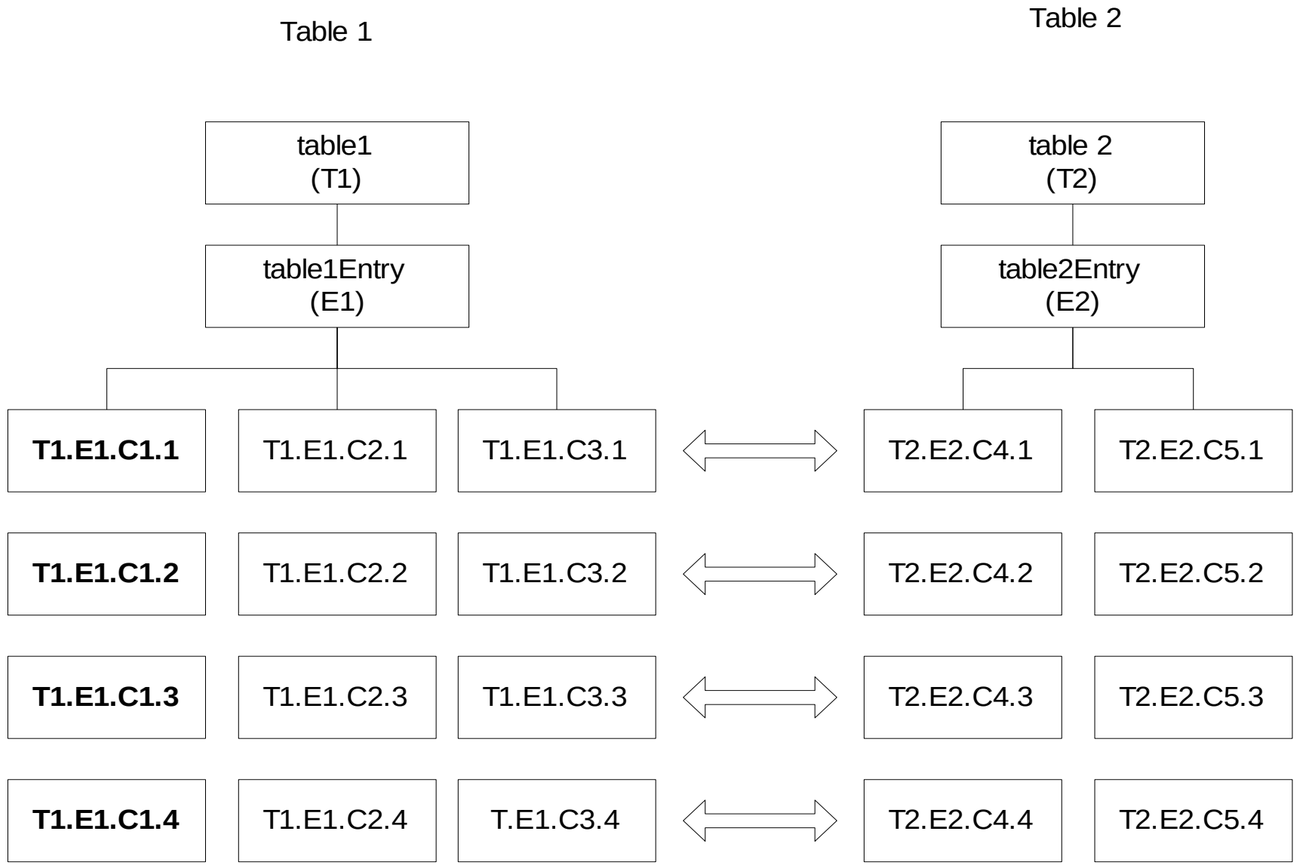
• Augmentation of a table (dependent table) adds additional columns to an existing table (base table)

• Dense table enables addition of more rows to base table

• Sparse table supplements less rows to a base table

Notes

Augmentation of Tables



Index:
First columnar object in Table 1

Conceptual rows:
1. T1.E1.C1.1
2. T1.E1.C1.2
3. T1.E1.C1.3
4. T1.E1.C1.4

Figure 6.11 Augmentation of Tables

Augmentation of Tables: Example

```
ipAddrTable     OBJECT-TYPE
        SYNTAX              SEQUENCE OF IpAddrEntry
        MAX-ACCESS      not-accessible
        STATUS              current
        DESCRIPTION     "The table …"
        ::= {ip 20}
ipAddrEntry     OBJECT-TYPE
        SYNTAX              IpAddrEntry
        MAX-ACCESS      not-accessible
        STATUS              current
        DESCRIPTION     "The addressing information…”
        INDEX               {ipAdEntAddr}
        ;;= {ipAddrTable 1}
ipAugAddrTable          OBJECT-TYPE
        SYNTAX              SEQUENCE OF IpAugAddrEntry
        MAX-ACCESS      not-accessible
        STATUS              current
        DESCRIPTION     "The augmented table to IP
                            address table defining board and
                            port numbers"
        ::= {ipAug 1}
ipAugAddrEntry          OBJECT-TYPE
        SYNTAX              IpAugAddrEntry
        MAX-ACCESS      not-accessible
        STATUS              current
        DESCRIPTION     "The addressing information …"
        AUGMENTS        {ipAddrEntry}
        ::= {ipAugAddrTable 1}
```

Figure 6.13 Example of Augmentation of Tables

Textual Convention

- Enables defining new data types
- Makes semantics of data types consistent and human readable
- allows creation of new data types using existing ones and applying restrictions to them
- An important textual convention in SNMPv2, *RowStatus* creates and deletes rows

Textual Convention

- SNMPV1: textual convention for data type DisplayString
  - Defined as an ASN.1 type assignment

- SNMPv2: textual convention for data type DisplayString
  - Defined as a data type and used to convey syntax and semantics of a textual convention

```
DisplayString ::= OCTET STRING
-- This data type is used to model textual information
taken from the NVT
-- ASCII character set. By convention, objects with this
syntax are
-- declared as having
-- SIZE (0..255)
```

```
DisplayString ::= TEXTUAL-CONVENTION
        DISPLAY-HINT      "255a"
        STATUS            current
        DESCRIPTION       "Represents textual
                          information taken from  the NVT
                          ASCII character set, as defined in
                          pages 4, 10-11 of RFC 854. …."
        SYNTAX            OCTET STRING (SIZE  (0..255) )
```

Textual Convention

```
TEXTUAL-CONVENTION MACRO ::=
BEGIN
        TYPE NOTATION ::=
                        DisplayPart
                        "STATUS"  Status
                        "DESCRIPTION"  Text
                        ReferPart
                        "SYNTAX"  Syntax

        VALUE NOTATION ::=
                        value (VALUE Syntax)

         DisplayPart ::=        "DISPLAY-HINT" Text | empty
         Status ::=             "current" | "deprecated" | "obsolete"
                        .........
END
```

Figure 6.18 TEXTUAL-CONVENTION Macro [RFC 1903]

Table 6.3 SMIv2 Textual Conventions for Initial Data Types

DisplayString	Textual information from NVT ASCII character set [RFC 854]
PhysAddress	Media- or physical-level address
MacAddress	IEEE 802 MAC address
TruthValue	Boolean value; INTEGER {true (1), false (2)}
TestAndIncr	Integer-valued information used for atomic operations
AutonomousType	An independently extensible type identification value
VariablePointer	Pointer to a specific object instance; e.g., syscontact.0, ifInOctets.3
RowPointer	Pointer to a conceptual row
RowStatus	Used to manage the creation and deletion of conceptual rows and is used as the value of the SYNTAX clause for the status column of a conceptual row
TimeStamp	Value of sysUpTime at which a specific occurrence happened
TimeInterval	Period of time, measured in units of 0.01 seconds
DateandTime	Date–time specifications
StorageType	Implementation information on the memory realization of a

Creation of Row: RowStatus

Table 6.4 RowStatus Textual Convention

State	Enumer-ation	Description
active	1	Row exists and is operational
notInService	2	Operation on the row is suspended
notReady	3	Row does not have all the columnar objects needed
createAndGo	4	This is a one-step process of creation of a row; immediately goes into active state
createAndWait	5	Row is under creation and should not be commissioned into service
destroy	6	Same as Invalid in EntryStatus. Row should be deleted

Notes

- Status: A new column is added to the conceptual table
- SYNTAX of Status is RowStatus
- Value of RowStatus is Enumerated INTEGER

Row Creation and Deletion

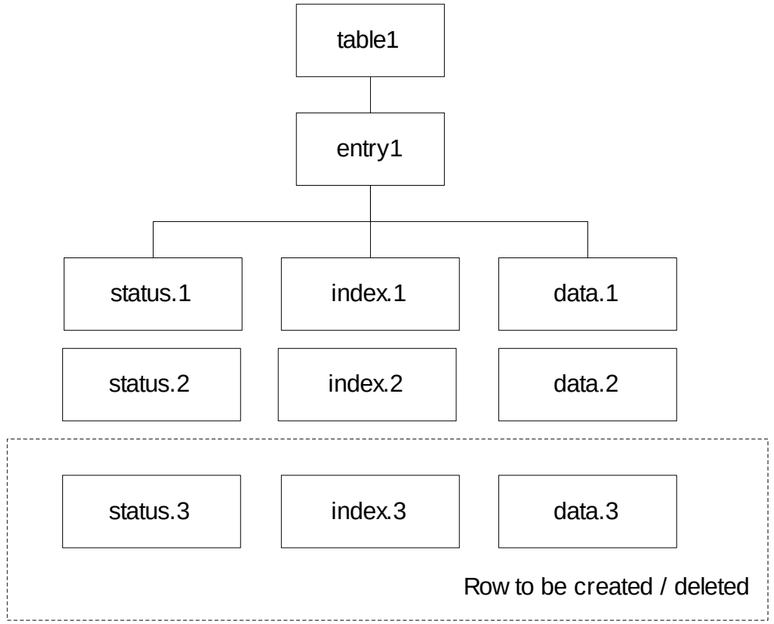


Figure 6.19 Conceptual Table for Creation and Deletion of Row

Notes

Create-and-Go Row Creation

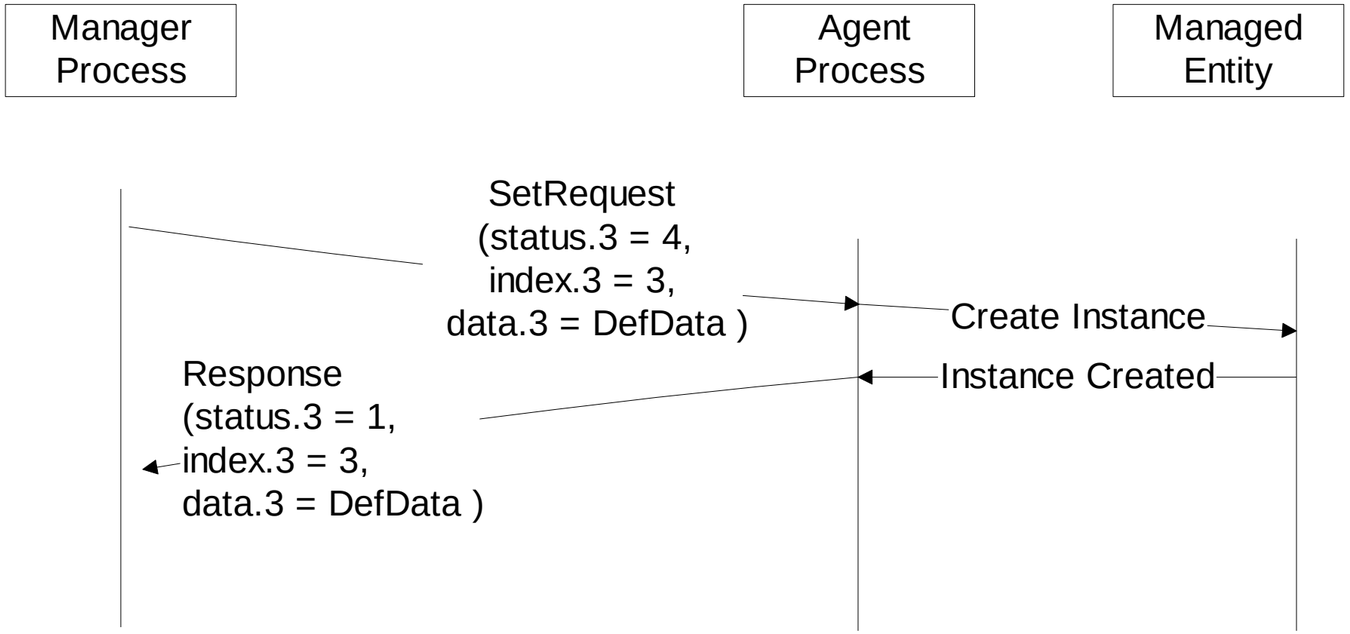


Figure 6.20 Create-and-Go Row Creation

Create-and-Wait: Row Creation

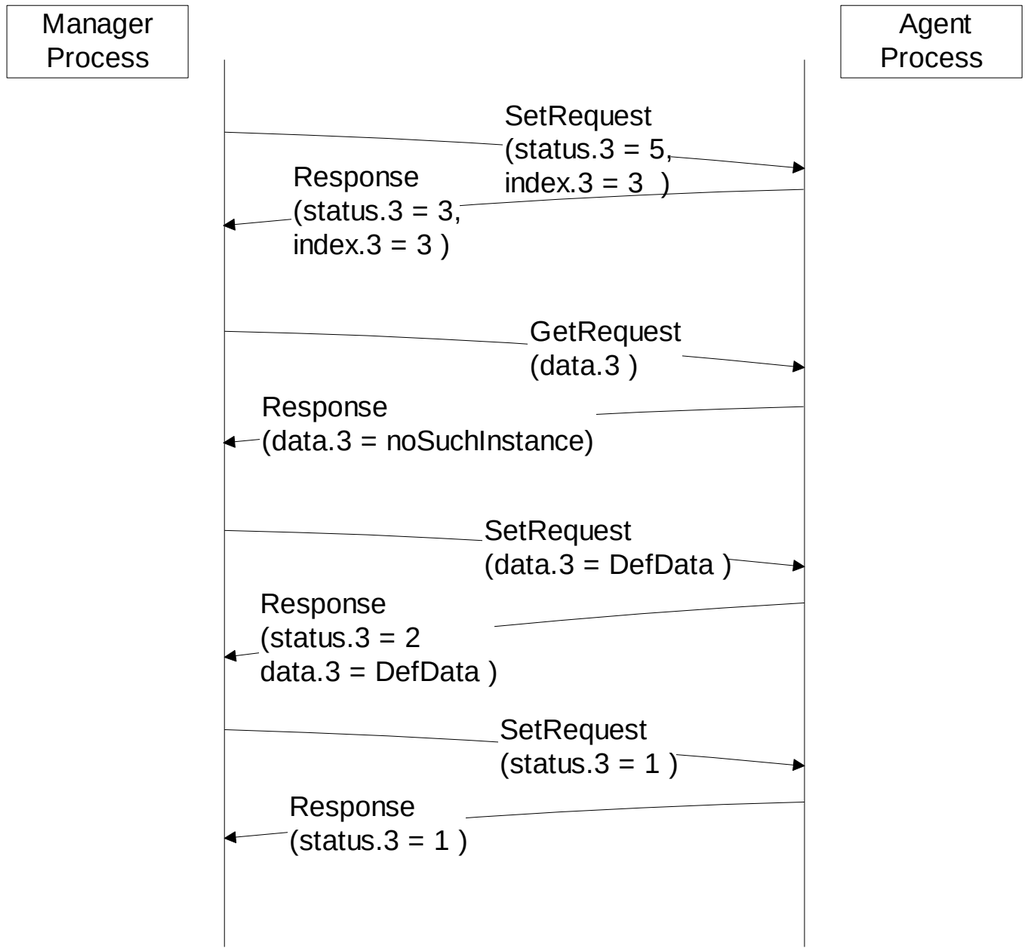


Figure 6.21 Create-and-Wait Row Creation

Row Deletion

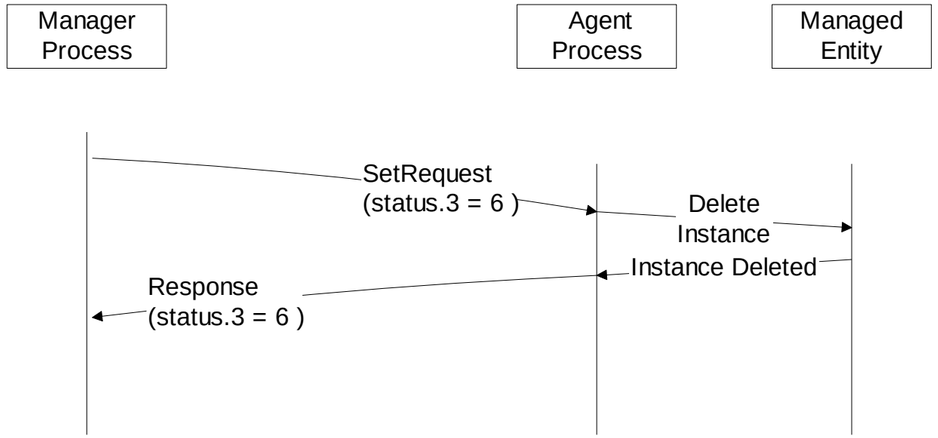
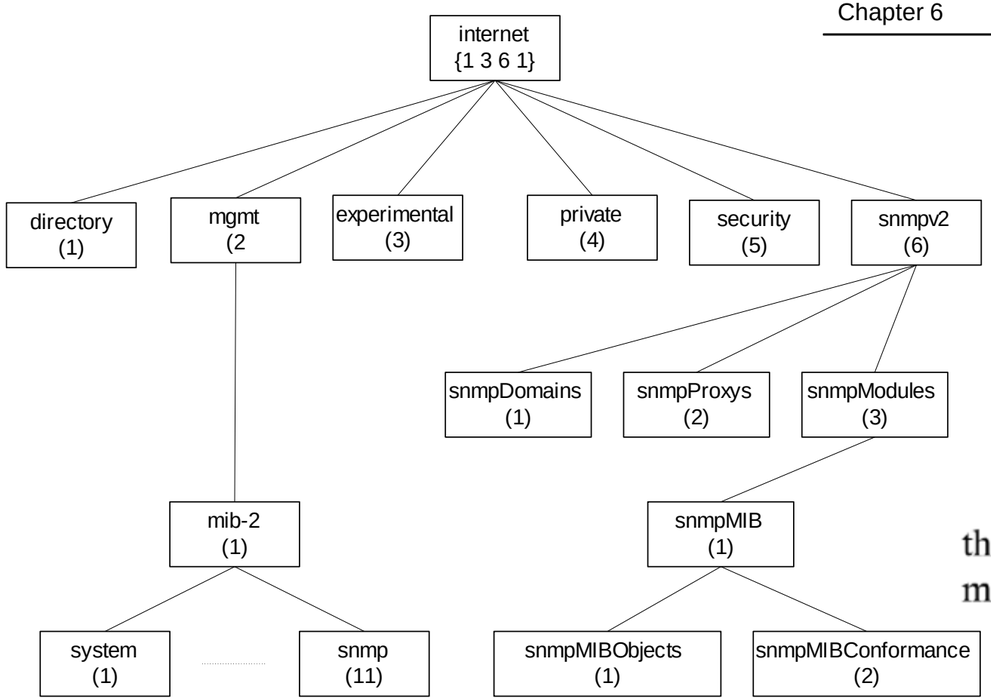


Figure 6.22 Row Deletion

SNMPv2 MIB



snmpDomains extends the SNMP standards to send management messages over transmission protocols other than UDP, which is the predominant and preferred way of transportation [RFC 1906]. Since UDP is the preferred protocol, systems that use another protocol need a proxy service to map on to UDP. Not much work has been done on snmpProxys, as of now.

The MIB module *snmpMIBObjects* addresses the new objects introduced in SNMPv2, as well as those that are obsolete. This is primarily concerned with trap, which has been brought into the same format as other PDUs. Also, many of the unneeded objects in the SNMP group have been made obsolete.

Figure 6.31 SNMPv2 Internet Group

conformance specifications and object groups are specified under the *snmpMIBconformance* module.

- Security is a placeholder
- System group: A table sysORTable added that lists resources that the agent controls; NMS configures NE through the agents.
- Most of the objects in the SNMPv1 obsoleted
- Object Groups and Notification Groups defined for conformance specifications.

Conformance statements

RFC 1904 defines SNMPv2 conformance statements for the implementation of network management standards. A product, generally, is considered to be in compliance with a particular standard when it meets the minimum set of features in its implementation. Minimum requirements for SNMPv2 compliance are called module compliance and are defined by an ASN.1 macro, MODULE-COMPLIANCE. It specifies the minimum MIB modules or a subset of modules that should be implemented. The actual MIB modules that are implemented in an agent are specified by another ASN.1 module, AGENT-CAPABILITIES. For the convenience of defining module compliance and agent capabilities, objects and traps have been combined into groups, which are subsets of MIB modules. Object grouping is defined by an ASN.1 macro, OBJECT-GROUP, and the group of traps is defined by the NOTIFICATION-GROUP macro.

Conformance: OBJECT-GROUP

- Conformance defined by
  - OBJECT-GROUP macro
  - NOTIFICATION-GROUP macro

Conformance: OBJECT-GROUP

OBJECT-GROUP

- Compiled during implementation, not at run time
- OBJECTS clause names each object
- Every object belongs to an OBJECT- GROUP
- Access defined by MAX-ACCESS, the maximum access privilege for the object

```
OBJECT-GROUP MACRO
BEGIN
        TYPE NOTATION ::=
                        ObjectsPart
                        "STATUS" Status
                        "DESCRIPTION" Text
                        ReferPart

        VALUE NOTATION ::=
                        value (VALUE OBJECT IDENTIFIER)

        ObjectsPart ::=     "OBJECTS" "{"objects"}"
        Objects ::=         Object | Objects "," Object
        Object ::=          value (Name  Object Name)
        Status ::=          "current" | "deprecated" | "obsolete"
        ReferPart ::=       "REFERENCE" Text  | empty

        -- uses the NVT ASCII character set
        Text ::=            """ string """
END
```

Figure 6.24 OBJECT-GROUP Macro

```
systemGroup         OBJECT-GROUP
        OBJECTS     {sysDescr, sysObjectID, sysUpTime, sysContact, sysName,
                    sysLocation, sysServices, sysORLastChange, sysORID,
                    sysORUptime, sysORDesc}
        STATUS      current
        DESCRIPTION     "The system group defines objects which are common
                        to all managed systems."
        ::= {snmpMIBGroups 6}
```

Figure 6.25 Example of OBJECT-GROUP Macro

Conformance: NOTIFICATION-GROUP

Notification Group. The notification group contains notification entities, or what was defined as traps in SMIv1.

- NOTIFICATION-GROUP
  - Contains trap entities defined in SMIv1
  - NOTIFICATIONS clause identifies the notifications in the group
  - NOTIFICATIONS-GROUP macro compiled during implementation, not at run time

Conformance: NOTIFICATION-GROUP

```
NOTIFICATION-GROUP MACRO
BEGIN
        TYPE NOTATION ::=
                        NotificationsPart
                        "STATUS" Status
                        "DESCRIPTION" Text
                        ReferPart

        VALUE NOTATION ::=
                        value (VALUE OBJECT IDENTIFIER)



NotificationsPart ::=       "NOTIFICATIONS" "{"Notifications"}"
Notifications ::=           Notification | Notifications "," Notification
Notification ::=            value (Name NotificationName)

        Status  ::=                 "current" | "deprecated" | "obsolete"
        ReferPart ::=               "REFERENCE" Text | empty

        -- uses the NVT ASCII character set
        Text ::= """ string """

END
```

Figure 6.26 NOTIFICATION-GROUP Macro

```
snmpBasicNotificationsGroup NOTIFICATION-GROUP
        NOTIFICATIONS           {coldStart, authenticationFailure}
        STATUS                  current
        DESCRIPTION     "The two notifications which an SNMP-2 entity is
                         required to implement."
        ::= {snmpMIBGroups 7}
```

Figure 6.27 Example of NOTIFICATION-GROUP Macro

Compliance

Module Compliance. The MODULE-COMPLIANCE macro, shown in Figure 6.28, defines the minimum set of requirements for implementation of one or more MIB modules. The expansion of the MODULE-COMPLIANCE macro is done during the implementation and not during run-time. The MODULE-COMPLIANCE macro can be defined as a component of the information module or as a companion module.

- Compliance has two classes of groups
  - MANDATORY-GROUPS ... Required
  - GROUP …Optional

```
-- compliance statements

snmpBasicCompliance MODULE-COMPLIANCE
        STATUS      current
        DESCRIPTION
                    "The compliance statement for SNMPv2 entities which
                    implement the SNMPv2 MIB."
        MODULE      -- this module
                MANDATORY-GROUPS  {snmpGroup, snmpSetGroup,
                                   systemGroup,
                                   snmpBasicNotificationsGroup}
                GROUP       snmpCommunityGroup
                DESCRIPTION
                    "This group is mandatory for SNMPv2 entities which support
                    community-based authentication."
        ::=  {snmpMIBCompliances  2 }

-- units of conformance
snmpGroup   OBJECT-GROUP    ::= {snmpMIBGroups 8}
snmpCommunityGroup      OBJECT-GROUP ::= {snmpMIBGroups 9}
snmpObsoleteGroup OBJECT-GROUP ::=  {snmpMIBGroups 10}
        …               …               …               …
```

Agent Capabilities

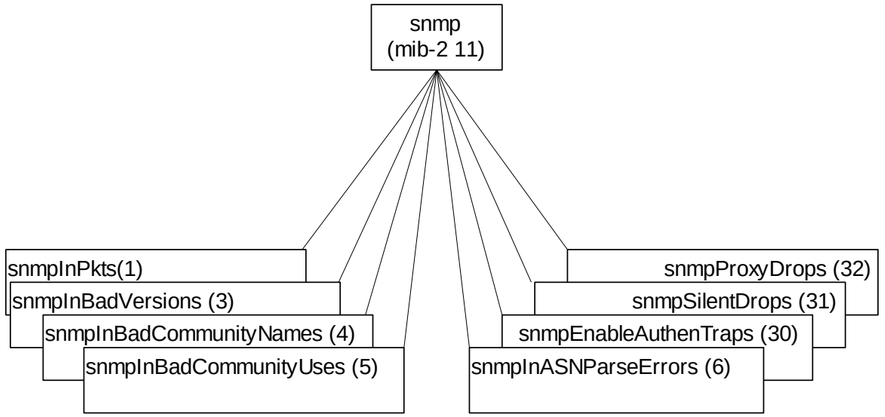
It specifies the minimum MIB modules or a subset of modules that should be implemented. The actual MIB modules that are implemented in an agent are specified by another ASN.1 module, AGENT-CAPABILITIES.

- AGENT-CAPABILITIES macro
  - SUPPORTS modules and includes groups
  - VARIATION identifies additional features

```
routerIsi123 AGENT-CAPABILITIES
        PRODUCT-RELEASE     "InfoTech Router isiRouter123 release 1.0"
        STATUS              current
        DESCRIPTION         "InfoTech High Speed Router"
        SUPPORTS            snmpMIB
                INCLUDES    {systemGroup, snmpGroup, snmpSetGroup,
                            snmpBasicNotificationsGroup  }
                VARIATION   coldStart
                        DESCRIPTION     "A coldStart trap is generated on all
                                        reboots."
        SUPPORTS            IF-MIB
                INCLUDES    {ifGeneralGroup, ifPacketGroup}
        SUPPORTS            IP MIB
                INCLUDES    {ipGroup, icmpGroup}
        SUPPORTS            TCP-MIB
                INCLUDES    {tcpGroup}
        SUPPORTS            UDP-MIB
                INCLUDES    {udpGroup}
        SUPPORTS            EGP-MIB
                INCLUDES    {egpGroup}
        ::= {  isiRouter 1  }
```

Figure 6.30 Example of AGENT-CAPABILITIES Macro

SNMPv2 SNMP MIB



SNMP Group Objects	
1,3,6,30,31,32	snmpGroup
4,5	snmpCommunity Group
7,23	not used
2,8-23, 24-29	snmpObsoleteGroup

Figure 6.33 SNMPv2 SNMP Group

Notes

- Compare this to SNMPv1 MIB!

snmpMIBObjects MIB

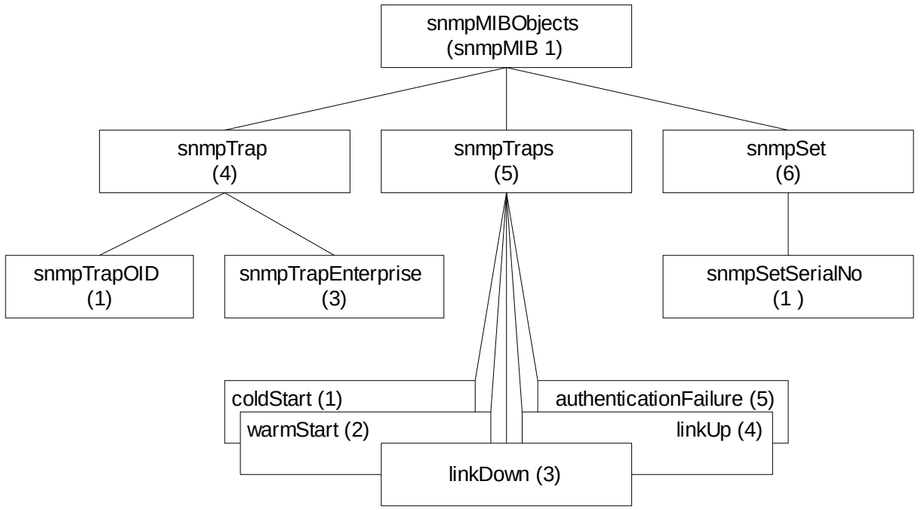


Figure 6.34 MIB Modules under snmpMIBObjects

Notes

SNMPv2 PDU

PDU Type	RequestID	Error Status	Error Index	VarBind 1 name	VarBind 1 value	…	VarBind n name	VarBind n value

Figure 6.37 SNMPv2 PDU (all but Bulk)

Notes

- Standardized format for all messages
- Interpretation of error status and error index fields; In v1, if error occurs status and index field filled, but varBindList blank

Interpretation in v2	Status	Index
varBindList ignored	x	
varBind of index field ignored	x	x

SNMPv2 PDU and Error Status

Table 6.11 Values for Types of PDU and Error-status Fields in SNMPv2 PDU

Field	Type	Value
PDU	0	Get-Request-PDU
	1	GetNextRequest-PDU
	2	Response-PDU
	3	Set-Request- PDU
	4	obsolete
	5	GetBulkRequest-- PDU
	6	InformRequest- PDU
	7	SNMPv2 - Trap- PDU
Error Status	0	noError
	1	tooBig
	2	noSuchName
	3	badValue
	4	readOnly
	5	genErr
	6	noAccess
	7	wrongType
	8	wrongLength
	9	wrongEncoding
	10	wrongValue
	11	noCreation
	12	inconsistentValue
	13	resourceUnavailable
	14	commitFailed
	15	undoFailed
	16	authorizationError
	17	notWritable
	18	inconsistentName

SNMPv2 GetBulkRequest PDU

PDU Type	RequestID	Non-Repeaters	Max Repetitions	VarBind 1 name	VarBind 1 value	…	VarBind n name	VarBind n value

Figure 6.38 SNMPv2 GetBulkRequest PDU

Notes

- *Error status* field replaced by *Non-repeaters*
- *Error index* field replaced by *Max repetitions*
- No one-to-one relationship between request and response

Get-Bulk-Request: Generic MIB

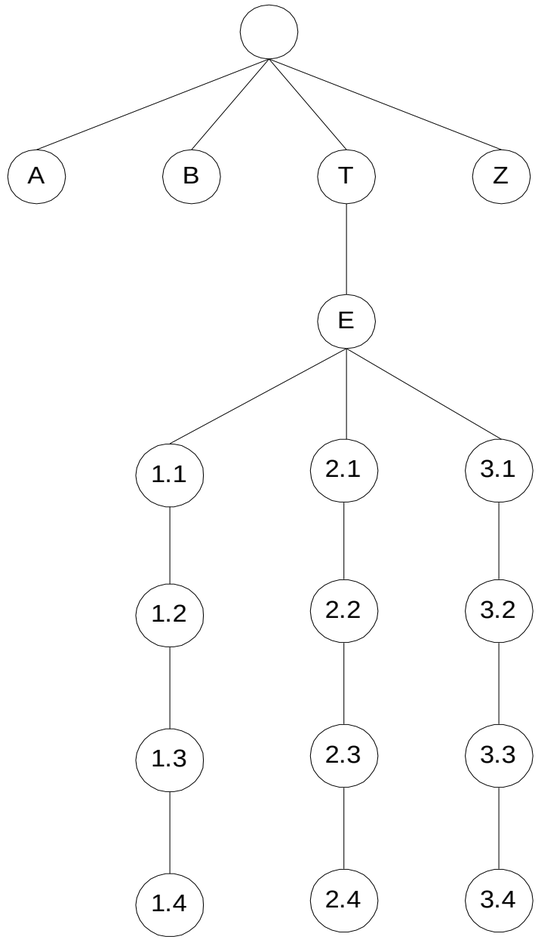
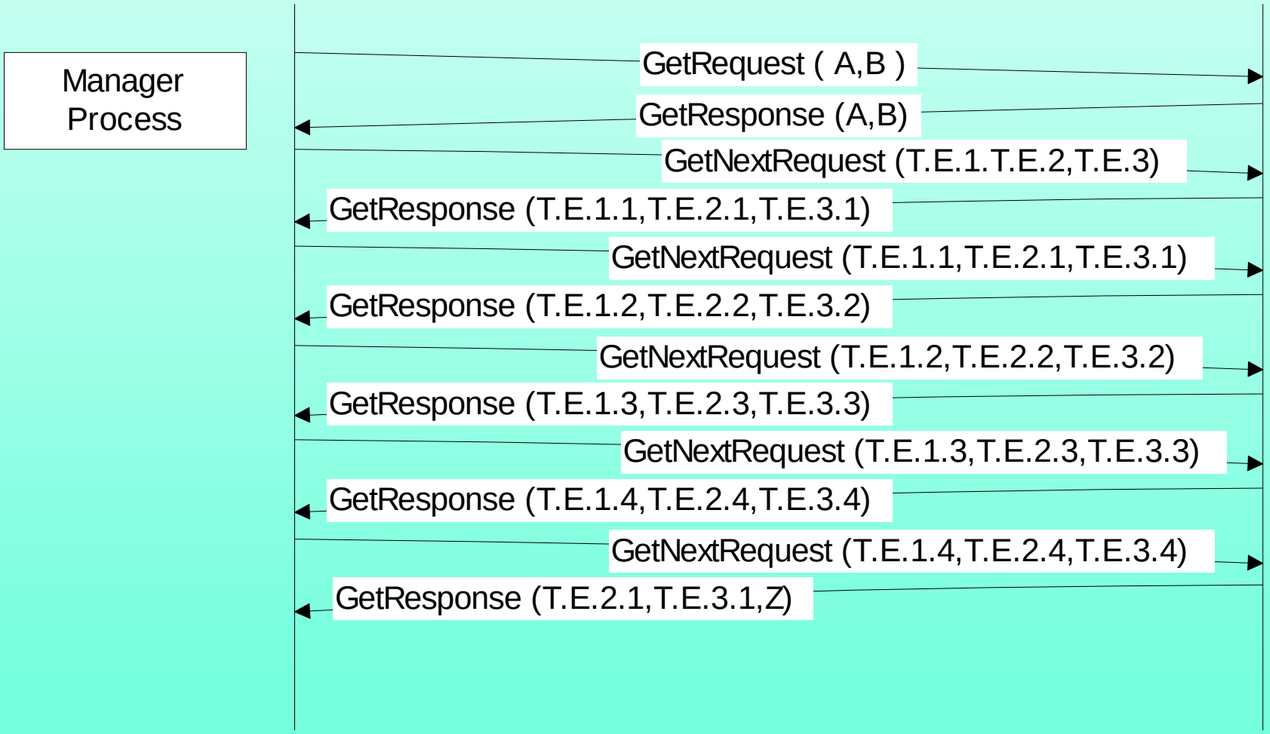
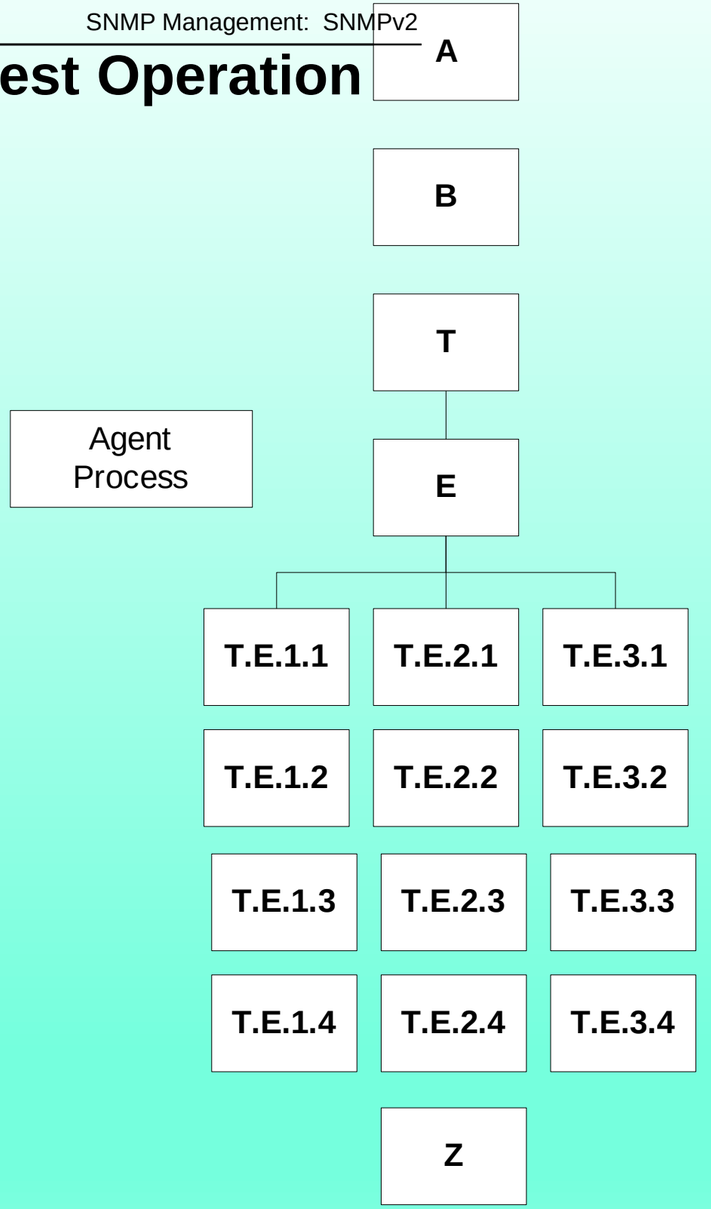


Figure 6.39 MIB for Operation Sequences in Figures 6.40 and 6.41

Get-Next-Request Operation

1st GetNextRequest correct to: GetNextRequest(T.E.1,T.E.2,T.E.3)





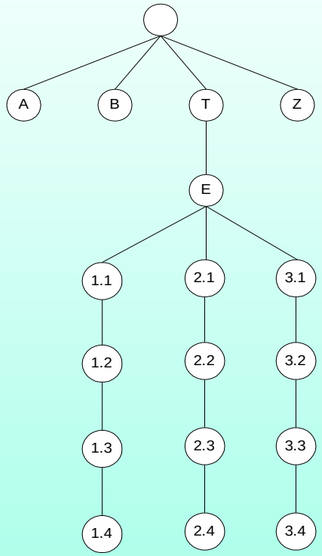


Figure 6.39 MIB for Operation Sequences in Figures 6.40 and 6.41

Figure 6.40 Get-Next-Request Operation for MIB in Figure 6.39

Get-Bulk-Request Operation

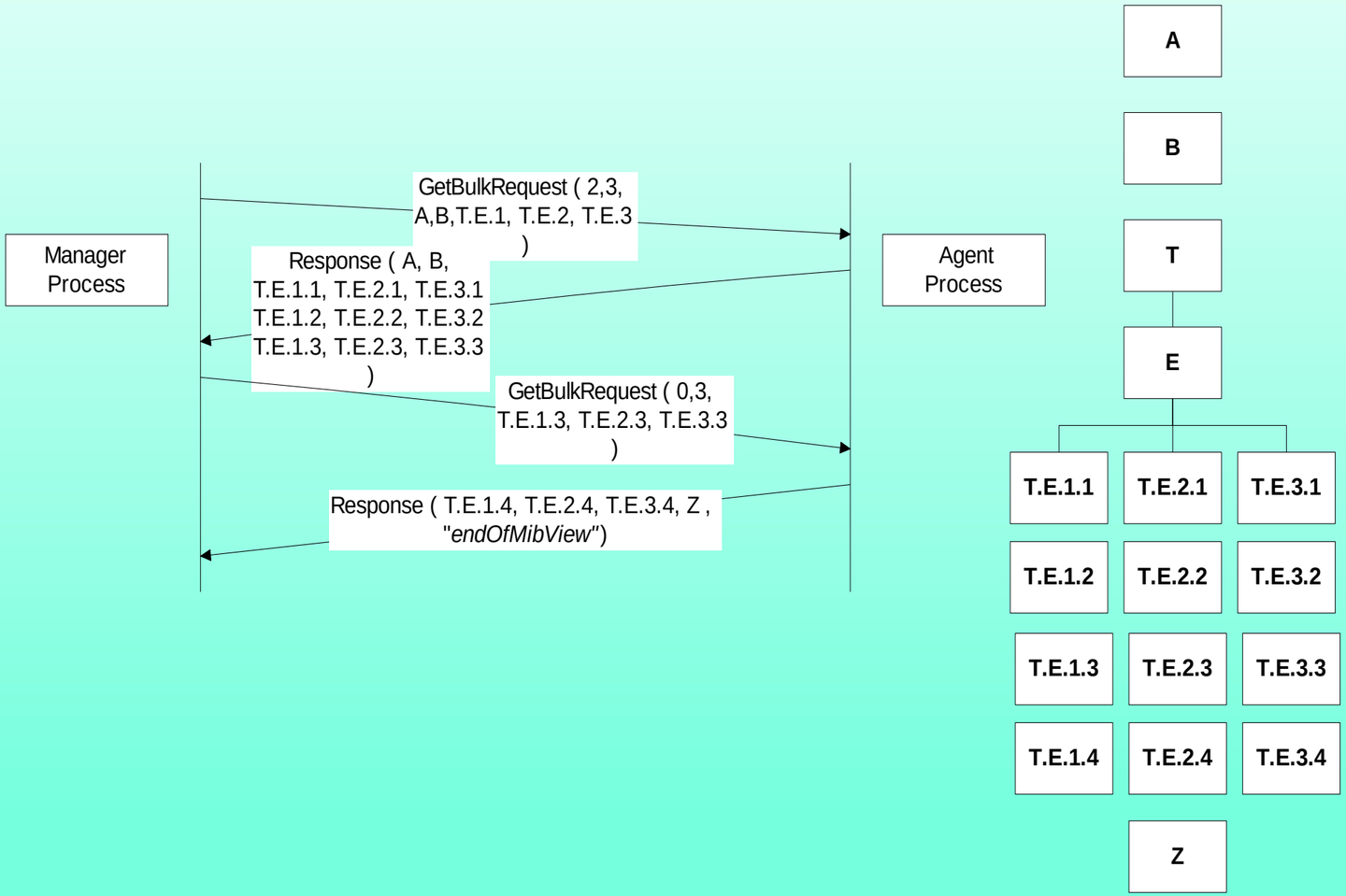


Figure 6.41 Get-Bulk-Request Operation for MIB in Figure 6.39

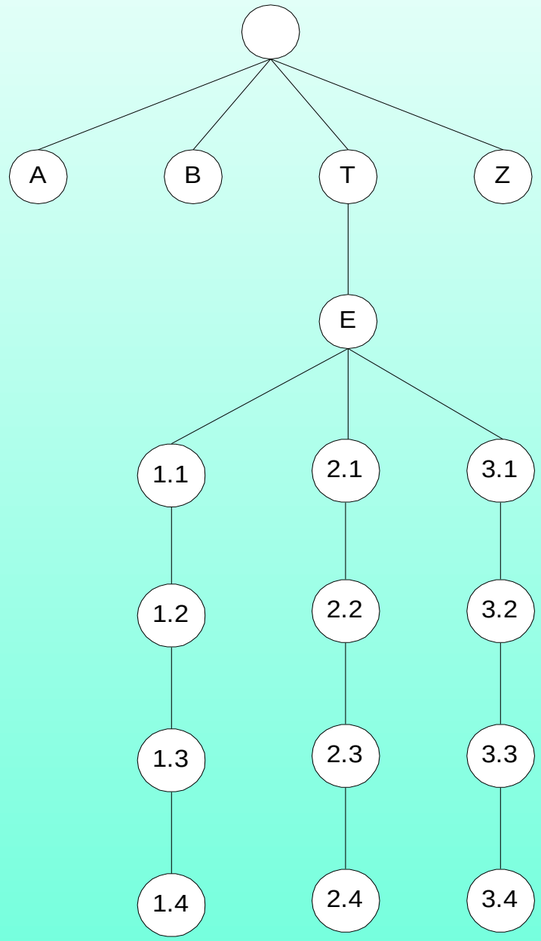
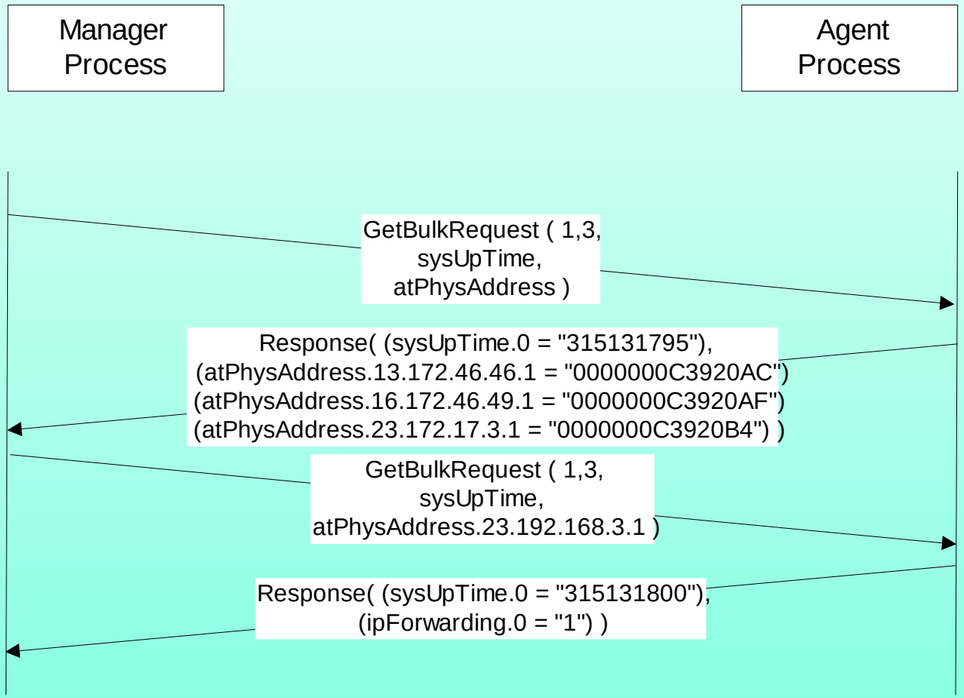


Figure 6.39 MIB for Operation Sequences in Figures 6.40 and 6.41

Get-Bulk-Request Example



atIfIndex	atPhysAddress	atNetAddress
23	0000000C3920B4	192.68.3.1
13	0000000C3920AC	172.46.46.1
16	0000000C3920AF	172.46.49.1

Figure 6.42 Get-Bulk-Request Example

Notes

SNMPv2 Trap

PDU Type	RequestID	Error Status	Error Index	VarBind 1 sysUpTime	VarBind 1 value	VarBind 2 snmpTrapOID	VarBind 2 value	...

Figure 6.43 SNMPv2 Trap PDU

Notes

- Addition of NOTIFICATION-TYPE macro
- OBJECTS clause, if present, defines order of variable bindings
- Positions 1 and 2 in VarBindList are sysUpTime and snmpTrapOID

```
linkUp NOTIFICATION-TYPE
        OBJECTS         {  ifIndex  }
        STATUS          current
        DESCRIPTION     "A linkUp trap signifies that the SNSMPv2 entity,
                        acting in an agent role, recognizes that one of the
                        communication links represented in its configuration
                        has come up."
```

Figure 6.44 Example of OBJECTS Clause in NOTIFICATION-TYPE

Inform-Request

PDU Type	RequestID	Error Status	Error Index	VarBind 1 sysUpTime	VarBind 1 value	VarBind 2 snmpTrapOID	VarBind 2 value	.. .

Figure 6.43 SNMPv2 Trap PDU

Notes

- Inform-Request behaves as trap in that the message goes from one manager to another unsolicited
- The receiving manager sends response to the sending manager

Bilingual Manager

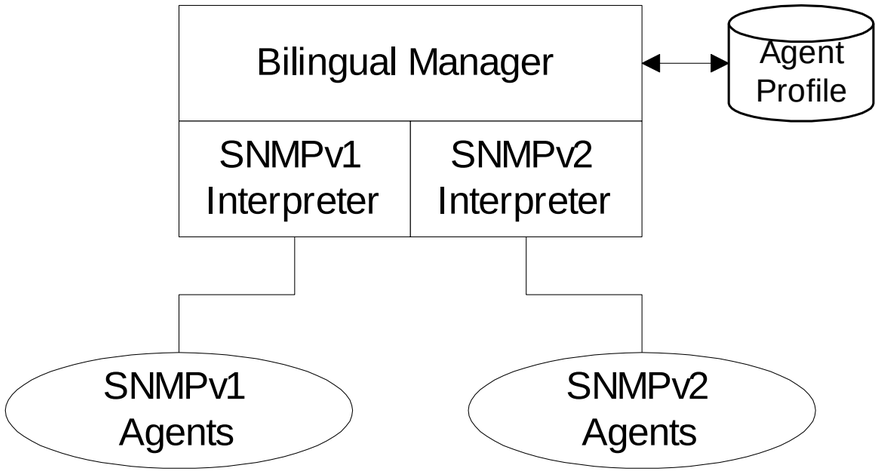


Figure 6.45 SNMP Bilingual Manager

Notes

- Compatibility with SNMPv1
  - Bilingual Manager
  - Proxy Server
- Bilingual Manager expensive in resource and operation

SNMP Proxy Server

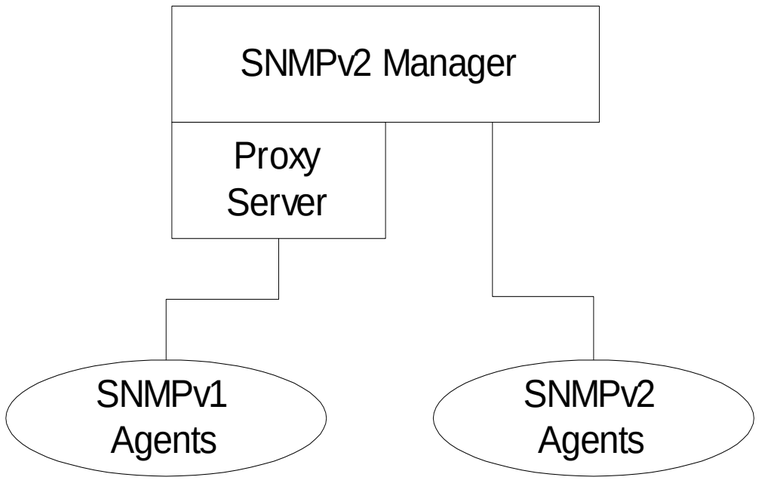


Figure 6.46 SNMPv2 Proxy Server Configuration

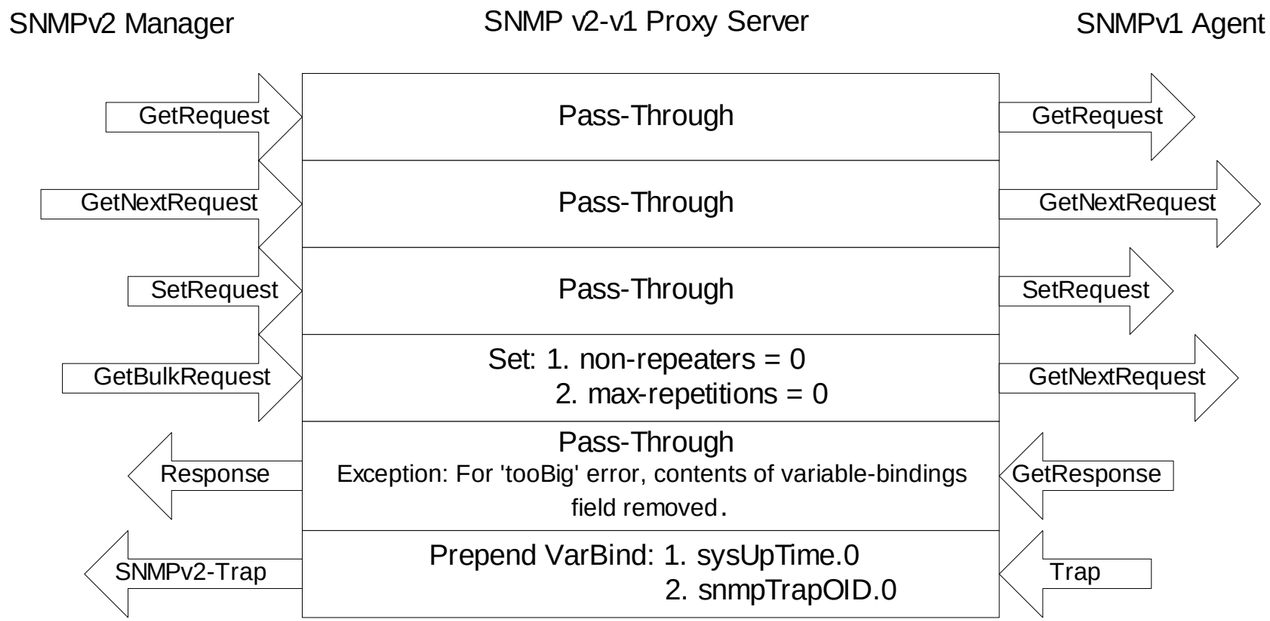


Figure 6.47 SNMP v2-v1 Proxy Server