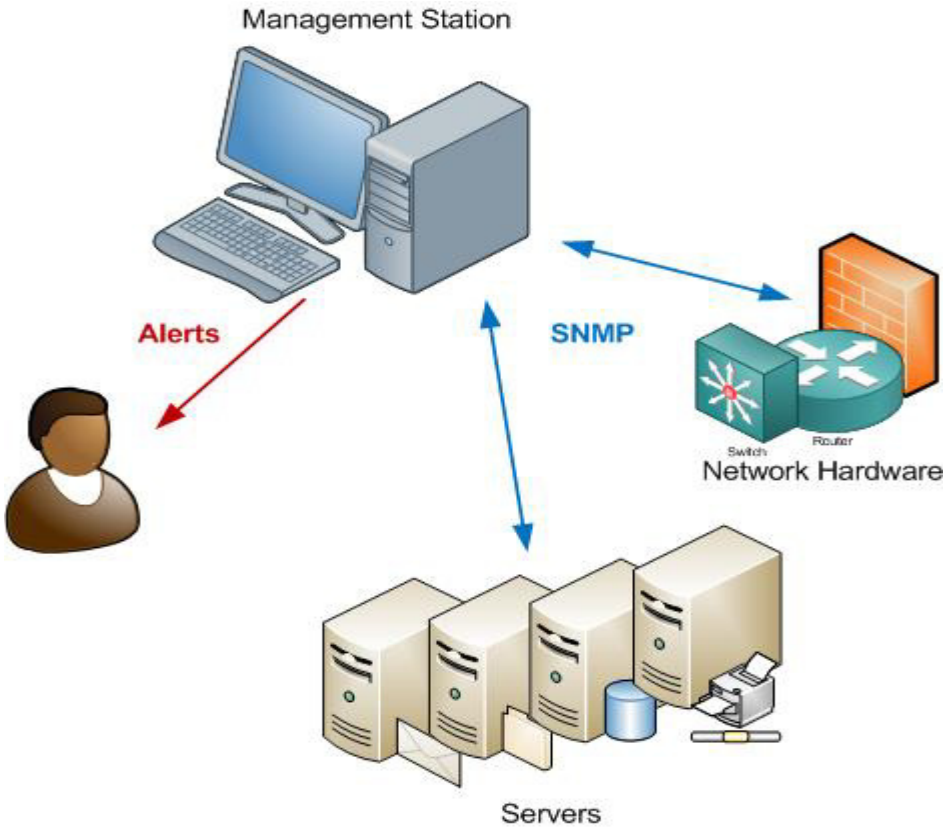
Administration Systèmes et Réseaux

Chapitre 4

Le Protocole SNMP
(Simple Network Management Protocol)

SNMP : Motivation

- Nécessité d ’avoir un protocole permettant de remonter des informations sur l ’activité des différentes ressources du réseau (les serveurs, les routeurs, les commutateurs, etc).



Présentation de SNMP

- Protocole d'administration de machines supportant TCP/IP
  - SNMP Version 1 (SNMPv1) Défini dans la RFC 1157
    - Mécanisme de sécurité basé sur la notion de communauté (mot de passe en clair dans les requêtes et réponses)
  - SNMP Version 2 (SNMPv2) Défini dans les RFC 1905, 1906 et 1907
    - Introduit deux nouveaux types de paquets get-bulk-request et inform-request (communication entre plate-formes)
  - SNMP Version 3 (SNMPv3) Défini dans les RFC 2570, 2571, 2572, 2573, 2574 et 2575
    - Introduit de nouveaux mécanismes de sécurité (authentification forte et confidentialité)

Présentation de SNMP

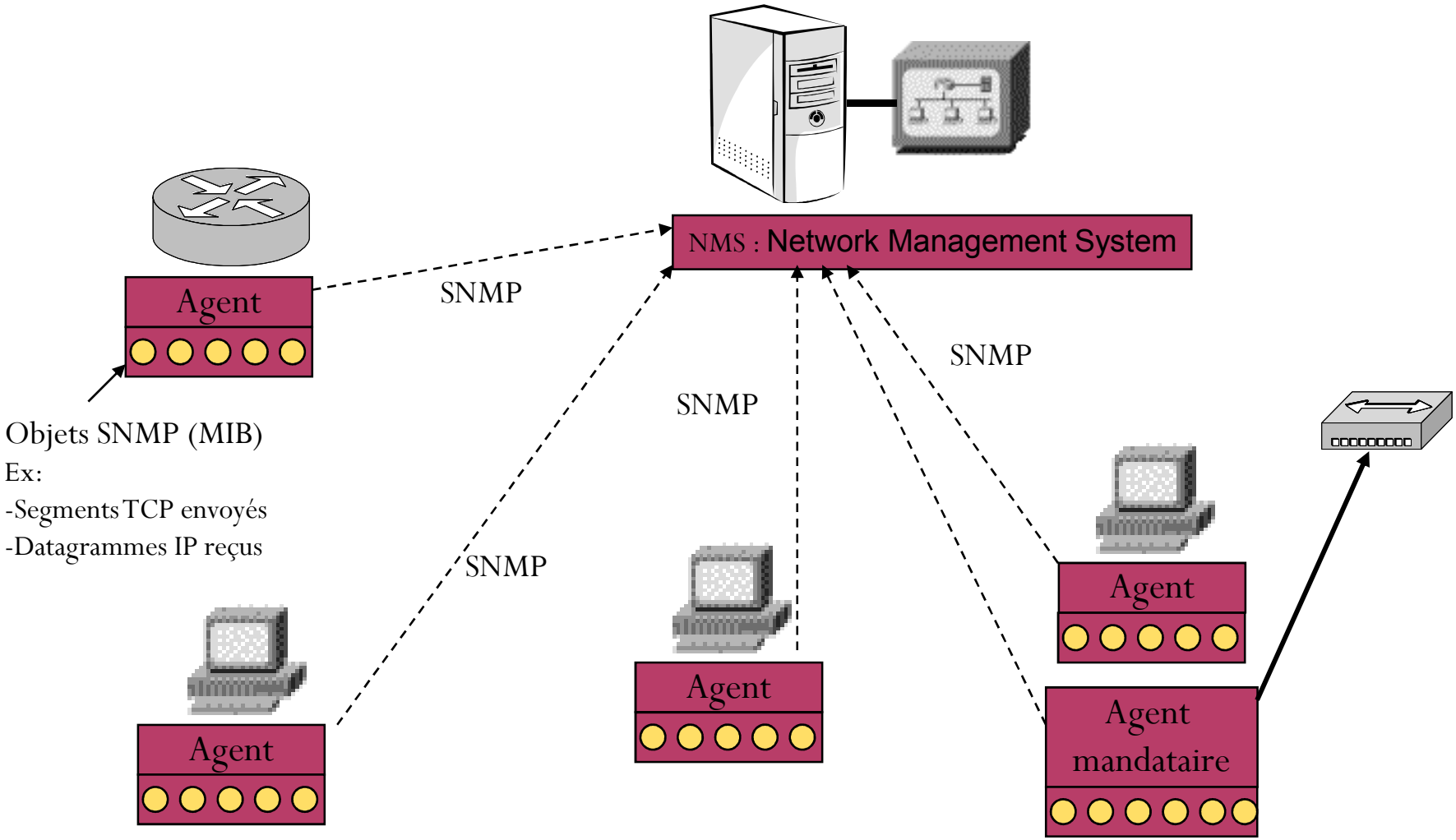
- Répond à un grand nombre de besoins :
  - Administrer à distance des machines indépendamment de leur **architecture**
  - Disposer d'une cartographie du réseau
  - Fournir un inventaire précis de chaque machine
  - Mesurer la consommation d'une application
  - Signaler les disfonctionnements

Modèle d ’administration SNMP

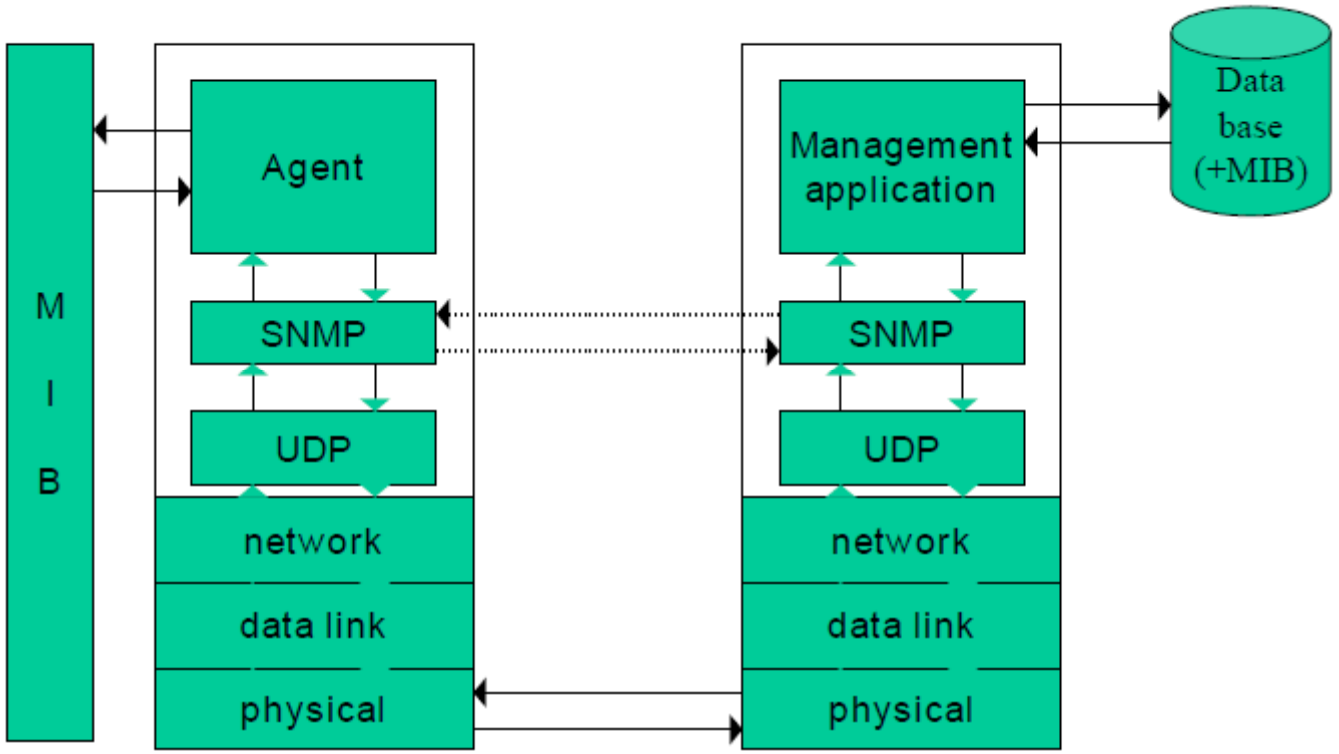
- Une administration SNMP est composée de trois types d'éléments :
  - des **agents** chargés de superviser un équipement. On parle d'agent SNMP installé sur tout type d'équipement.
  - une ou plusieurs **stations de gestion** capables d'interpréter les données
  - une **MIB** (Management Information Base) décrivant les informations gérées (objets administrés).

- SNMP permet **la supervision, le contrôle et la modification** des paramètres des éléments du réseau.

SNMP - éléments principaux



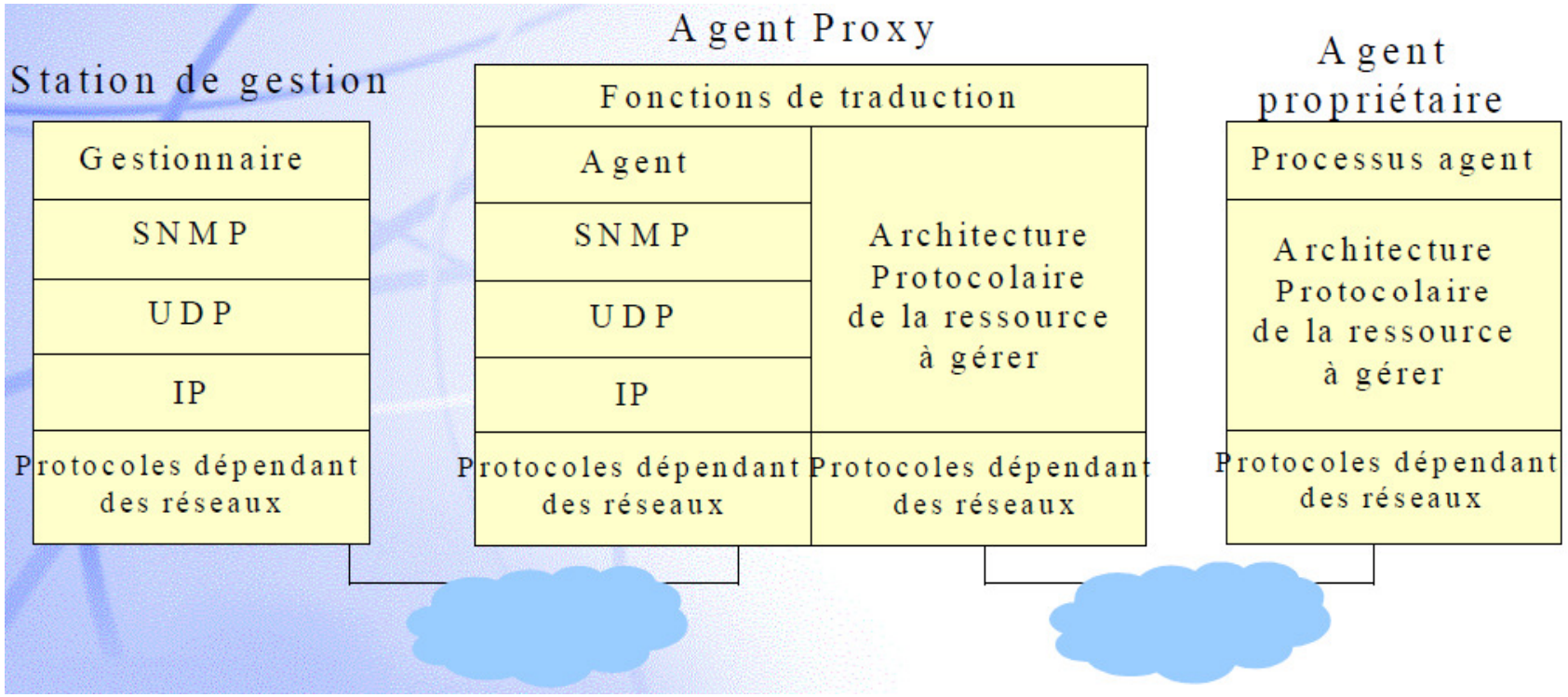
Plan Architectural



Agent mandataire (Proxy)

- L'utilisation de SNMP suppose que tous les agents et les stations d'administration supportent IP et UDP.
- ☹ Ceci limite l'administration de certains périphériques qui ne supportent pas la pile TCP/IP.
- ☹ De plus, certaines machines (ordinateur personnel, station de travail, contrôleur programmable,... implantent TCP/IP pour supporter leurs applications, mais ne souhaitent pas ajouter un agent SNMP.
- **=> utilisation de la gestion mandataire (les proxies)**
- ✓ Un agent SNMP agit alors comme mandataire pour un ou plusieurs périphériques

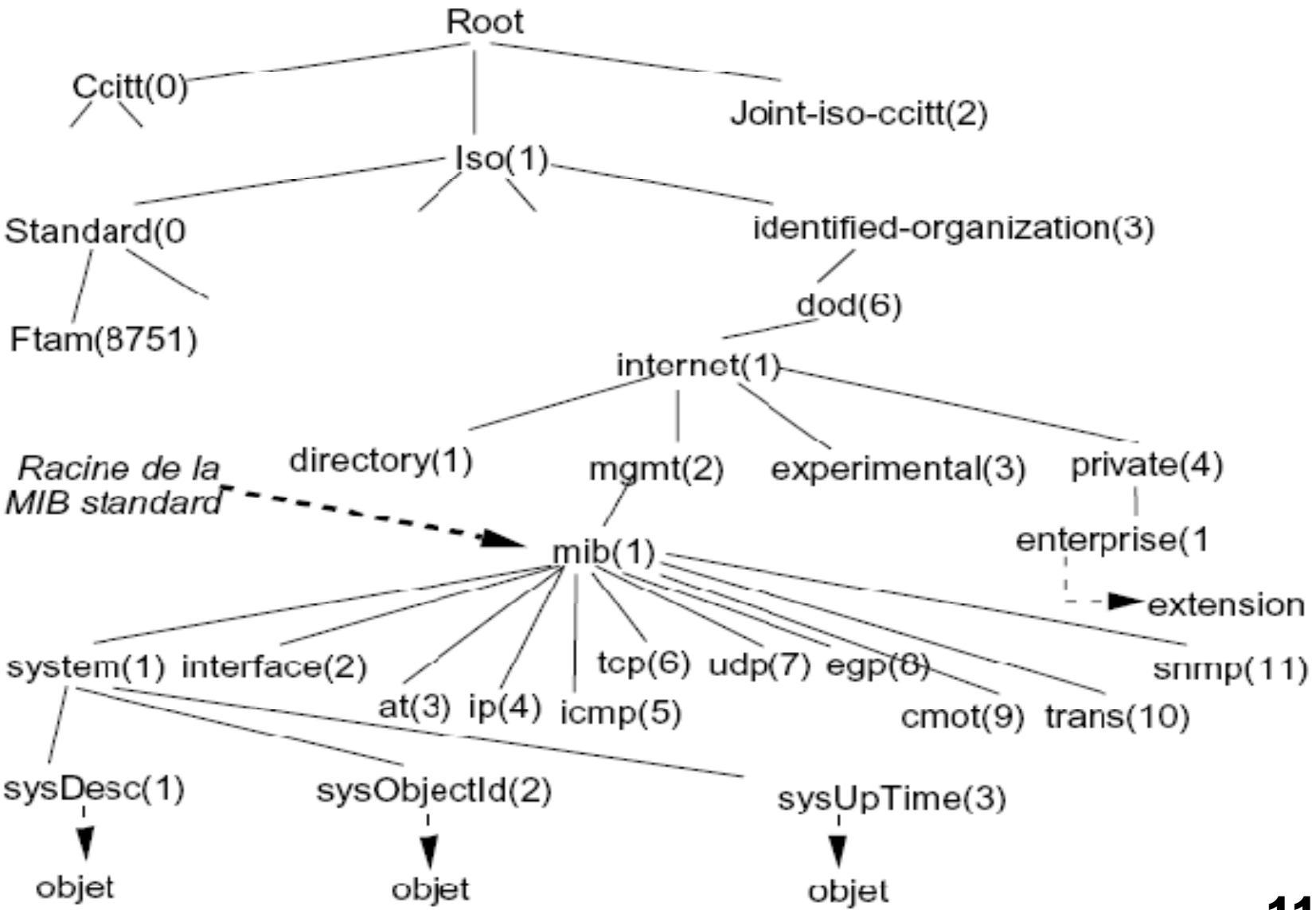
Agent mandataire (Proxy)



La MIB (Management Information Base)

- 1 ressource à gérer = 1 objet
- Les objets administrables sont une abstraction des ressources physiques (interfaces, équipements, etc.) et logiques (connexion TCP, paquets IP, etc.)
- **MIB** : collection structurée d'objets reconnus par les agents
- Chaque nœud dans le système doit maintenir une MIB qui reflète l'état des ressources gérées
  - Une entité d'administration peut accéder aux ressources du nœud en lisant les valeurs de l'objet ou en les modifiant
- **MIB: 2 objectifs**
  - Un schéma commun : SMI (Structure of Management Information)
  - Une définition commune des objets et de leur structure

Arbre des MIB accessibles

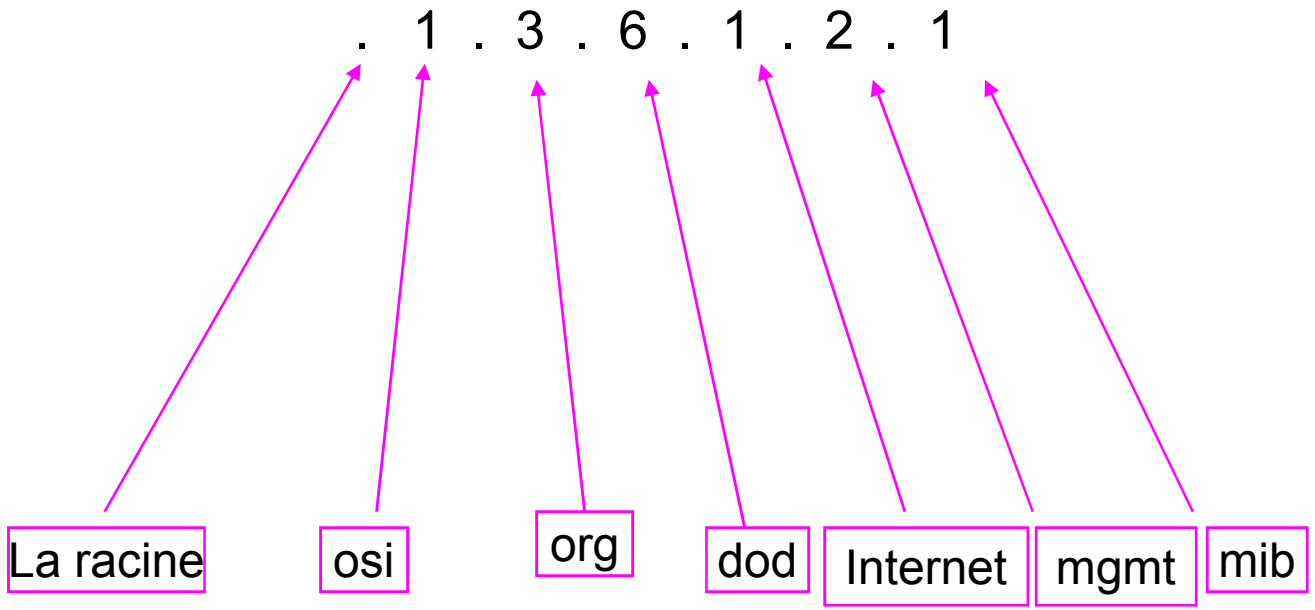


Identificateur d ’un objet de la MIB

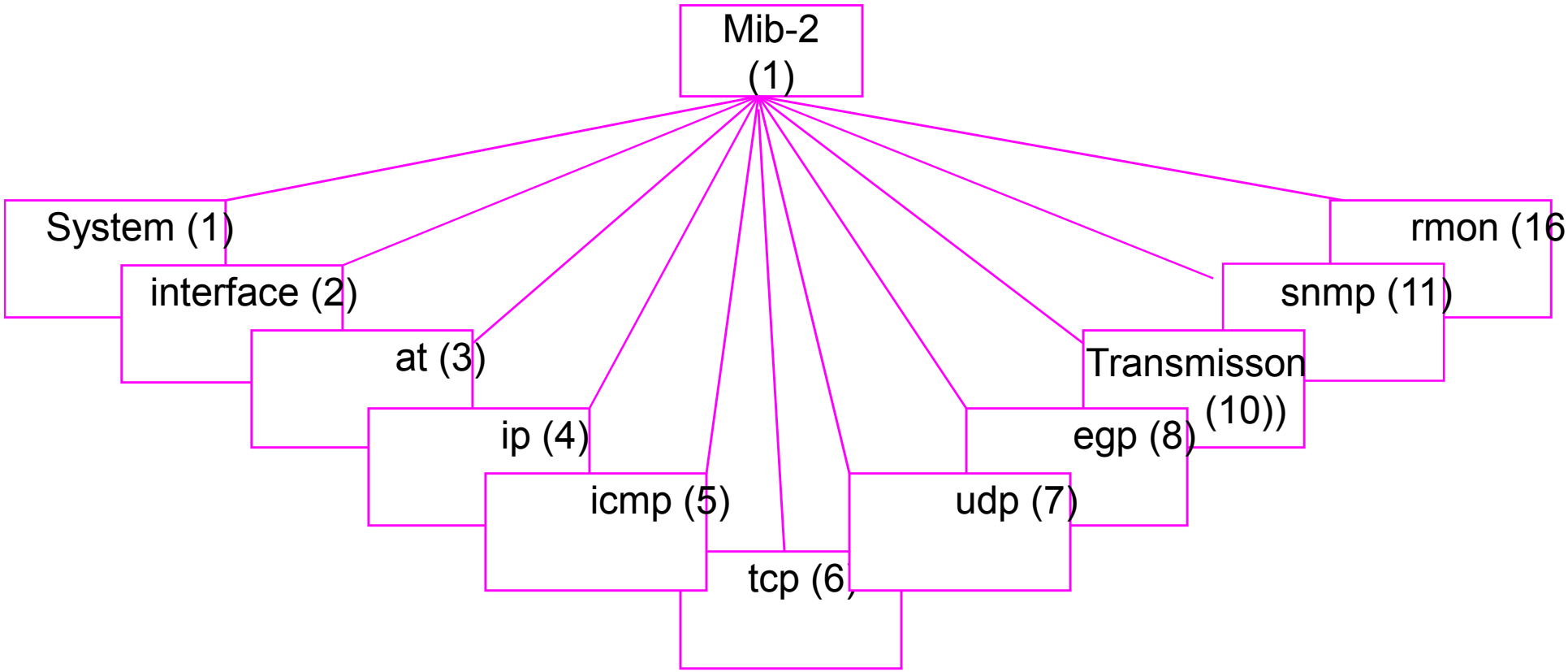
- Identificateur d ’un objet:

  ➔ Identificateur unique = séquence d ’entiers dont chacun représente la position de ces successeurs dans l ’arbre.

  - Exemple**: identificateur de l ’objet MIB** :



Le groupe MIB-2



Le groupe MIB-2

MIB-2

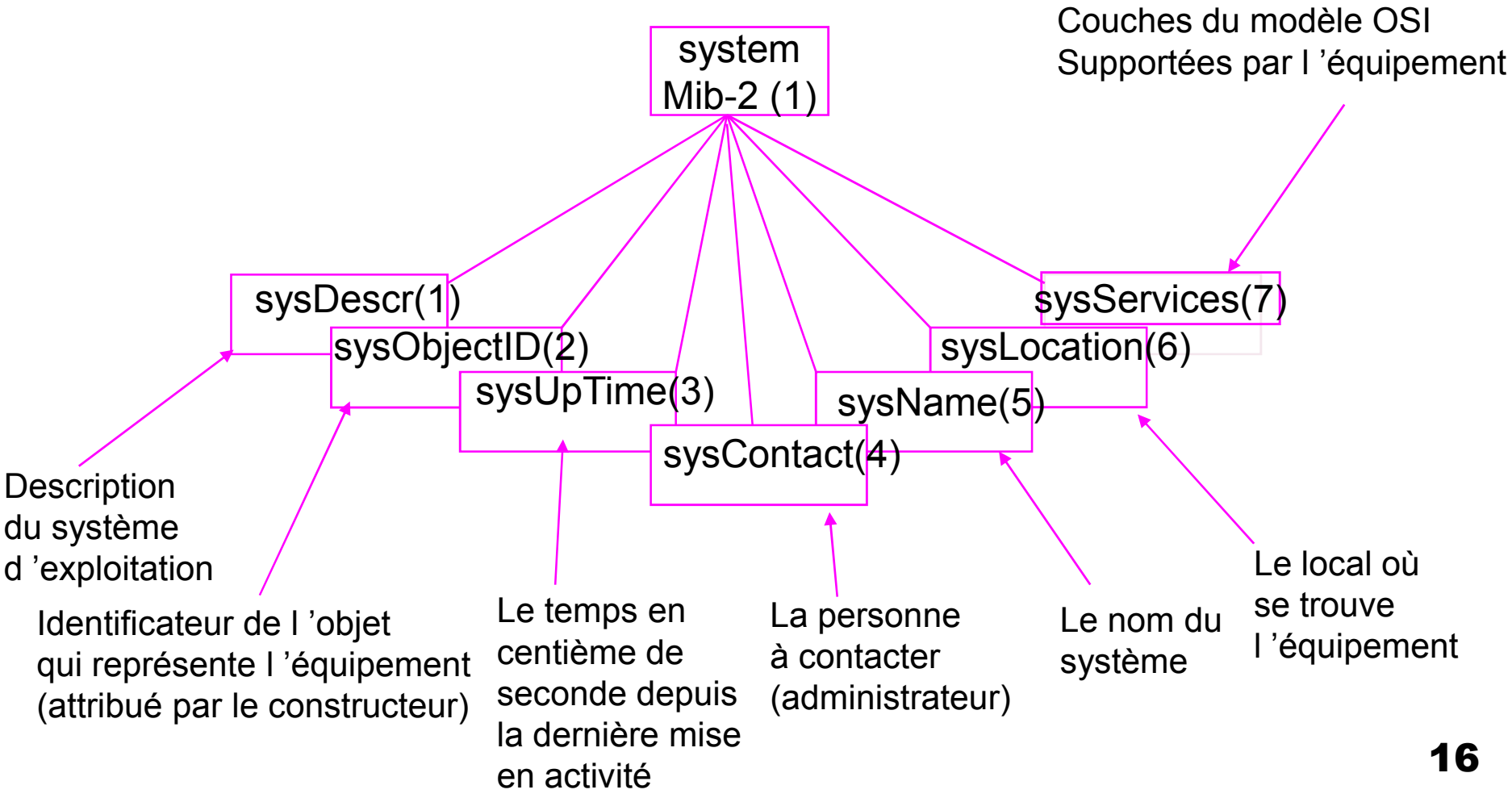
groupe	nbre éléments	commentaire
system	7	nœud dans le réseau
interfaces	25	interfaces réseau
at	5	IP address translation
ip	65	Internet Protocol
icmp	26	Internet Control Message Protocol
tcp	21	Transmission Control Protocol
udp	8	User Datagram Protocol
egp	22	Exterior Gateway Protocol
transmission	114	informations sur la transmission
snmp	28	SNMP
rmon	218	Remote network monitoring

La structure numérique de la MIB-2

system	1.3.6.1.2.1.1
interfaces	1.3.6.1.2.1.2
at	1.3.6.1.2.1.3
ip	1.3.6.1.2.1.4
icmp	1.3.6.1.2.1.5
tcp	1.3.6.1.2.1.6
udp	1.3.6.1.2.1.7
egp	1.3.6.1.2.1.8
rmon	1.3.6.1.2.1.9
transmission	1.3.6.1.2.1.10
snmp	1.3.6.1.2.1.11

Le groupe « System »

- **System** : correspond au nom de l'agent, n° de version, type de la machine, nom du système d'exploitation, etc.



Le groupe « Interface »

ifNumber : le nombre d'interfaces réseau

ifTable: Table qui contient une ligne par interface

ifIndex : Index de l 'interface (son numéro)

ifDescre : Description de l 'interface

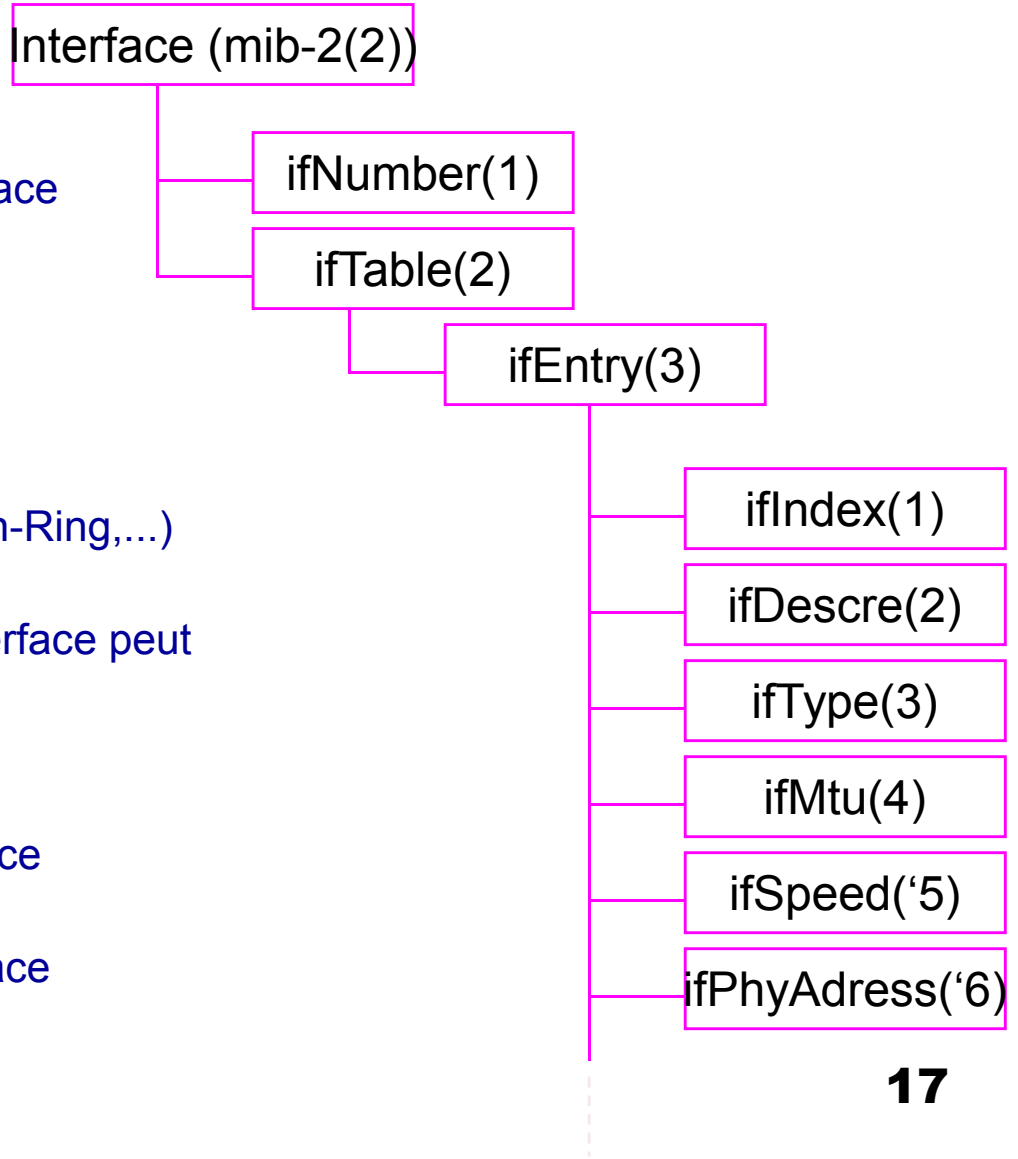
ifType : le type de l 'interface (Ethernet, Token-Ring,...)

i**fMtu** : le nombre maximum d 'octet que l 'interface peut envoyer ou recevoir

ifSpped : Une estimation du débit de l 'interface

ifPhyAdress : l 'adresse physique de l 'interface

....



Le groupe « IP »

ipForwarding : Agit comme passerelle, ou non

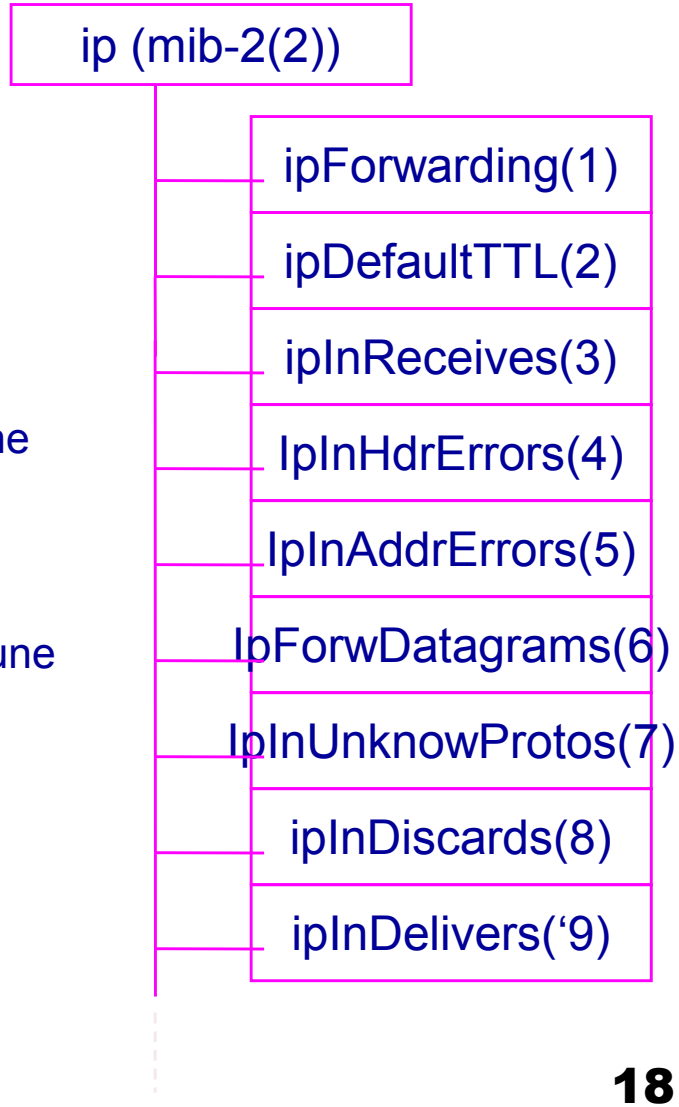
ipDefault TTL : la valeur par défaut du TTL ajouté dans un paquet IP

ipInReceives : Le nombre total de paquets IP reçus

IpInHdrErrors : Le nombre total de paquets écartés dus à une erreur sur l 'en-tête

IpInAddrErrors : Le nombre total de paquets écartés dus à une erreur sur l 'adresse de destination

IpForwDatagrams : Le nombre total de paquets dont l'entité réceptrice ne représente pas la destination finale.



Les autres groupes

icmp : statistiques sur les paquets ICMP reçus, envoyés et erronés

exemple: compter le nombre total de messages icmp reçus, reçus par erreur ou non envoyés

tcp : rend compte des connexions TCP en cours et leurs paramètres

de type nombre max de connexions simultanées permises, nombre d'ouvertures actives, l'état de chaque connexion (écoute, time-wait,...).

udp : - 4 compteurs renseignent sur le nombre de datagramme

UDP envoyés, reçus, en erreur, ...

egp : gère le protocole egp (External gateway protocol)

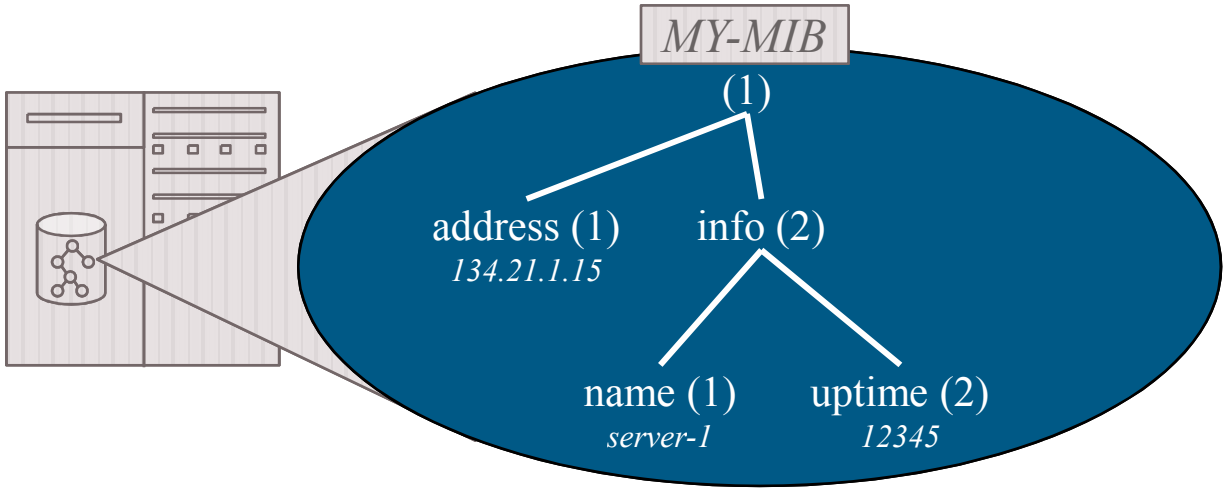
(routage des paquets entre routeurs). On a le nbre de paquets entrants, sortants, en erreur,

la table des routeurs adjacents, des infos sur les routeurs...

snmp : requis pour chaque entité mettant en oeuvre le protocole SNMP.

Contient le nombre de messages SNMP entrants et sortants, le nombre de mauvaises versions reçues ou de nom de communauté invalide, la répartition du type de requêtes reçues et envoyées (get, get_next, set et trap)

MIB: Nommage des Managed Objects: MOs (I)

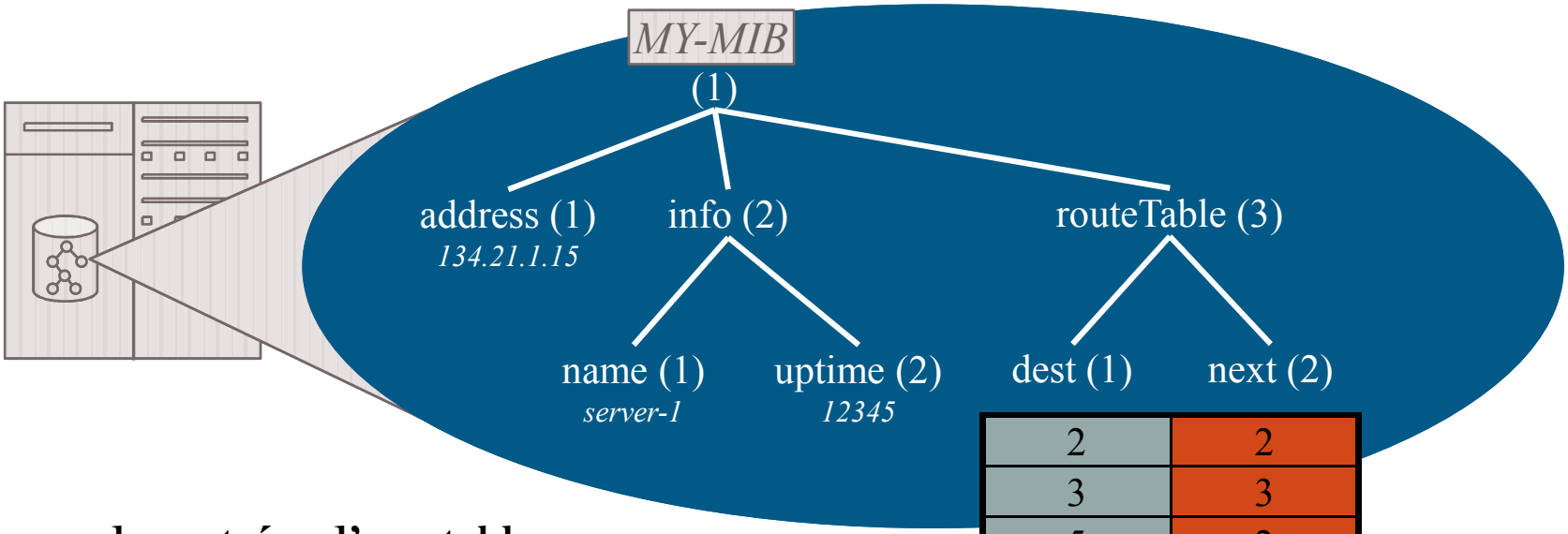


- **Nommage d'un MO de type scalaire**
  - **Par concaténation des identificateurs de la racine à l'objet et en rajoutant un 0 à la fin.**

- **Exemples**
  - **.1.1.0 ⇒ 134.21.1.15**
  - **.1.2.1.0 ⇒ "server-1"**
  - **.1.2.0 ⇒ ERREUR**

  Alternative (symbolique):
  - **.MY-MIB.info.uptime ⇒ 12345**

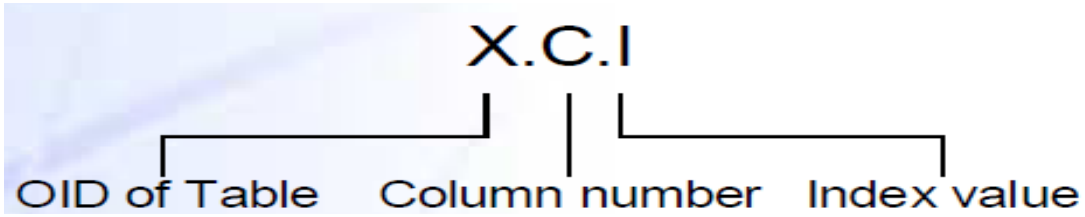
MIB : Nommage des MOs (II)



- **Nommage des entrées d'une table**
  - **Au lieu du 0 final, les valeurs des variable(s) qui forment l'index de la table sont rajoutées.**
  - **Une table est accédée séquentiellement, valeur par valeur et non pas dans son ensemble.**

- Exemples:
  - 1.3.2.5 ⇒ 2
  - 1.3.1.7 ⇒ 7

Indexation d'une table



EXEMPLES:

OID de la Table = 1.3

1.3.1.5 => *5*

1.3.2.5 => *2*

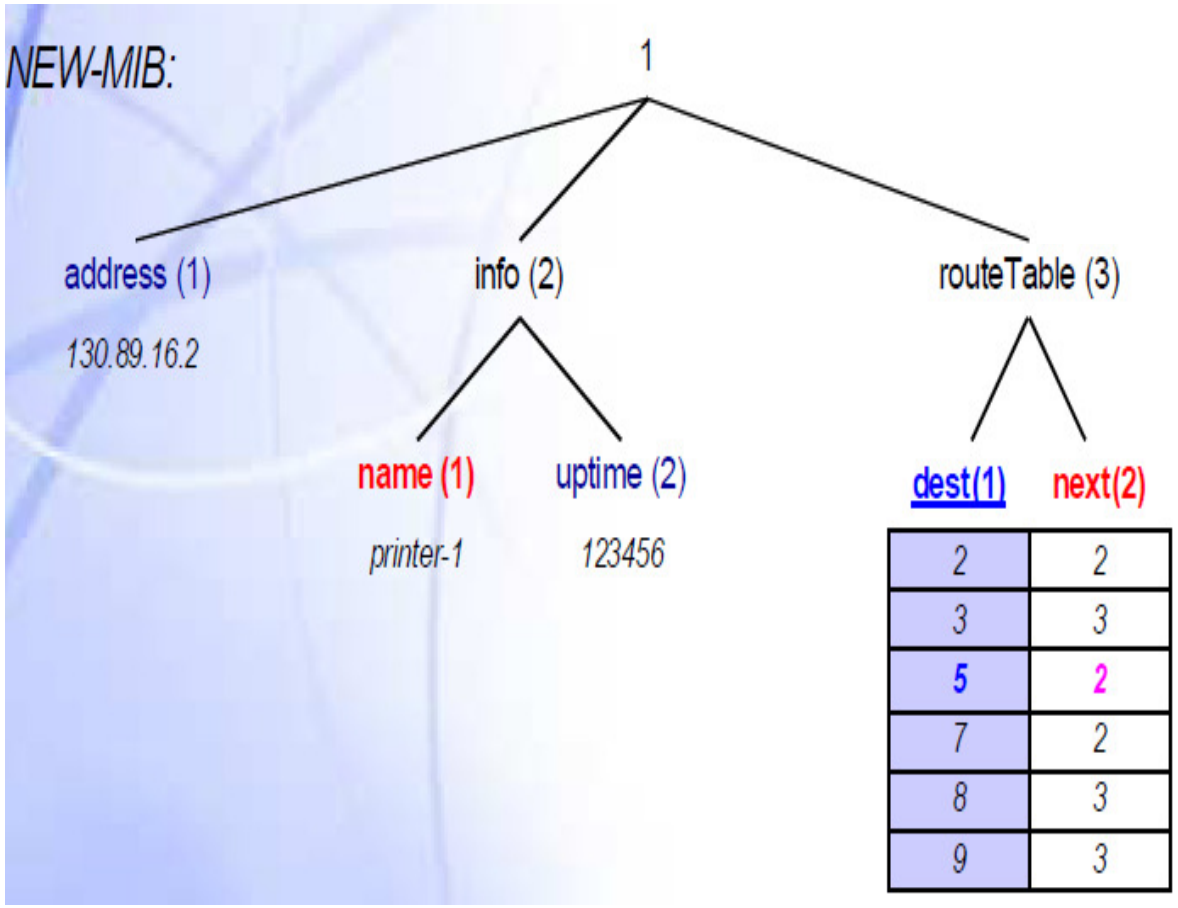
1.3.1.1 => Entrée inexistante

1.3.1.9 => 9

1.3.2.1 => Entrée inexistante

1.3.2.9 => *3*

1.3.2.7 => 2



Indexation d'une table

- Un index n'est pas nécessairement un entier : Ici l'index est une adresse IP

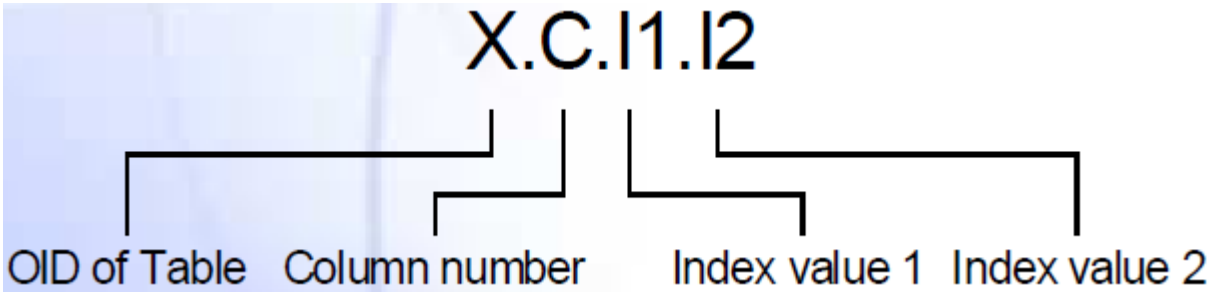
routeTable (3)

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

- *EXEMPLES:* OID de la 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*

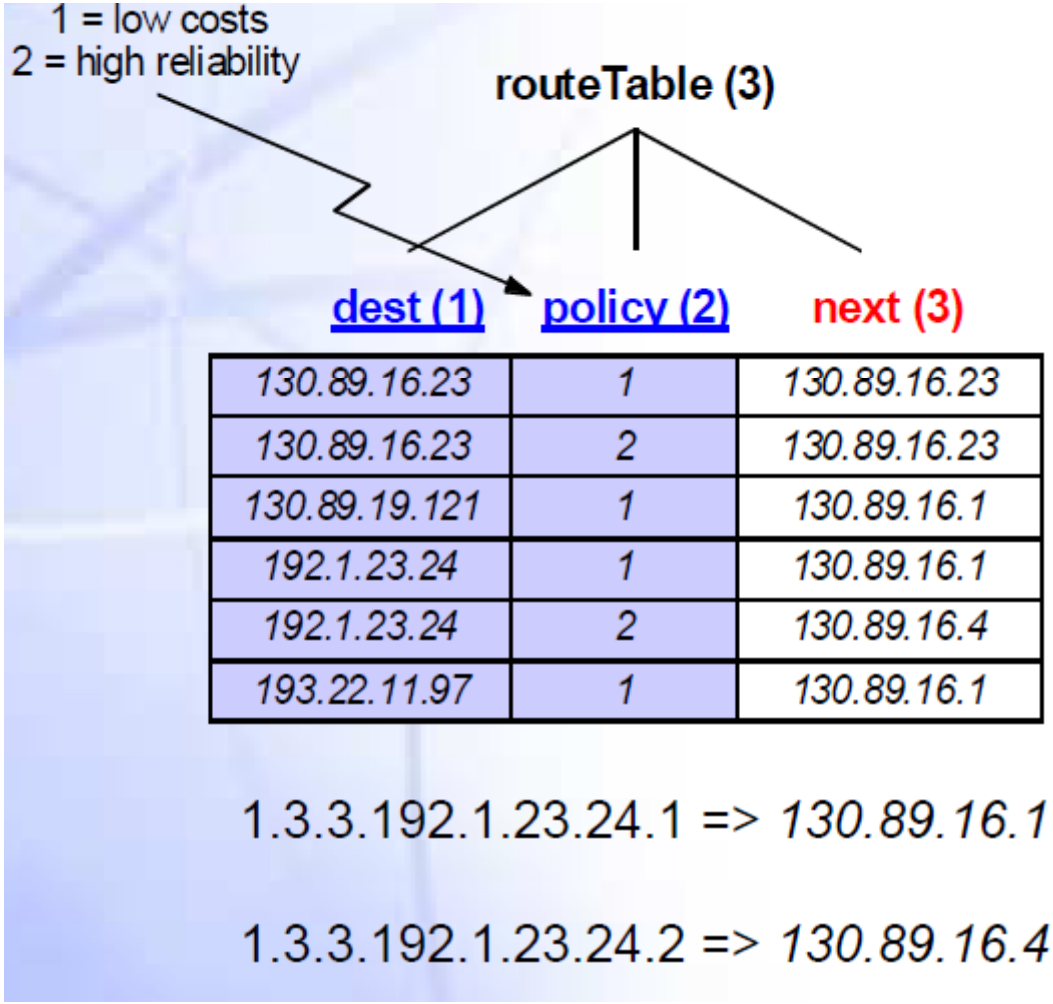
Indexation multiple d'une table

- Un index n'est pas toujours unique et par la suite on aura besoin de définir plus qu'un index pour s'assurer de l'unicité de la combinaison de ces index :
- Dans le cas d'une table de routage un noeud peut être atteint par différents chemins et par la suite l'indexation de la table sur la seule adresse IP destination ne suffit plus.



Indexation multiple d'une table

- Exemple:



SMI

- La SMI (Structure of Management Information) spécifie les règles de définition des 'Managed Objects' (MO) qui sont:
  - Variables typées simples (scalaires); elles peuvent être organisées en tables à 2 dimensions au maximum
  - 'Basés objets' mais pas orientées objets; les opérations offertes sont uniquement la lecture et l'écriture
  - Spécifiés par un sous-ensemble de Abstract Syntax Notation 1 (ASN.1, Version 1988)
- Ces règles sont valables quelque soit le protocole de gestion utilisé.
- Un MO est défini par
  - Type (Syntax), mode d'accès (Access), état de définition (Status), description et un Identificateur unique…

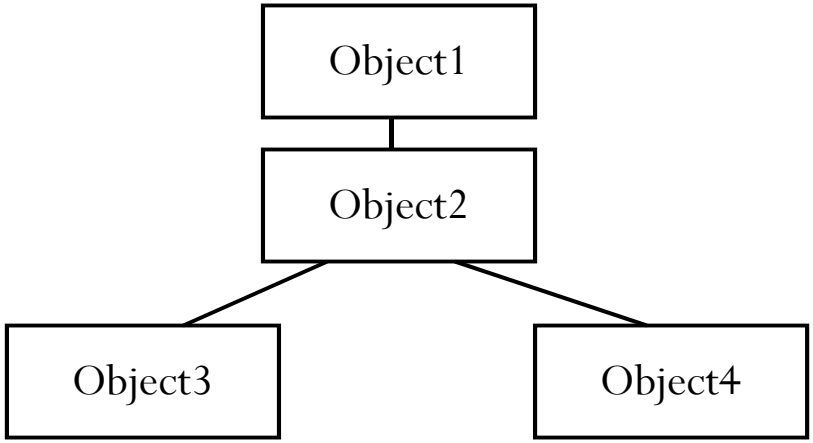
Utilisation de SMI

- SMI (Structure of Management Information)
  - Constitue un moyen normalisé de représenter des informations de gestion :
  - Définition de la structure d'une MIB particulière
  - Définition de chacun des objets de la MIB (syntaxe et valeur)

- Définitions formelles en A.S.N.1 (Abstact Syntax Not.1)

La structure des informations d 'Administration (SMI)

- Un objet peut agréger plusieurs objets :



- object3 **Object Identifier** {object2 1}
- object4 **Object Identifier** {object2 2}
- object2 **Object Identifier** {object1 1}

Définition d'un objet

- Un objet est défini par les champs suivants :
  - **Syntax** : ce champ indique la syntaxe du type d'objet. La syntaxe doit être définie dans les structures SMI.
  - **Max-Access** : ce champ indique le niveau d'accès de cet objet.
  - **Status** : le niveau de support que requiert cet objet.
  - **Description** : contient une description textuelle de l'objet.

- Le nom et **l'identificateur** de l'objet sont écrits respectivement au début et a la fin de la macro de définition de l'objet.

- Les noms de variables MIB sont extraits d 'un espace des noms d'identificateurs d'objets gérés par l 'ISO et l 'UIT-T.

- La responsabilité des règles de nommage est décomposée, à chaque niveau, en domaines
  - Chaque groupe a la responsabilité du choix de certains noms sans avoir à consulter l'autorité supérieure pour chaque décision.

SMI – *Syntax* RS2

SMIv1	SMIv2	
INTEGER	INTEGER Integer32	Nombre entier signé compris entre -2^{31} et $2^{31}-1$
OCTET STRING	OCTET STRING	Chaîne d'octets
OBJECT IDENTIFIER	OBJECT IDENTIFIER	Identificateur unique
-	Unsigned32	Entier naturel compris entre 0 et $2^{32}-1$
Gauge	Gauge32	Entier qui s'incrémente ou se décrémente dans un intervalle
Counter	Counter32 Counter64	Compteur circulaire (0 à $2^{32}-1$ resp. $2^{64}-1$)
TimeTicks	TimeTicks	Durée exprimée en 1/100 seconde (0 à $2^{32}-1$)
IpAddress	IpAddress	4 octets d'une adresse IPv4
Opaque	Opaque	Type ASN.1 quelconque
Network Address	-	Adresse réseau quelconque (non-IP)
-	BitString	Chaîne de bits nommés (ex: liste de flags)

- Definit les types simples des *Managed Objects*

Diapositive 30

RS2 Network Address ????

Rudolf Scheurer; 27/04/2002

SMI– *Access/Status*

- ACCESS/Opérations d'accès d'un MO (SMIv1)
  - ***not-accessible*** – pour la définition des tables
  - ***read-only*** – modifiable uniquement par l'Agent
  - ***read-write*** – modifiable aussi par le Manager
  - ***write-only*** – uniquement accès en écriture

  ➔ Changements avec SMIv2
    - Elimination de « write-only »
    - Nouveau: « ***read-create »*** pour créer des MO dans des tables

- STATUS/Etat de définition d'un MO
  - ***mandatory***– le MO doit être disponible/implémenté
  - ***optional*** – l'implémentation du MO n'est pas nécessaire
  - ***obsolete*** – le MO va disparaître à la prochaine version

  ➔ Changements avec SMIv2
    - « mandatory » remplacé par « ***current »***
    - « optional » a été éliminé

Définition d'un objet scalaire

OBJECT-TYPE:

SYNTAX
- INTEGER
- OCTET STRING
- OBJECT IDENTIFIER
- BITS
- IpAddress
- Integer32
- Counter32
- Counter64
- Gauge32
- TimeTicks
- Opaque
- New Type

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

STATUS
- current
- deprecated
- obsolete

DESCRIPTION ""

- ✓ IpAdress : adresse Internet sur 32 bits.
- ✓ Counter : entier non signé dont la valeur peut évoluer jusqu'à une borne au-delà de laquelle le compteur est remis à zéro.
- ✓ Gauge : entier non signé : qui ne revient pas a 0.
- ✓ TimeTicks : entier positif qui cumule en centième de secondes le temps écoulé depuis une date initiale.
- ✓ Opaque : permet le passage entre une syntaxe ASN.1 et une syntaxe arbitraire.

SMI – Exemples de définition

Définition d'une Adresse (objet scalaire):

address OBJECT-TYPE

SYNTAX *IpAddress*

MAX-ACCESS *read-write*

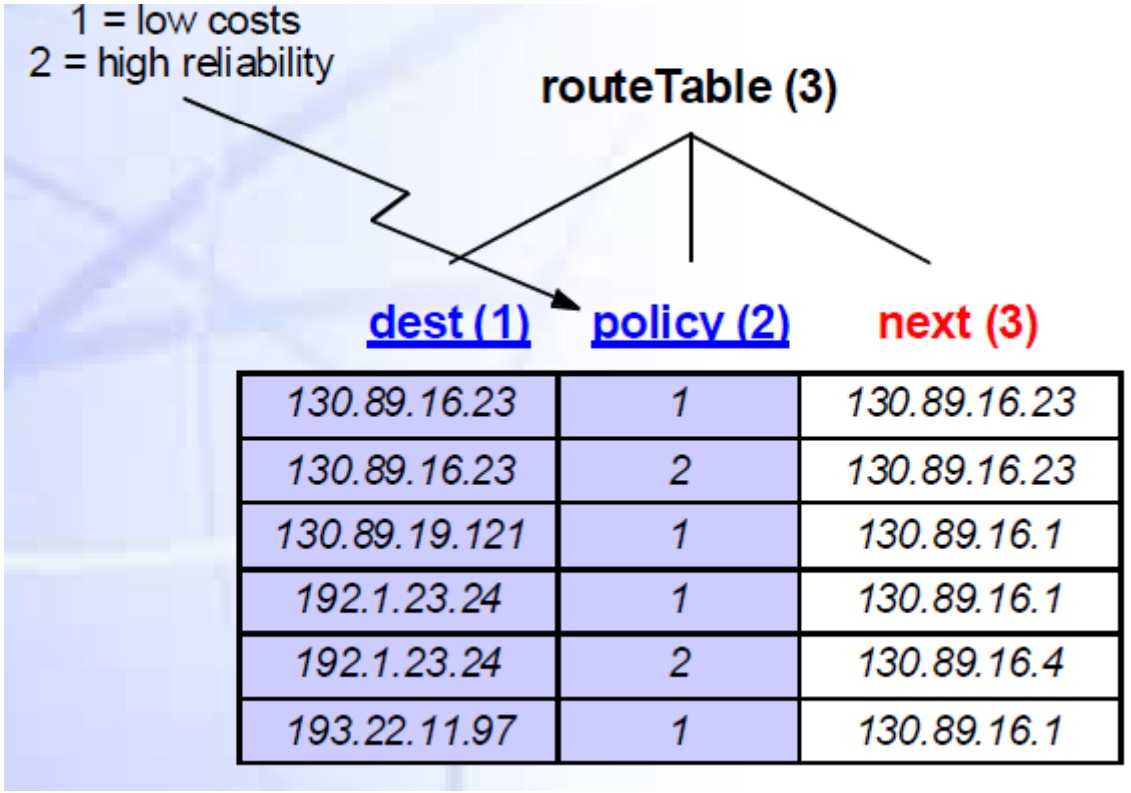
STATUS *current*

DESCRIPTION "The Internet address of this system"

::= {MY-MIB 1}

Définition d'une table

- Principe:
  - Une table de routage est une séquence d'entrée
  - Chaque entrée est composée d'une @source, d'une @dest et d'un critère de choix.



dest (1)	policy (2)	next (3)
130.89.16.23	1	130.89.16.23
130.89.16.23	2	130.89.16.23
130.89.19.121	1	130.89.16.1
192.1.23.24	1	130.89.16.1
192.1.23.24	2	130.89.16.4
193.22.11.97	1	130.89.16.1

Définition de la table de routage

routeTable OBJECT-TYPE
  SYNTAX **SEQUENCE OF routeEntry**
  MAX-ACCESS **not-accessible**
  STATUS **mandatory**
  DESCRIPTION **"This entity's routing table"**
::= {NEW-MIB 3}

routeEntry OBJECT-TYPE
  SYNTAX **ligne**
  MAX-ACCESS **not-accessible**
  STATUS **mandatory**
  DESCRIPTION **"A route to a particular destination"**
  INDEX **{dest, policy}**
::= **{routeTable 1}**

Définition d'une table (suite)

ligne::=

```
SEQUENCE {
        dest    ipAddress,
        policy  INTEGER,
        next    ipAddress}
```

- RouteEntry est une séquence (liste) de deux adresses IP et d'un entier

Définition d'une table (suite)

```
dest            OBJECT-TYPE
    SYNTAX      ipAddress
    ACCESS      read-only
    STATUS      mandatory
    DESCRIPTION "The address of a particular destination"
::= {route-entry 1}

policy          OBJECT-TYPE
    SYNTAX      INTEGER {
                        costs(1) -- lowest delay
                        reliability(2)} -- highest reliability
    ACCESS      read-only
    STATUS      mandatory
    DESCRIPTION "The routing policy to reach that destination"
::= {route-entry 2}

next            OBJECT-TYPE
    SYNTAX      ipAddress
    ACCESS      read-write
    STATUS      mandatory
    DESCRIPTION "The internet address of the next hop"
::= {route-entry 3}
```

Définition de nouveaux types

- Utilisation des conventions textuelles (TEXTUAL CONVENTIONS) pour raffiner la sémantique des types déjà existants.

- Exemple:

```
RunState ::= TEXTUAL CONVENTION
    STATUS          mandatory
    DESCRIPTION     "..."
    SYNTAX          INTEGER{
                        running(1)
                        runable(2)
                        waiting(3)
                        exiting(4)
                    }
```

Groupe d'objets

- La construction d'un groupe d'objets permet de regrouper un ensemble de types d'objets ayant une caractéristique en commun.

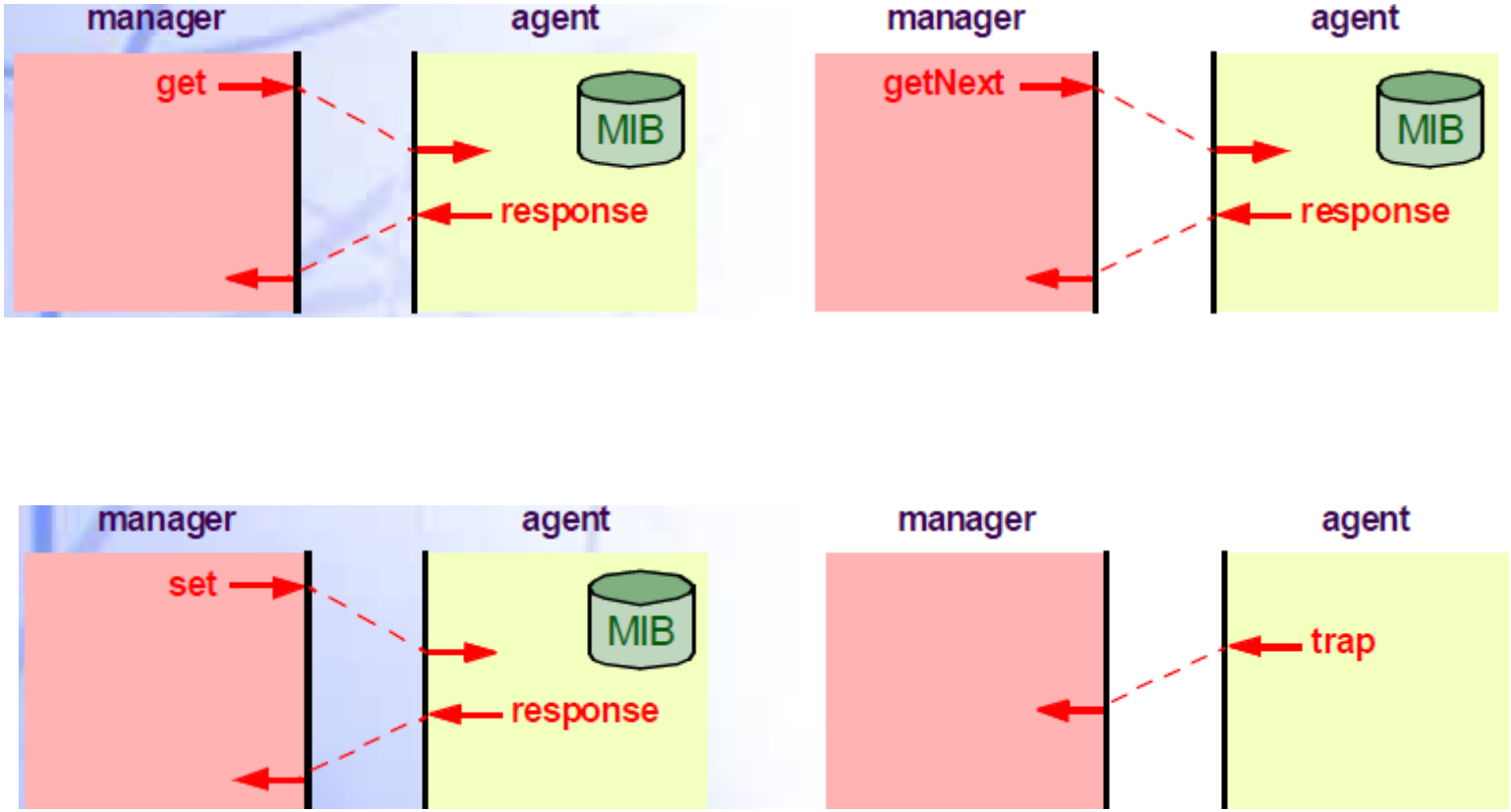
- Exemple:

```
myGroup3      OBJECT-GROUP
   OBJECTS    { address, name, uptime }
   STATUS     mandatory
   DESCRIPTION      "The collection of scalar objects."
   ::= { myGroups 3 }
```

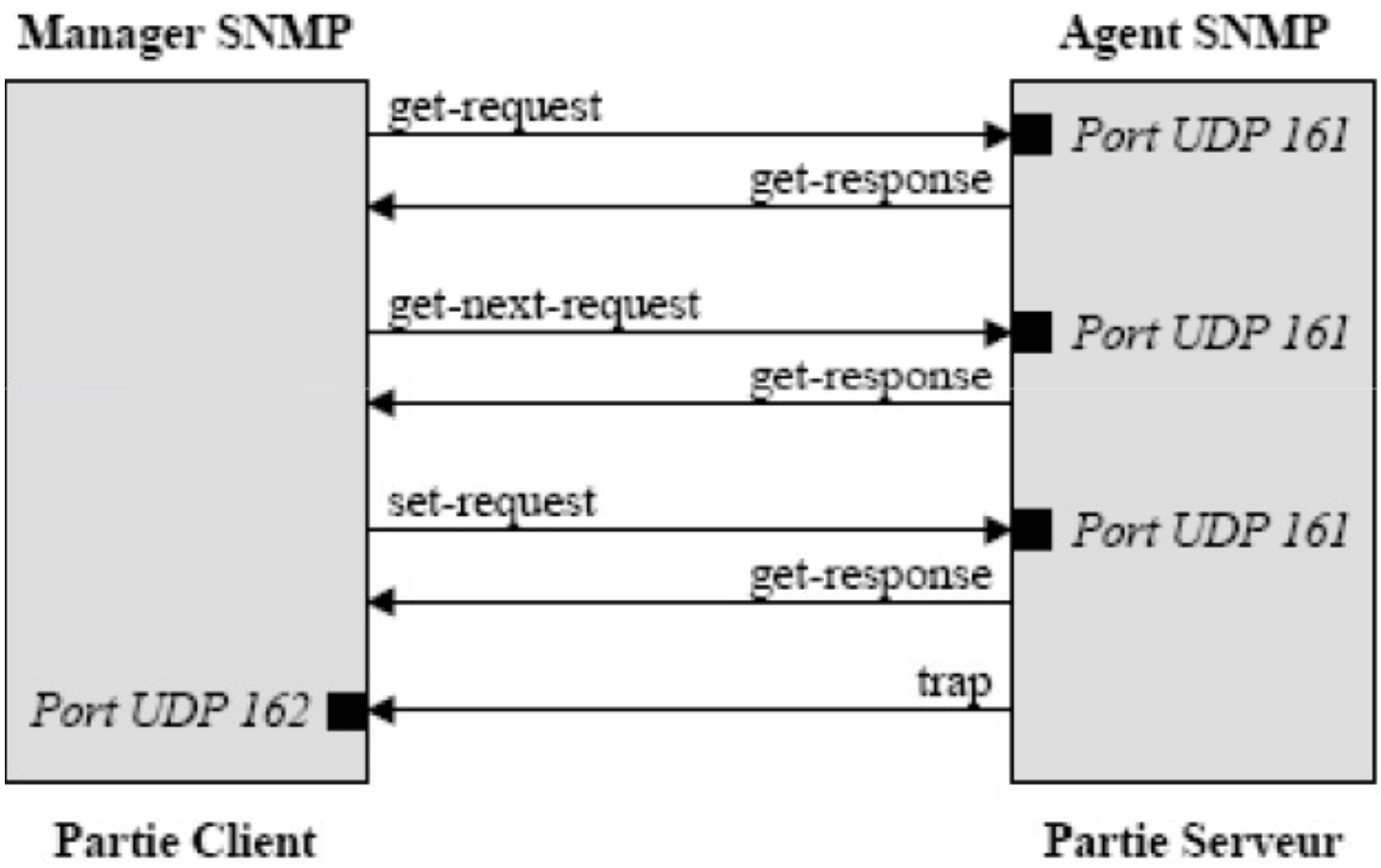
Opérations du protocole SNMP

- **LECTURE :** lit la valeur d 'une variable
  - *get-request, get-response*
- **ECRITURE :** affecte une valeur à une variable
  - *set-request*
- **PARCOURS :** pour connaître les variables effectivement gérées par un nœud
  - *get-next-request, get-response*
- **NOTIFICATIONS :** pour signaler un événement extraordinaire à un gestionnaire
  - *trap*

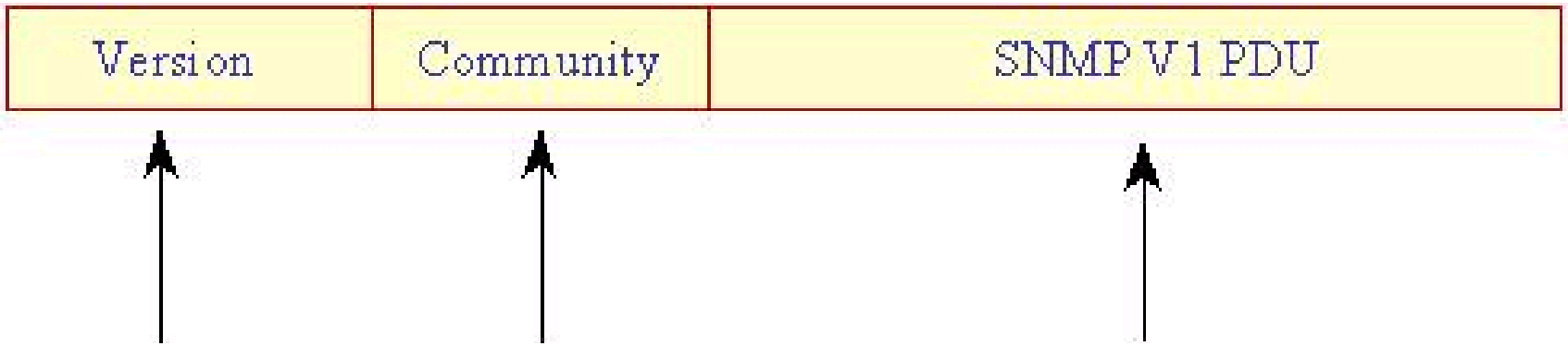
Aperçu


manager
agent
get
MIB
response
manager
agent
getNext
MIB
response
manager
agent
set
MIB
response
manager
agent
trap

Modèle Client / Serveur, N° des ports



Format général du Message SNMP



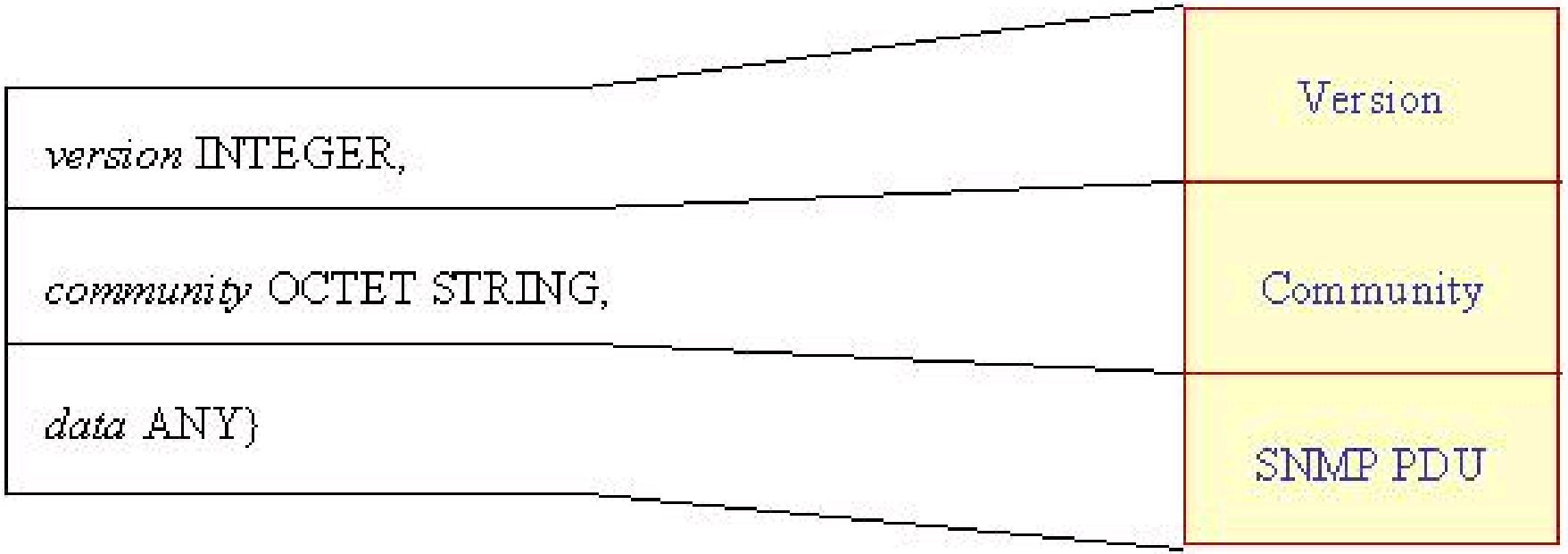
- SNMP community = Un ensemble d'administrateurs autorisés à utiliser l'agent
- Chaque communauté est définie en utilisant un nom unique
- Les administrateurs doivent préciser le nom de la communauté dans les requêtes SNMP

Définition ASN.1 du Message

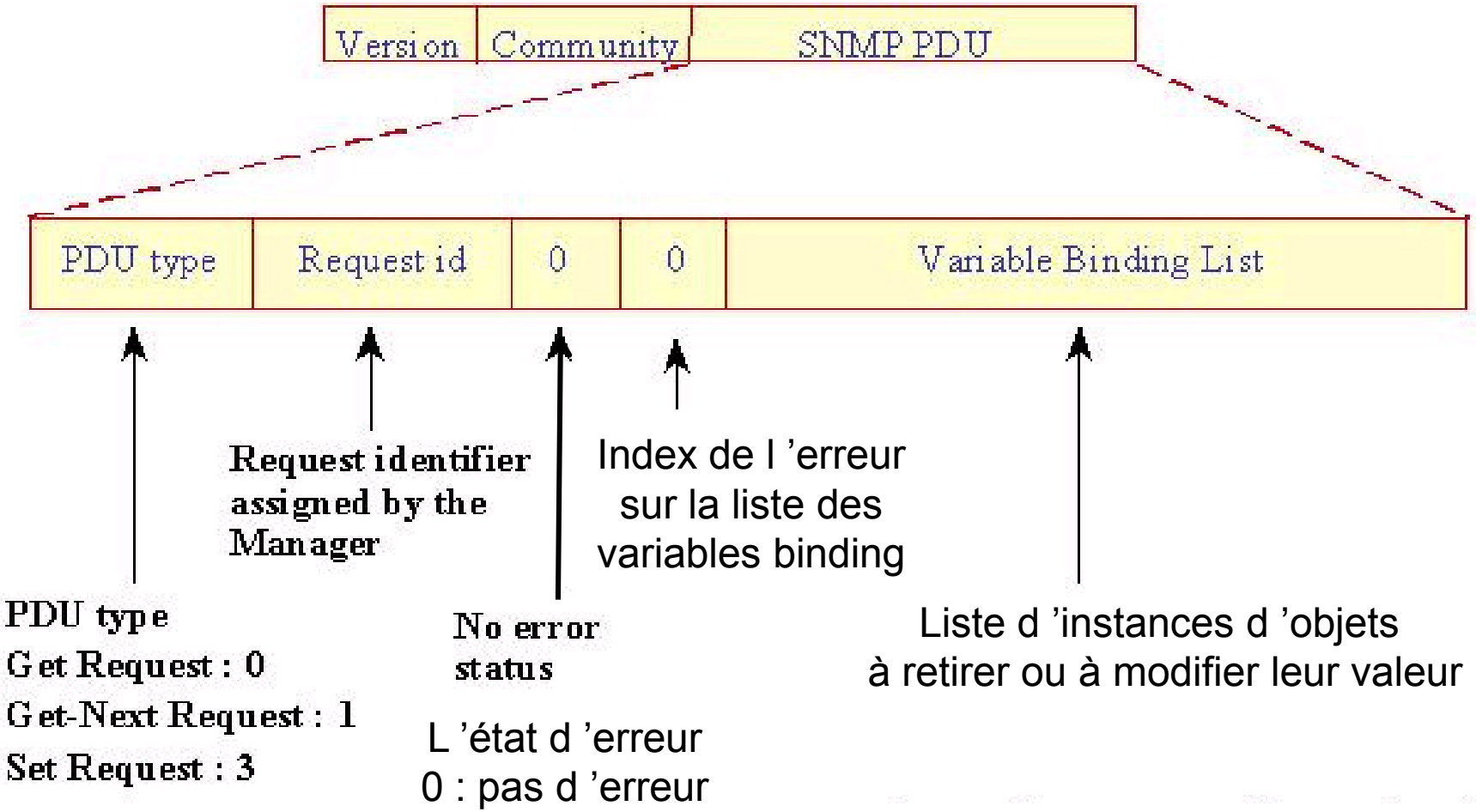
RFC1157-SNMP DEFINITIONS ::= BEGIN

IMPORTS *ObjectName*, *ObjectSyntax*, ... FROM *RFC1155-SMI*;

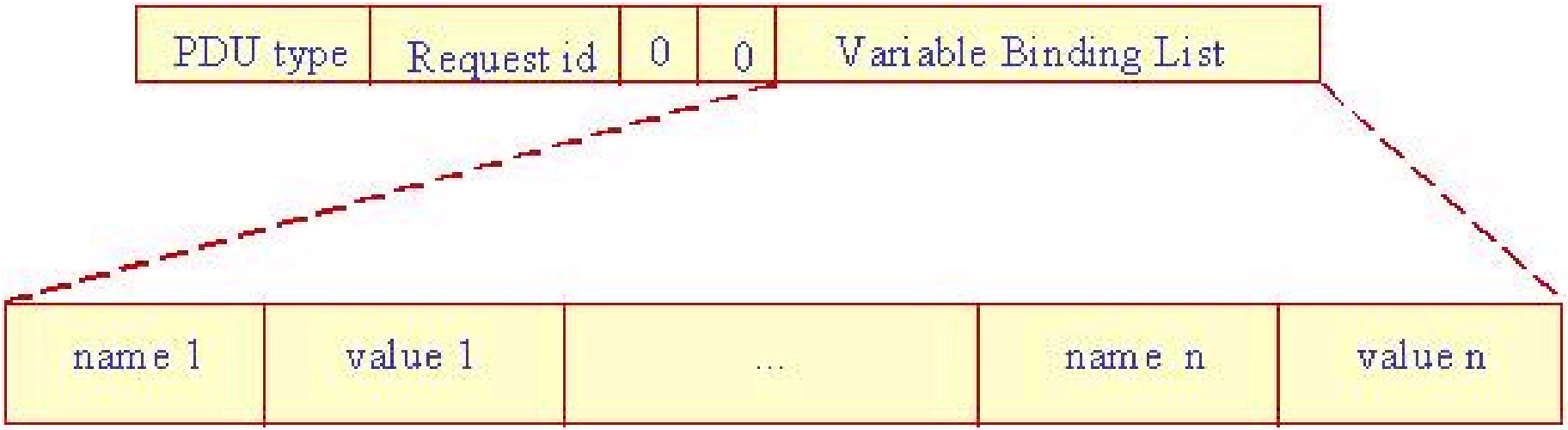
Message ::= SEQUENCE {



Format des Get, Get-Next et Set



Format de Variable Binding List



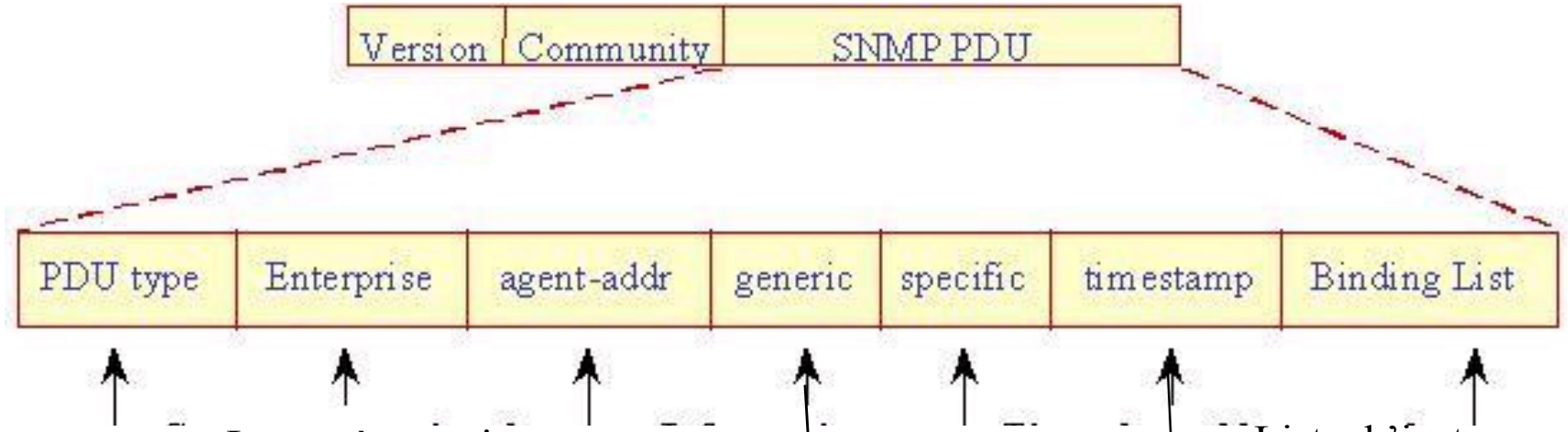
VarBind ::= **SEQUENCE** {

name ObjectName,

value ObjectSyntax }

VarBindList ::= **SEQUENCE OF** *VarBind*

Format de Trap



Type =4 pour trap

Le système qui a généré la trap (sysObjectID du groupe system)

L 'adresse IP de l 'agent

Information à propos de la nature de l 'événement

Information à propos de la nature de trap

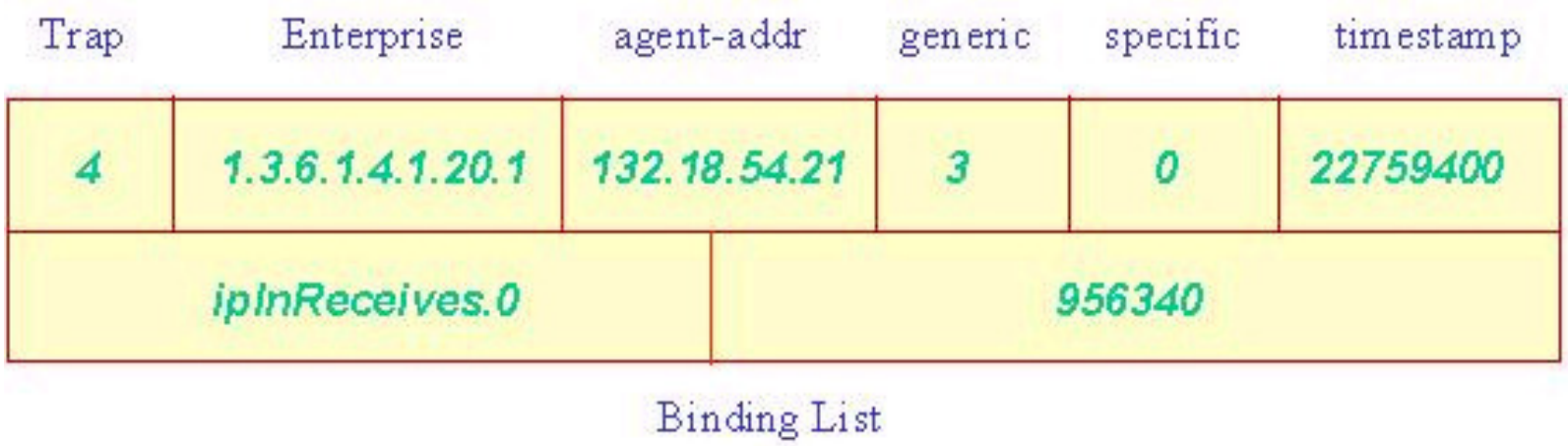
Le temps entre la dernière initialisation de l 'agent et la génération de la trap

Liste d 'instances d 'objets à retirer ou à modifier leurs valeurs

Le champ “Generic”

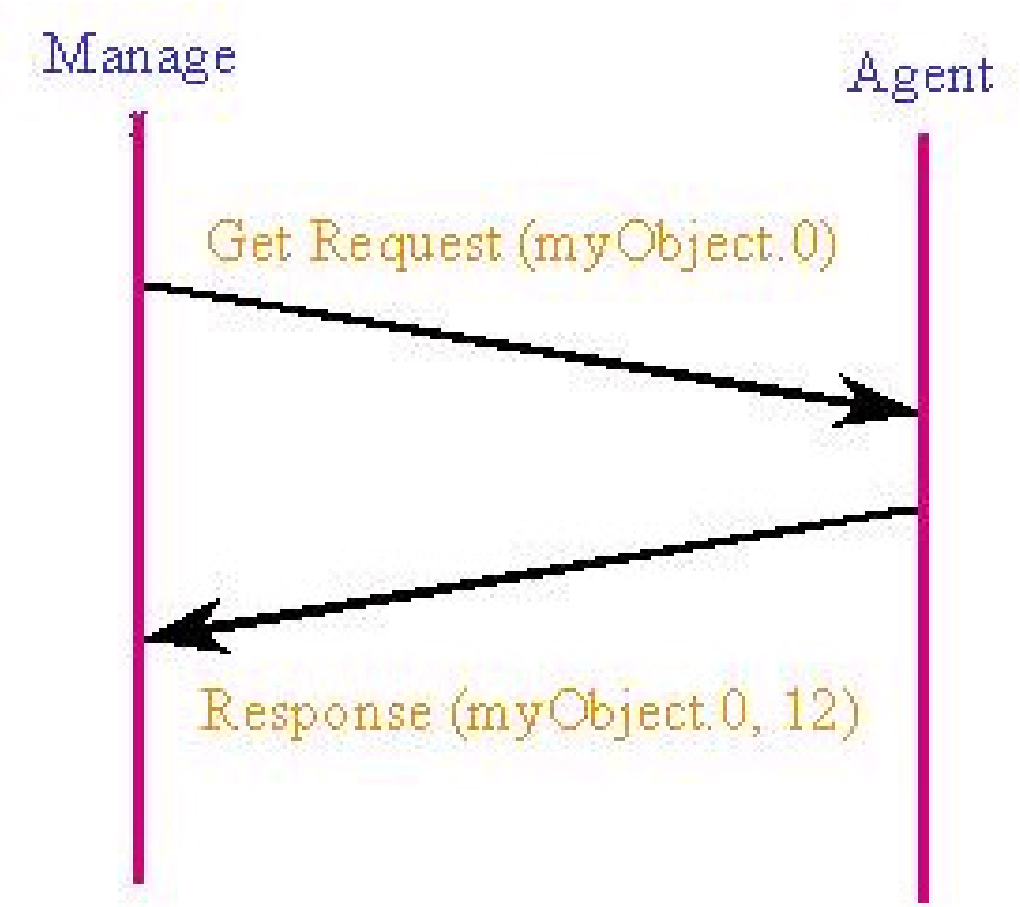
- Le champ “Generic” peut prendre une des valeurs suivantes :
  - **coldStart (0) :** Une réinitialisation inatendue due à une défaillance.
  - **warmStart (1)** : Une défaillance mineur
  - **linkDown (2)** : Une défaillance survenue sur une interface physique.
  - **linkUp (3)** : Une interface devient active.
  - **authenticationFailure (4) :** L’agent a reçu un message avec une authentification impropre
  - **egpNeighborLoss (5) :** Un routeur voisin utilisant EGP (External Gateway Protocol) est déclaré comme étant non focntionnel
  - **enterpriseSpecific (6) :** L’événement relatif à “enterprise-specific” est survenu

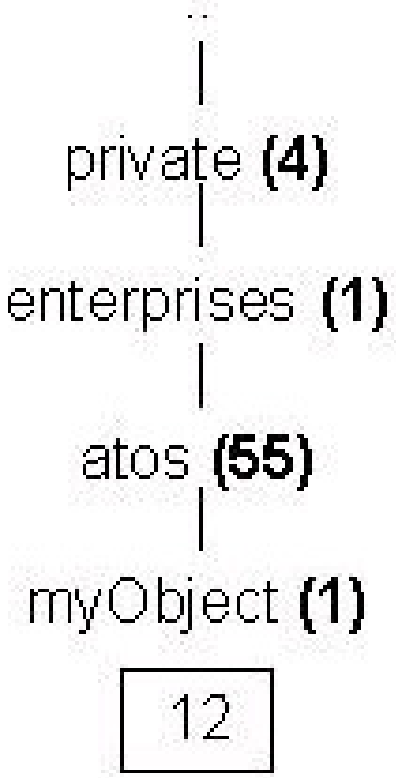
Exemple de Trap



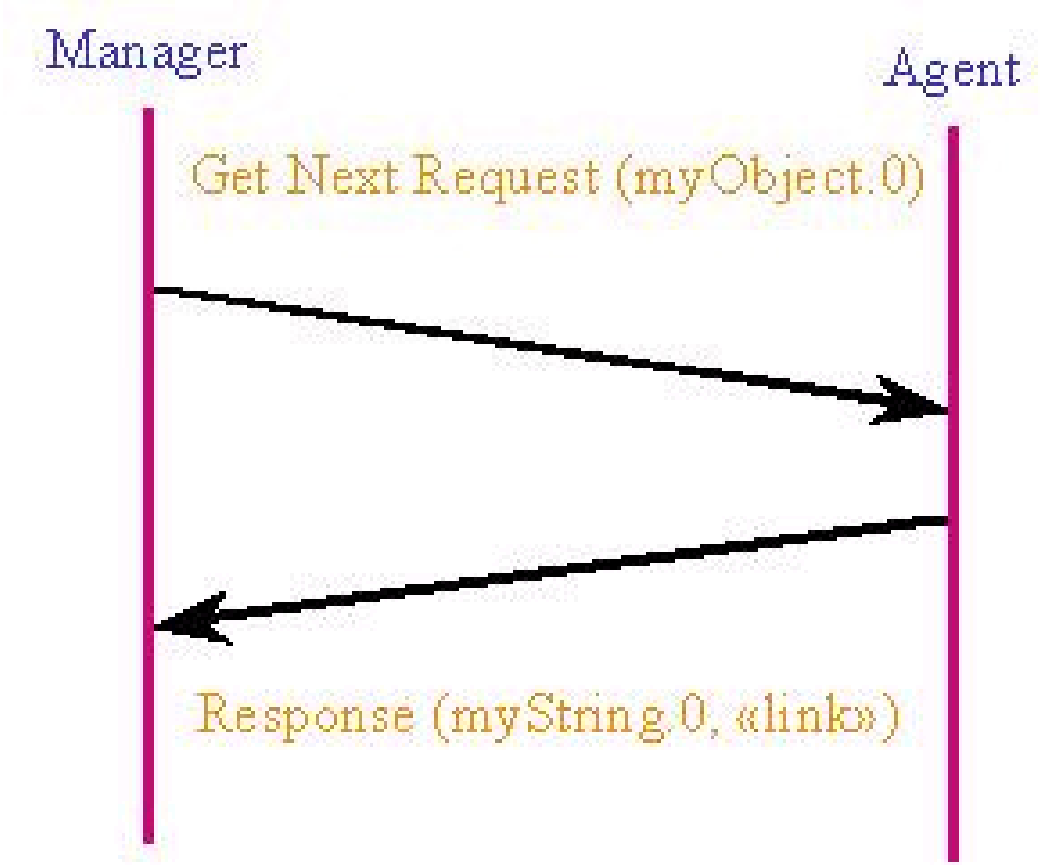
- L ’adresse IP de agent émetteur : 132.18.54.21
- L ’objet concerné par la trap est : 1.3.6.1.4.1.20.1 (MIB privée)
- Type de trap : link up (generic=3)
- Indication : le nombre de paquets reçus est 956340
- La dernière réinitialisation de l’agent : 6 heures passées.

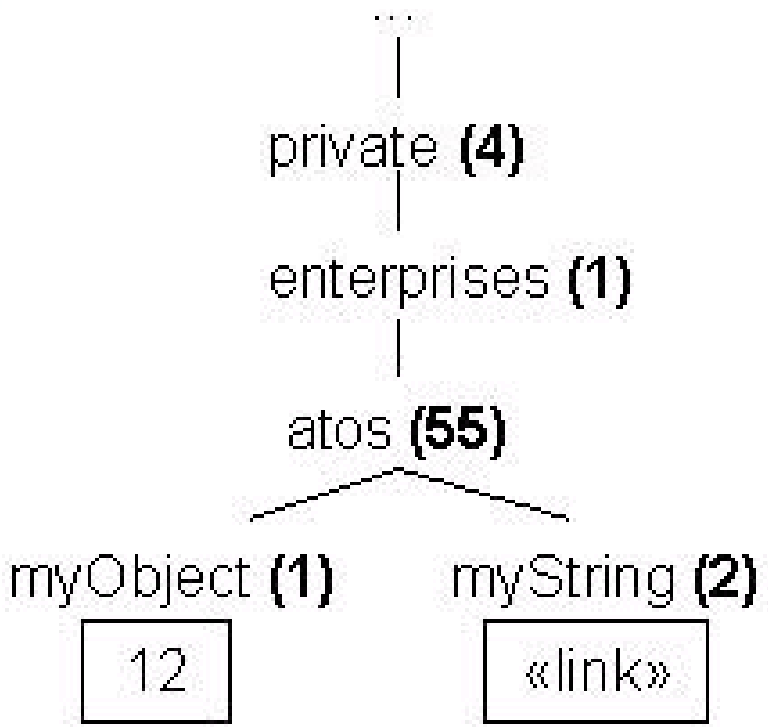
La requête GET



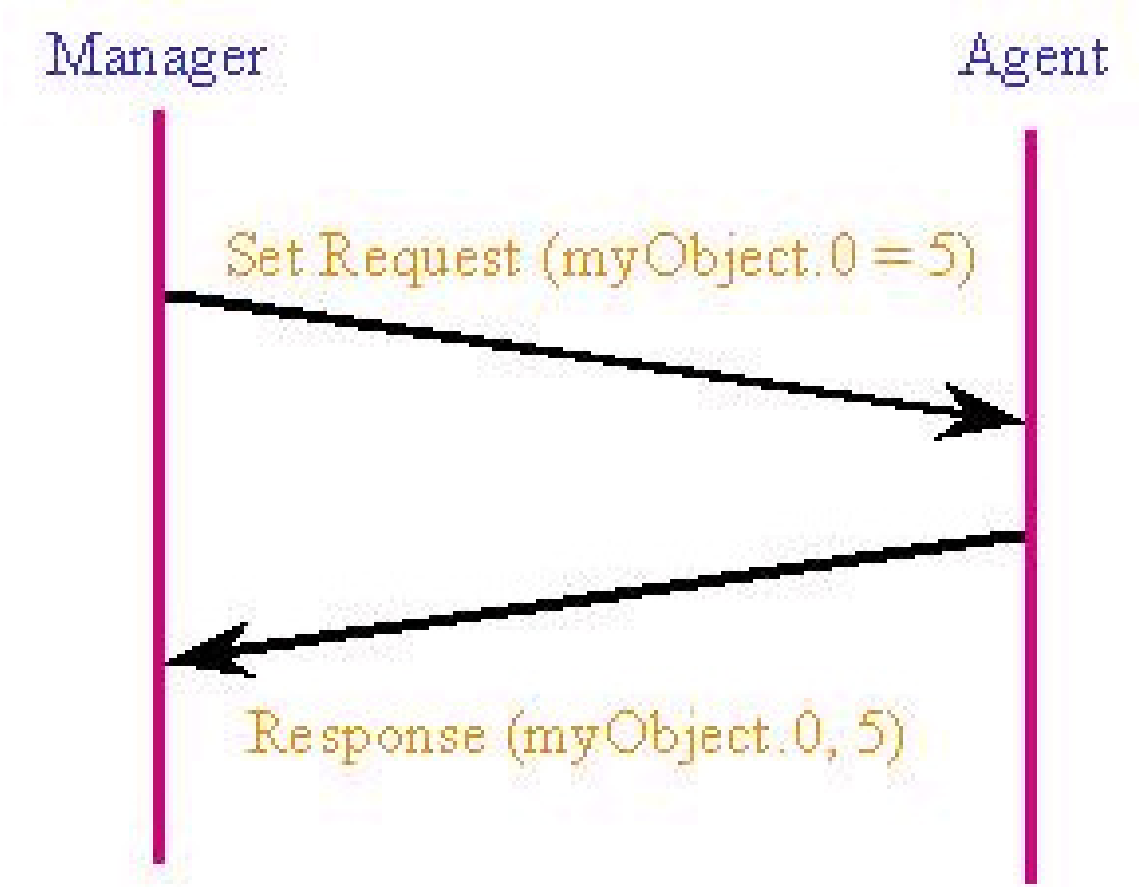


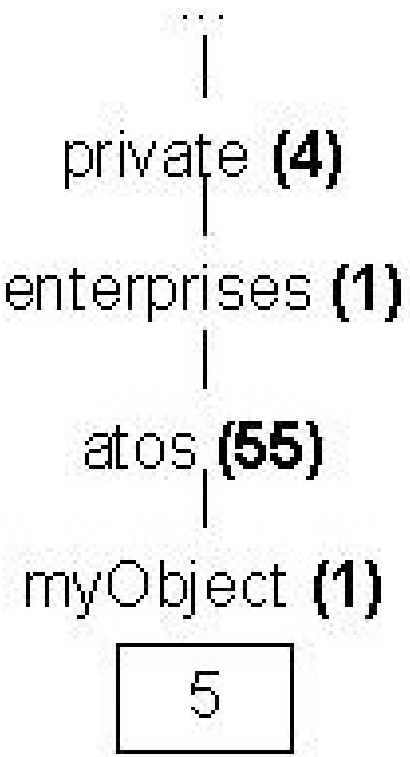
La requête GETNextRequest



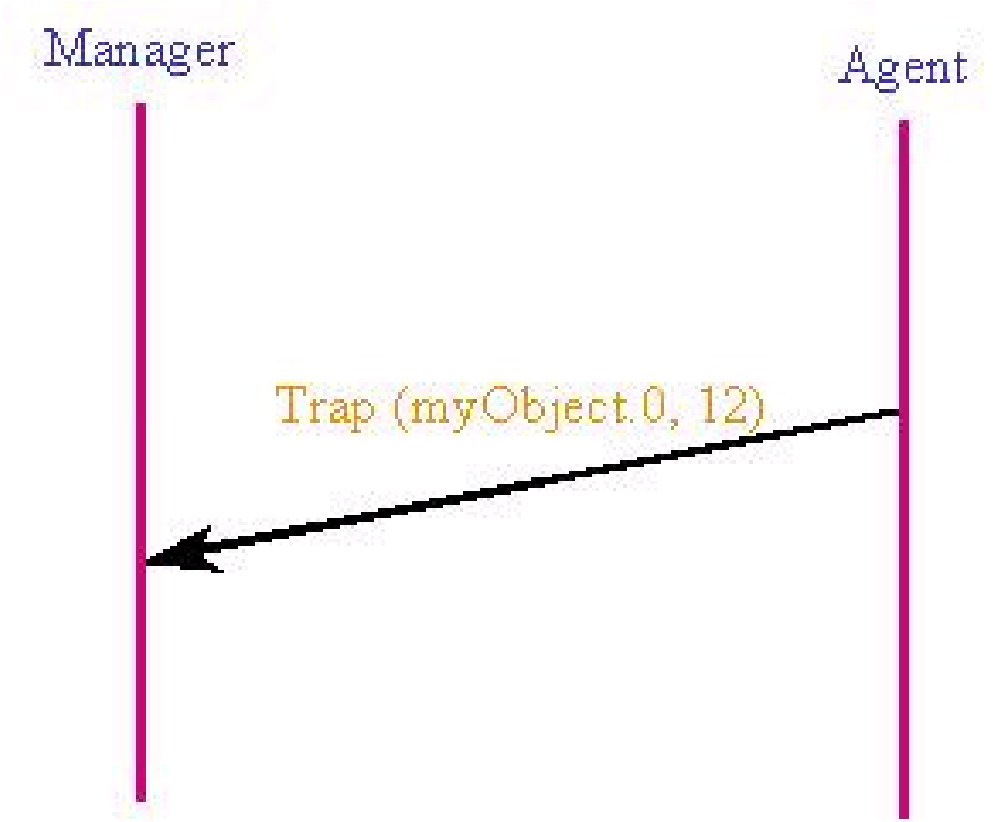


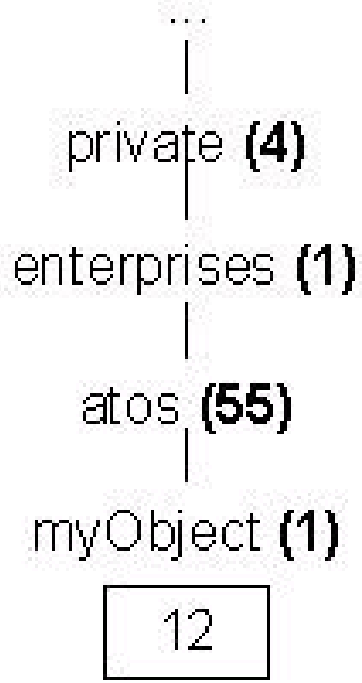
La requête Set





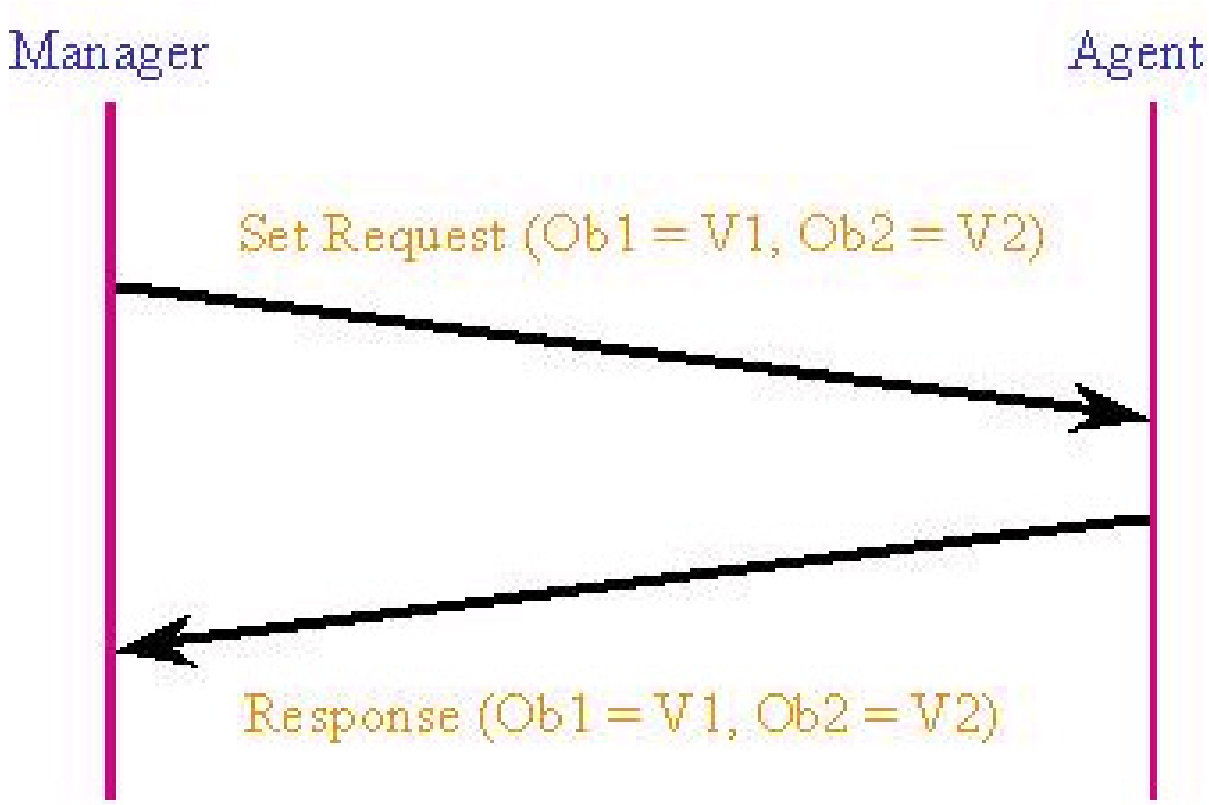
La notification TRAP



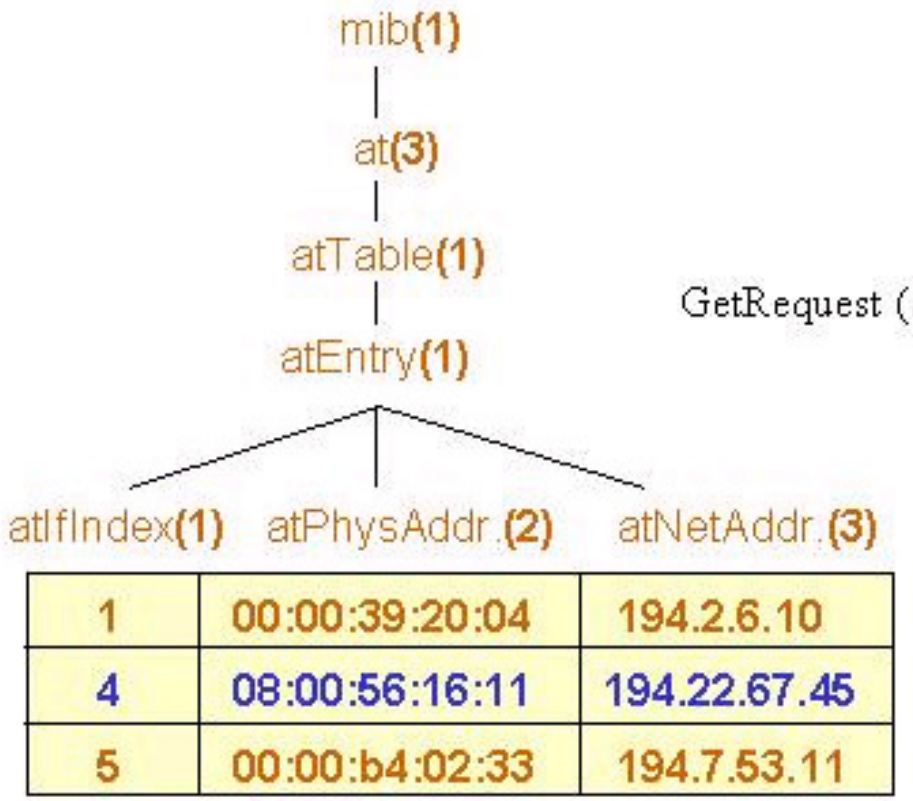


Les requêtes multiples

- Les requêtes Get, Get Next et Set Request peuvent préciser plusieurs objets à lire ou à modifier leurs valeurs.



Exemple de Get Request



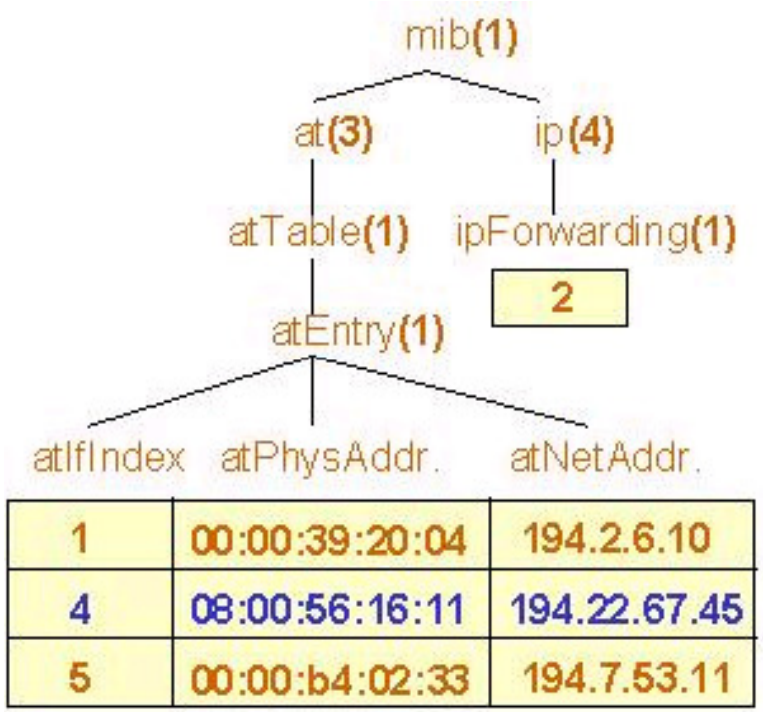
1	00:00:39:20:04	194.2.6.10
4	08:00:56:16:11	194.22.67.45
5	00:00:b4:02:33	194.7.53.11

To get the second row

GetRequest (atIfIndex.4, atPhysAddress.4, atNetAddress.4)

Response (atIfIndex.4 = 4,
atPhys.4 = 08:00:56:16:11,
atNet.4 = 194.22.67.45)

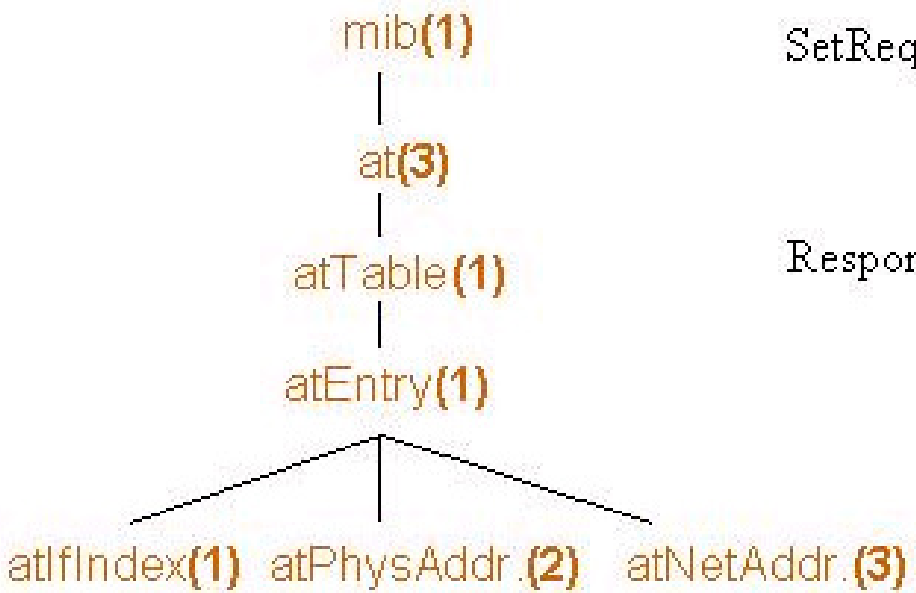
Exemple de GetNext Request



GetNextRequest (atIfIndex.1, atPhys.1, atNet.1)

Response (atIfIndex.4 = 4,
atPhys.4 = 08:00:56:16:11,
atNet.4 = 194.22.67.45)

Exemple de Set Request



1	00:00:39:20:04	194.2.6.10
4	00:00:77:b1:45	194.22.67.45
5	00:00:b4:02:33	194.7.53.11

SetRequest (atPhysAddress.4 = 00:00:77:b1:45)

Response (atPhysAddress.4 = 00:00:77:b1:45)