Administration de réseaux informatiques : protocole SNMP

par **Olivier WILLM**
Responsable de l'exploitation des systèmes décentralisés, Air France

Contexte

Le système d'information et les réseaux sont aujourd'hui un élément critique de la compétitivité de l'entreprise. De mauvaises performances de ceux-ci se déclinent en pertes de productivité et les indisponibilités de service se traduisent en pertes de vente, irrégularités d'exploitation ou interruption de la production. La qualité du service global rendu aux utilisateurs doit être garantie, ce qui implique la maîtrise d'un ensemble complexe de systèmes, réseaux, middleware et applications. Cela doit se faire à un coût minimal. C'est l'enjeu des activités liées à l'administration de réseaux (et plus généralement du système d'information) que d'offrir cette maîtrise au meilleur coût. Les éléments essentiels à maîtriser sont :

*— les **coûts** : à la fois de développement et d'intégration, mais aussi et surtout les coûts opérationnels (maintenance, de support, d'évolutions, etc.) qui sont souvent négligés mais constituent une partie importante de la facture finale ;*

*— la **qualité de service** (QoS : Quality of Service), qui se décline sur plusieurs critères, du point de vue de l'utilisateur final :*

- *la disponibilité,*
- *le temps de réponse,*
- *la fiabilité,*
- *la sécurité (qui est un sujet en soi) ;*

*— la **réactivité** :*

- *devant les évolutions technologiques qu'il est nécessaire de suivre, ce qui implique de faire des choix technologiques évolutifs,*

• face aux besoins de croissance ou d'évolution, qui ne sont pas toujours faciles à maîtriser (un exemple classique est celui d'une application bien adaptée à un fonctionnement en réseau local mais qui ne sera pas adaptée à un fonctionnement sur réseaux étendus, si elle utilise de nombreux échanges protocolaires entraînant des temps de réponse élevés lorsque le réseau présente une latence non négligeable),

• face aux besoins de changements (déménagements, restructurations, nouveaux besoins, etc.),

• face aux problèmes qui se présentent nécessairement.

Dans ce contexte, l'administration de réseaux représente toutes les activités qui permettent de ***planifier***, ***superviser*** *et* ***exploiter*** *les réseaux, en respectant les contraintes de coût et de QoS.*

Problématique et mise en œuvre

Il est rare qu'une entreprise ait à sa disposition, sur site, des experts capables de diagnostiquer et d'intervenir de manière efficace sur tous les problèmes potentiels. Il est donc nécessaire de mettre en place des outils permettant à des spécialistes de diagnostiquer et d'intervenir à distance.

Par ailleurs, la complexité des réseaux augmente de manière combinatoire avec le nombre de systèmes à gérer. Enfin, l'administrateur d'un réseau, de nos jours, doit souvent gérer une hétérogénéité complexe, au niveau des protocoles et standards d'interopérabilité, des technologies et de la manière dont elles sont mises en œuvre, des systèmes d'exploitation des équipements, des flux de trafic, etc.

Les solutions doivent donc être évolutives et souples, de manière à permettre de prendre en compte cet ensemble. Il est important de ne pas tomber dans les deux excès : vouloir tout traiter avec le même outil ou « faire son marché » sans démarche globale et réfléchie concernant les moyens à mettre en œuvre. Dans le premier cas, on risque de consacrer beaucoup de temps et d'argent à mettre en œuvre des solutions décevantes. Dans le second, on risque de passer son temps à acquérir de nouveaux produits sans les utiliser efficacement et sans qu'ils permettent d'avoir une vision globale de ce qui se passe sur le réseau. Un compromis est donc nécessaire entre des solutions « génériques » et standard, relativement lourdes à mettre en œuvre mais qui apportent de grands bénéfices une fois en place, et entre des solutions permettant de répondre à un besoin bien ciblé.

En particulier, il est souhaitable d'appuyer autant que possible son administration de réseaux sur des standards : le protocole SNMP (Simple Network Management Protocol) est actuellement la technologie de base qui permet d'administrer un réseau TCP/IP, et plus généralement les réseaux informatiques (systèmes, applications, etc.). Même si ce n'est pas la technologie permettant de résoudre tous les problèmes, il convient de la mettre en œuvre dès que l'on a affaire à un réseau TCP/IP de quelque importance, complétée par d'autres si nécessaire. Les réseaux de télécommunications (opérateurs) s'appuient quant à eux sur CMIP (Common Management Information Protocol) et sur l'architecture TNM (Telecommunications Network Management).

Il est par ailleurs important de mettre en œuvre un projet global, qui ne soit pas seulement technique, mais qui comprenne aussi les aspects essentiels que sont l'organisation de l'activité et la gestion des ressources humaines.

Sigles et abréviations

API	Application Progamming Interface	CIM	Common Information Model
APM	Application Performance Measurement	CLNS	Connexion-Less Network Service
ARP	Address Resolution Protocol	CMIP	Common Management Information Protocol
ASN	Abstract Syntax Notation	CONS	Connexion Oriented Network Service
BER	Basic Encoding Rules	CPU	Central Processing Unit
CBC	Cipher Block Chaining	DES	Data Encryption Standard
DMI	Desktop Management Interface	PDU	Protocol Data Unit

Sigles et abréviations			
DMTF	Distributed Management Task Force	PPP	Point-to-Point Protocol
DNS	Domain Name System	PTOPO	Physical Topology
DoD	Department of Defense	QoS	Quality of Service
DS	Digital Signal	RFC	Request For Comments
EGP	Exterior Gateway Protocol	RMON	Remote Monitoring
FDDI	Fiber Distributed Data Interface	SHA	Secure Hash Standard
GDMO	Guidelines for the Definition of Managed Objects	SLA	Service Level Agreement
HMAC	keyed-Hash Message Authentication Code	SLM	Service Level Management
HTML	HyperText Markup Language	SMI	Structure of Management Information
HTTP	HyperText Transfer Protocol	SMON	Switched RMON
IANA	Internet Assigned Numbers Authority	SNA	Systems Network Architecture
ICMP	Internet Control Message Protocol	SNMP	Simple Network Management Protocol
IETF	Internet Engineering Task Force	SONET	Synchronous Optical Network
IP	Internet Protocol	TCP	Transport Control Protocol
IPX	Internet Protocol eXtension	TFTP	Trivial File Transfer Protocol
ISDN	Integrated Services Digital Network	TNM	Telecommunications Network Management
LAPB	Link Access Procedure Balanced	UDP	User Datagram Protocol
MAC	Media Access Control	VACM	View-based Access Control Model
MAU	Medium Attachment Unit	VLAN	Virtual Local Area Network
MD5	Message-Digest Algorithm	WBEM	Web Based Enterprise Management
MIB	Management Information Base	WMI	Windows Management Interface
MTTR	Mean Time To Repair	XML	eXtensible Markup Language
NMVT	Network Management Vector Transport	XMP	X/Open Management Protocols API
OID	Object Identifier	PC	Personal Computer
PC	Personal Computer		

1. Mise en œuvre

Définitions

L'ISO (International Standard Organization) a regroupé les activités d'administration de réseaux en cinq domaines fonctionnels :

— la gestion des anomalies ;
— la gestion de la comptabilité ;
— la gestion des performances ;
— la gestion des configurations ;
— la gestion de la sécurité.

Ces activités sont aussi communément classées de la façon suivante :

— la **supervision** (*monitoring*) consiste à surveiller les systèmes et récupérer les informations sur leur état et leur comportement, ce qui peut être fait par interrogation périodique (*polling*) ou par remontée non sollicitée d'information de la part des équipements de réseau eux-mêmes. Cela recoupe plutôt les activités appartenant à la « gestion des anomalies » et la « gestion des performances » de l'ISO ;

— l'**administration** désigne plus spécifiquement les opérations de contrôle « à froid » du réseau, avec la gestion des configurations, de la sécurité, etc. ;

— l'**exploitation** désigne l'ensemble des activités qui permettent de traiter les problèmes opérationnels sur le réseau, ce qui recouvre la supervision, mais aussi la maintenance, le support et l'assistance technique.

1.1 Outils

1.1.1 Outils locaux

La plupart des outils mis en œuvre sont prévus pour travailler sur un modèle client/serveur, ce qui permet des interventions à distance. Toutefois, il est toujours utile, et parfois indispensable, d'utiliser des outils de diagnostic locaux tels que ping, traceroute, nslookup... qui font partie de la « trousse de premiers secours » de tout administrateur sur site. De même, des outils tels qu'un PC portable connecté en émulation VT100 sur le port console d'un équipement ou un testeur de câblage sont parfois nécessaires, ainsi qu'un analyseur de protocole (hardware ou logiciel), dans le cas où le réseau est inaccessible à distance ou si les outils de téléadministration ne sont pas disponibles.

1.1.2 Plates-formes d'administration

1.1.2.1 Définition

Les plates-formes d'administration de réseaux sont – en principe – aux applications d'administration ce que sont les systèmes d'exploitation aux applications, à savoir un système offrant :

— des services de base et des outils qui donnent à l'utilisateur un environnement homogène à partir duquel il lancera les applications dont il a besoin pour travailler ;

Figure 1 – HP/Openview

— un certain nombre d'API (Application Programming Interface) permettant au développeur de réaliser des applications qui tireront profit des possibilités de la plate-forme, sans qu'il ait besoin de « réinventer la roue ».

Dans le principe, l'idée est séduisante car elle permet de capitaliser ses efforts sur un produit de base autour duquel on bâtira une solution homogène, en s'appuyant sur des produits complémentaires de l'éditeur ou d'autres fournisseurs. Toutefois, la solution demande des efforts d'intégration non négligeables car les produits tiers ne sont pas toujours aussi bien « intégrés » qu'on le souhaiterait et il se présente bien souvent des problèmes à la mise en œuvre, ne serait-ce qu'au niveau des grilles de compatibilité de versions. Par ailleurs, les plates-formes constituent souvent une grosse « boîte à outils » qui nécessite du temps pour être maîtrisée et en tirer le meilleur profit. Mais cela en vaut la peine dès lors que le réseau à administrer est suffisamment grand.

1.1.2.2 Services de base

Découverte et gestion de la topologie du réseau

Actuellement, toutes les plates-formes savent découvrir de manière automatique les machines IP du réseau, et dessiner la topologie de niveau 3 avec les interconnexions, les routeurs, les réseaux IP. Elles savent aussi découvrir la topologie de niveau 2, avec plus ou moins de bonheur si on a mis en place des VLAN (Virtual Local Area Network). Enfin, elles maintiennent à jour une base de données des objets avec la description des objets et de leurs attributs (interfaces, adresses, est-ce un hub, un pont, un routeur, un serveur Web...).

Gestion des événements

Les notifications envoyées par les équipements ainsi que les alarmes générées par la plate-forme elle-même font l'objet d'un archivage dans des fichiers plats ou une base de données. Elles sont affichées grâce à une application graphique ou triées par type, apparaissent avec des codes de couleur correspondant aux sévérités, etc. Il est possible de faire des filtres ou des corrélations plus ou moins évolués, de manière à limiter le nombre d'événements et à améliorer la visibilité des plus significatifs. Par ailleurs, ces événements peuvent être récupérés par les applications, comme par exemple celle qui affiche les vues topologiques de réseau, qui affichera les éléments avec des couleurs différentes selon leur état (figure **1**). Il est enfin possible d'automatiser le lancement d'actions sur réception d'événements particuliers.

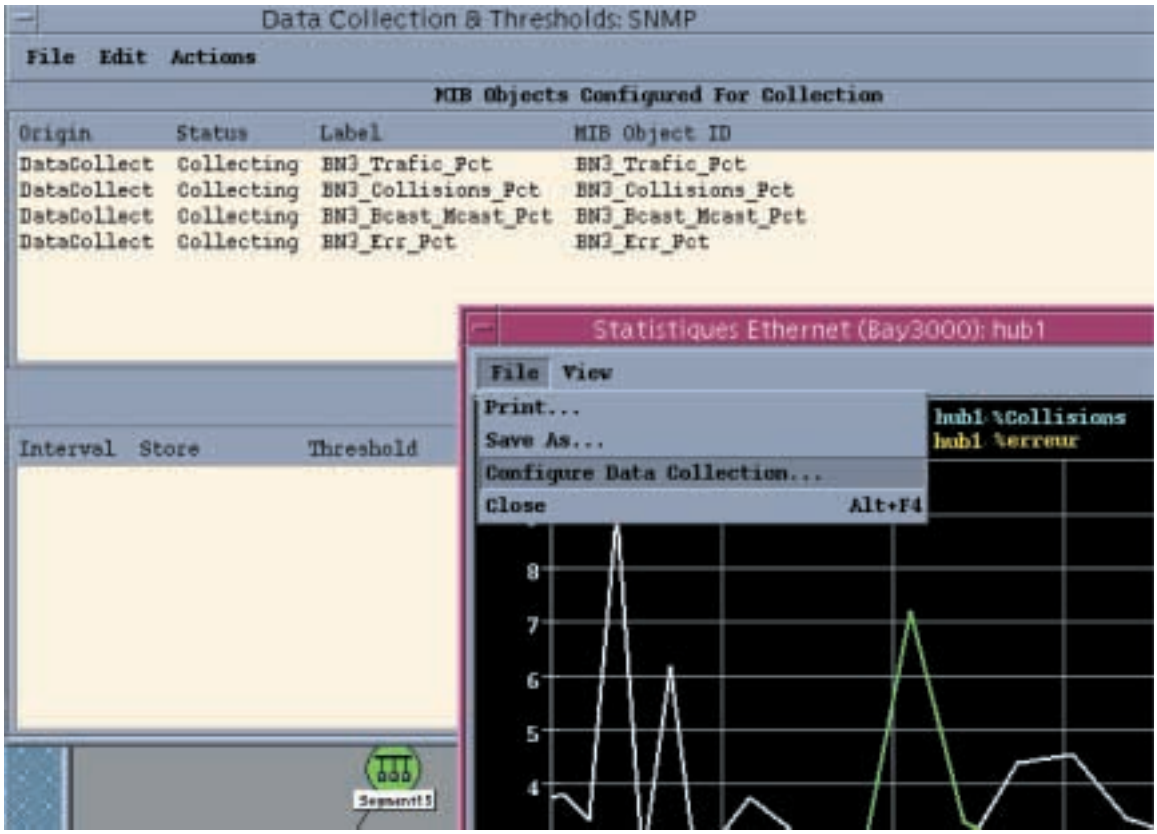

Figure 2 – Outil de collecte d'HP/Openview

Fonctions d'interrogation et de collecte

Elles permettent de récupérer l'état des composants du réseau en temps réel et si nécessaire de générer des alarmes. La plupart des plates-formes savent utiliser le protocole SNMP et peuvent éventuellement utiliser d'autres protocoles d'administration, CMIP en particulier.

Outils de collecte et de visualisation des informations sur le réseau (outils de gestion des collectes, « grapheur », « explorateur de MIB » ou *MIB browser,* etc.) (figure **2**).

Outils de développement rapide

Ils permettent de construire des requêtes prédéfinies pour les intégrer dans les menus contextuels de l'interface graphique (par exemple : récupérer et afficher sous forme de tableau la table des interfaces d'un équipement, grapher les collectes et les informations en temps réel sur la CPU d'une machine...).

1.1.2.3 Interfaces de programmation

Les API permettent à des programmes tiers d'accéder aux services de la plate-forme : accès aux informations de topologie, à la base des objets, aux collectes, actions ou récupérations d'informations sur les objets, accès aux événements (génération et récupération), etc. Les API publiées par les éditeurs et utilisées pour l'intégration d'applications tiers sont en général propriétaires. Un standard a bien été publié par l'Open Group (anciennement X/Open) en 1994 : il s'agit de l'**API XMP** qui fournit un accès aux protocoles SNMP et CMIP, mais bien que disponible – sous forme d'extension parfois – sur les plates-formes, son usage reste souvent limité aux applications devant utiliser CMIP. Cela explique que jusqu'à présent, les éditeurs de solutions tierces parties hésitent en général à développer en s'appuyant sur les services des plates-formes, car le marché veut qu'ils rendent leurs applications disponibles pour les quelques plus courantes. L'intégration faite est donc en général minimum, de manière à limiter les coûts de développement. Par exemple, les éditeurs utiliseront leur propre pile de protocole SNMP, leur propre base de données des objets, et développeront des programmes batch de synchronisation pour les données communes comme le paramétrage des objets.

Cette situation est probablement amenée à évoluer si les standards **WBEM** (Web Based Enterprise Management) et **CIM** (Common Information Model) promus par le DMTF (Distributed Management Task Force) se développent et sont fournis par les éditeurs de plates-formes. Ce dernier donne en effet un modèle de programmation objet et est ouvert pour le développement d'applications d'administration. Poussé par Microsoft pour l'administration système de sa plate-forme Windows ainsi que par la plupart des acteurs du marché de l'administration, il est probable qu'il devienne un standard de fait.

1.1.2.4 Services étendus

Les services étendus qui devraient être proposés par une plate-forme mais qu'il faut souvent aller chercher ailleurs en complément sont les suivants.

Gestion distribuée : un protocole comme SNMP étant assez verbeux, il est souvent nécessaire de distribuer l'administration entre plusieurs machines qui se répartissent le travail d'interrogation (*polling*) des équipements et de filtrage des événements. Cela permet de limiter le trafic sur le réseau et aussi de distribuer à des fins

d'organisation pour donner accès aux outils à des gestionnaires de site.

- **Gestion des collectes de données** : il est possible avec une plate-forme de collecter les données et de constituer un dépôt de données (*datamart*) qu'il sera possible d'exploiter avec des produits complémentaires (outils de traitement statistique et réalisation de tableaux de bord pour analyse de la QoS, gestion des capacités et des performances, gestion de la facturation, etc.). La plupart du temps, les outils fournis dans la plate-forme sont relativement rudimentaires et tout un marché indépendant s'est développé pour répondre à ces besoins, en dehors des plates-formes dont cela devrait pourtant être l'une des fonctions.
- **Gestion étendue des topologies**, avec découverte et visualisation des topologies de niveau 2, des VLAN, des réseaux IPX (Internet Protocol eXtension), etc.
- **Gestion de vues orientées « métier »** plus que topologiques, permettant de visualiser les alertes liées aux processus métier de l'entreprise. Ces vues s'imposent dès que l'on ne se limite pas au réseau mais que l'on cherche à intégrer sur la plate-forme l'administration du système d'information.
- **Systèmes experts** apportant une aide au diagnostic.

1.1.3 Gestion des incidents

Ces outils permettent s'assurer le suivi administratif des dossiers d'incidents (ou « tickets d'incidents ») ouverts lors de problèmes. Utilisés par l'ensemble des acteurs du support, de l'exploitation et de la maintenance, ils permettent de contrôler le *workflow* de traitement des incidents entre les intervenants, de coordonner leurs actions, de déclencher les escalades, d'assurer la traçabilité des problèmes et de faire le suivi des contrats de QoS support négociés avec les utilisateurs en termes de MTTR (Mean Time To Repair), disponibilité, etc.

L'interface avec une plate-forme d'administration permettra d'ouvrir de manière automatique des dossiers, pour des problèmes bien identifiés, le risque étant ici de générer des dossiers non pertinents.

1.1.4 Tableaux de bord

Les plates-formes sont souvent utilisées pour le traitement « à chaud » des problèmes, au moyen des alarmes qu'elles fédèrent. Il est toutefois utile – et important – d'utiliser l'ensemble des informations collectées, que ce soit les alarmes ou les échantillons de données sur les statistiques et les états remontés par les équipements. Les usages qui peuvent en être faits sont les suivants :

— établissement de tableaux de bord techniques permettant d'avoir une vision synthétique de l'état et de l'évolution du réseau, de manière à détecter les anomalies et à anticiper les problèmes de performance ou de blocage potentiels. Cette vision complète la vision en temps réel des alertes en permettant de dégager les tendances :

- les rapports d'exception permettent de pointer les problèmes,
- les rapports de tendance permettent de faire un *capacity planning* et d'anticiper les évolutions ;

— établissement de tableaux de bord de QoS, qui permettront de suivre des indicateurs de qualité de service négociés avec les utilisateurs lors de l'établissement de conventions de service (SLA : Service Level Agreements). Ce sera par exemple des temps de réponse réseau, des disponibilités, des temps de réparation des pannes (MTTR), etc.

On trouve dans cette catégorie d'outils soit des applications complémentaires à la plate-forme qui exploiteront les possibilités de collecte de cette dernière et s'occuperont de la gestion des rapports, soit des applications « verticales » capables de prendre en charge collecte, agrégation et présentation des données.

1.1.5 Applications constructeurs

Chaque constructeur accompagne ses produits des outils de configuration, de diagnostic et de dépannage, d'analyse, etc. Ils s'appuient à la fois sur les standards et sur les aspects propriétaires des équipements et permettent de donner aux administrateurs et aux exploitants des outils souvent plus ergonomiques que les outils de base qu'on trouve ou qu'on peut intégrer à partir d'une plate-forme d'administration (figure **3**). Par exemple, une application commune permet de visualiser la face avant de l'équipement à administrer ; des codes de couleur indiquent l'état de chaque composant et des menus contextuels permettant d'accéder à des fonctions statistiques ou de configuration. On retrouve aussi les applications de type plate-forme, mais qui ne permettent d'adresser que la problématique d'un réseau monoconstructeur : ce pourront être des vues topologiques de réseau souvent plus riches (support des VLAN par exemple) que celles proposées par une plate-forme généraliste, mais au détriment de la vue d'ensemble qu'offre cette dernière. Ces applications sont souvent intégrées de manière plus ou moins étroite aux plates-formes « ouvertes », la forme la plus rudimentaire consistant à fournir de nouveaux menus à la plate-forme qui permettront de lancer les outils, les formes plus évoluées prenant en compte la base de données et les fonctions de la plate-forme.

1.1.6 Applications génériques

Il convient de choisir judicieusement des applications « génériques » qui permettent de traiter des problèmes généraux ou particuliers en relation avec des équipements multiconstructeurs. Ce sont par exemple des applications s'appuyant sur les standards RMON (Remote Monitoring) et RMON2 pour l'administration des réseaux locaux (figure **4**), une application de configuration de routeurs multiconstructeurs, etc., l'intérêt étant dans ce cas d'éviter d'avoir autant d'outils que de types d'équipements.

1.1.7 Agents

L'ensemble des outils d'administration s'appuie sur les informations reportées par les agents distribués sur le réseau. Il est indispensable que ceux-ci soient de bonne qualité, avec en premier lieu la remontée d'indications fiables. Ce n'est pas toujours le cas, en particulier pour les agents « embarqués » sous forme logicielle dans les équipements eux-mêmes et qui partagent les mêmes ressources de calcul que les fonctions vitales de la machine. Par exemple, un routeur ou un commutateur réseau dont la fonction principale est le transfert de paquets accordera une priorité moindre aux processus de traitement SNMP et au comptage des statistiques, sauf si ces fonctions sont réalisées dans une carte indépendante. C'est pourquoi il est commun d'utiliser des équipements dédiés, comme par exemple une sonde RMON qui analyse le fonctionnement d'un réseau local.

Par ailleurs, ces agents sont plus ou moins respectueux des standards établis (nombre de standards supportés et qualité de l'implémentation) et feront un usage de fonctions propriétaires pour répondre à des besoins spécifiques. Il est nécessaire, dans le choix des équipements, de s'assurer du support des versions les plus récentes des standards et des capacités et de la stratégie d'évolution concernant le support de ces standards. Cela permettra d'assurer la pérennité des solutions choisies, avec éventuellement la possibilité de changer ou de mettre en place de nouveaux produits d'administration qui sauront exploiter les informations apportées par les agents sans impliquer le choix d'un fournisseur.

L'implémentation des agents n'est pas standardisée : certains seront embarqués sous forme *firmware* dans des équipements, alors que d'autres intégreront les fonctions dans l'OS ou sous forme de programmes additionnels.

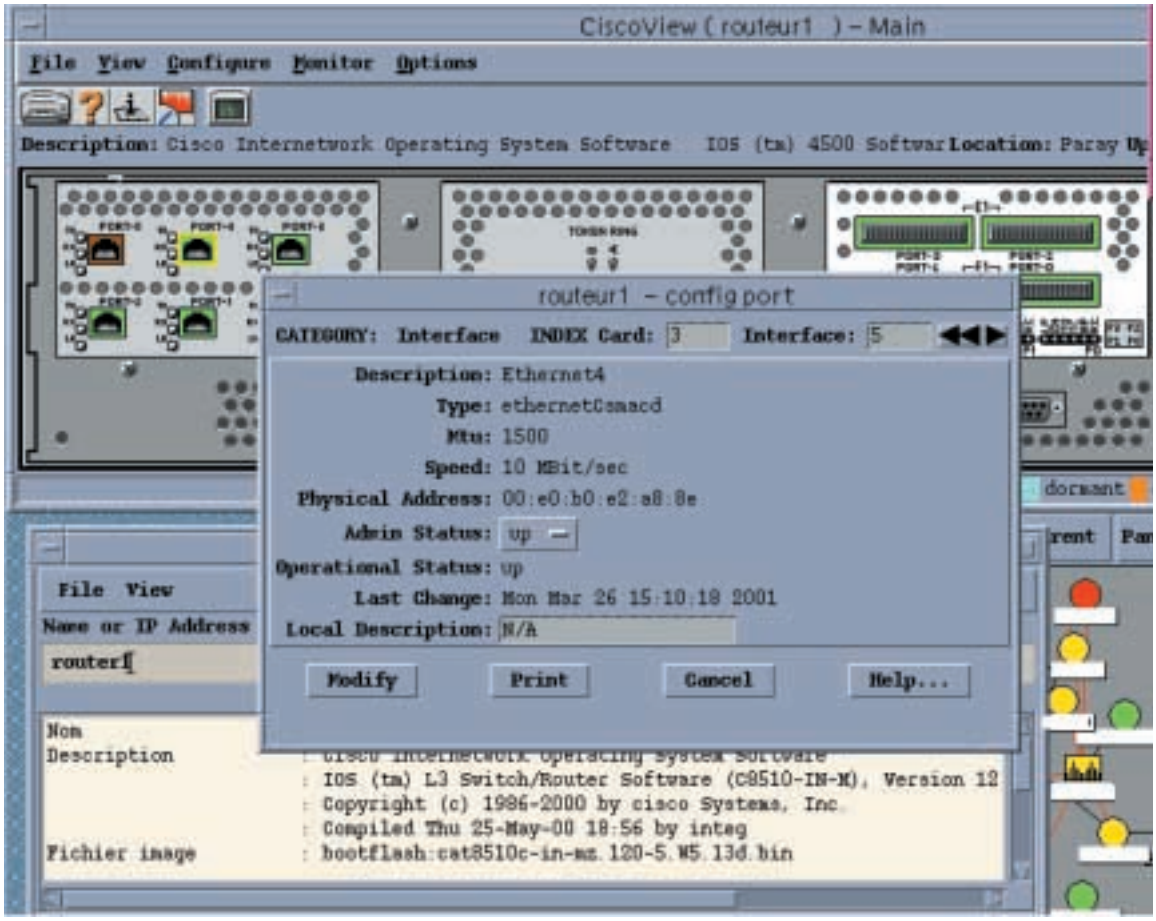

Figure 3 – CiscoWorks et HP/Openview

De manière générale, leur code est modifiable par le constructeur de façon à pouvoir prendre en compte de nouvelles fonctionnalités (support de nouvelles MIB standard par exemple). Sur les systèmes d'exploitation généralistes (Unix ou Windows), il est intéressant d'utiliser des agents extensibles et modulaires avec des API d'extension publiées ou utilisant une technologie d'extension standard, comme AgentX.

Exemple : hubs et switchs

Pour un hub, il est utile de vérifier l'implémentation des modules de MIB suivants :

— RMON (si possible RMON2) ;
— IEEE 802.3 Medium Attachment Units (MAU) (RFC 2668) ;
— MIB Etherlike Interface (RFC 2665) ;
— interface (RFC 2233) ;
— 802.3 Repeater (RFC 2108).

Pour un switch, on vérifiera l'implémentation de la MIB bridge (RFC 1493), de RMON (support d'au moins quatre groupes par port, support de RMON et RMON2 par copie de trafic sur un port dédié), SMON (Switched RMON), entity...

1.1.8 Stratégie de mise en œuvre d'outils

Il semble judicieux de savoir capitaliser sur des outils « génériques » et fédérateurs tels qu'une plate-forme (pour la supervision, le lancement des outils, la collecte de données), des outils de tableau de bord, RMON, etc., permettant d'avoir une bonne vision d'ensemble de ses réseaux, tout en sachant utiliser des outils plus productifs pour des besoins spécifiques (par exemple, configuration d'équipements, outils ayant une forte valeur ajoutée bien que s'appuyant sur des technologies propriétaires). La personnalisation de la plate-forme en vue de l'adapter aux besoins spécifiques est une tâche coûteuse en temps et en argent mais doit permettre de mettre en œuvre une solution qui répondra aux besoins, et non pas un produit qui restera un jouet inutile abandonné au bout de quelques mois. Quelques développements rapides pourront permettre de répondre à des besoins ponctuels non adressés par les solutions ou spécifiques à l'entreprise. Un certain nombre de logiciels libres sont par ailleurs d'un intérêt certain et peuvent être utilisés avec profit [Doc. H 2 840].

1.2 Organisation

1.2.1 Conventions de service

Avant de mettre en œuvre quelque solution que ce soit, il est important de bien savoir quelles en sont les finalités. C'est pourquoi

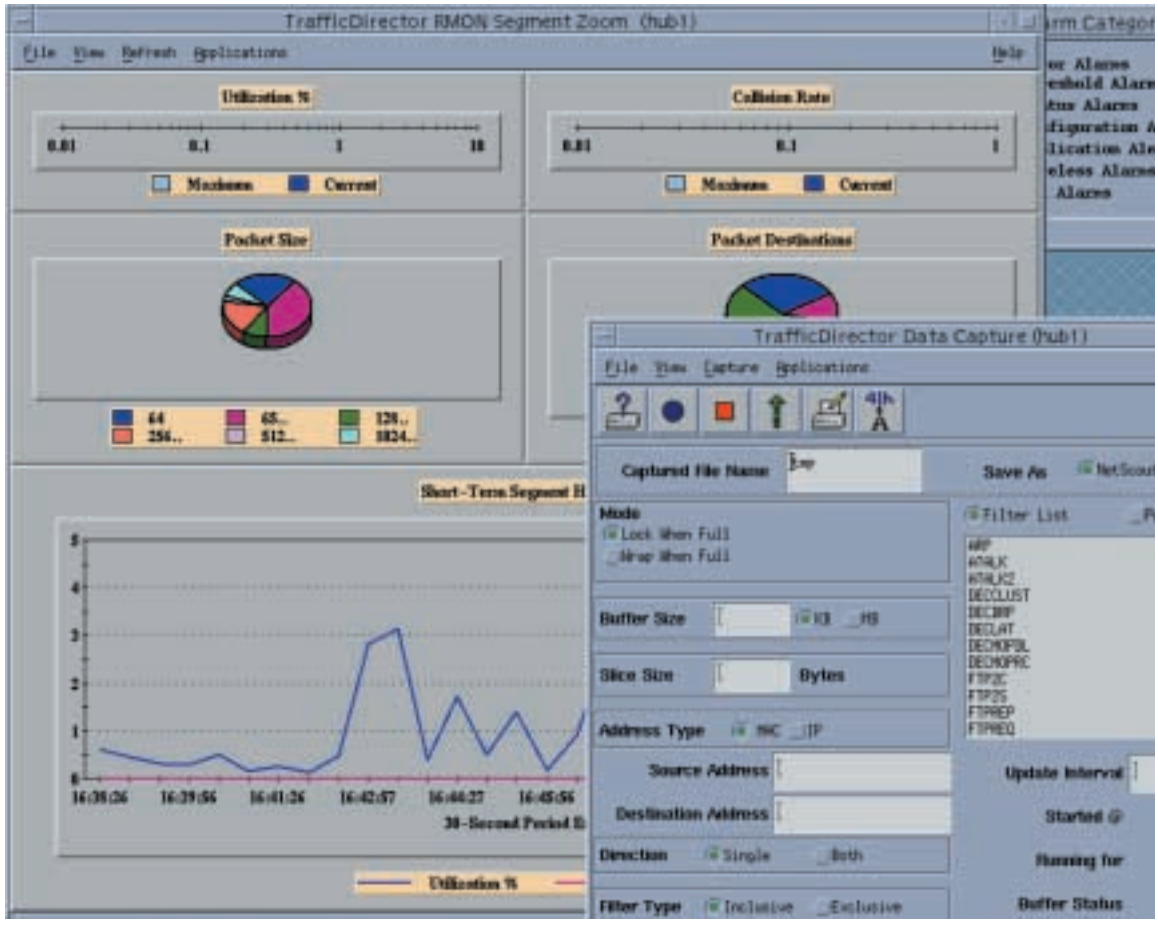

Figure 4 – Outils RMON de CiscoWorks

des conventions de service (SLA) sont à négocier avec les utilisateurs. Il faut déterminer avec eux de manière précise :

— leurs besoins et les services qu'ils attendent : disponibilité, plages de service, capacités, niveaux d'assistance, sécurité ;

— les coûts associés à ces services.

Sans SLA, l'utilisateur risque d'être sans cesse mécontent et le fournisseur toujours sous pression, sans moyens suffisants pour assurer la tâche non précisée qui lui est demandée.

1.2.2 Organisation des équipes

L'organisation se fait entre les deux pôles que sont l'assistance utilisateurs d'une part et l'exploitation et le déploiement des réseaux d'autre part, les activités s'articulant entre ces deux pôles et ayant toujours pour objectif la qualité du service rendu au client. On doit retrouver dans l'organisation des processus les circuits d'escalade technique suivis après l'ouverture de tickets d'incidents par le contact utilisateur ou les équipes de supervision.

En général, on organise le support sur une pyramide à trois niveaux, avec au premier niveau un point de contact unique pour l'utilisateur (*help desk* ou *hot line*) chargé de l'assistance téléphonique. L'objectif est ici de répondre le plus rapidement possible à une majorité de problèmes avec des procédures précises (40 % à 80 %). Cette équipe escalade les problèmes à des équipes de deuxième niveau, plus spécialisées par domaine, qui résoudront la majorité des problèmes restants. Pour les quelques pour-cent finaux, les escalades sont faites vers les équipes étude, les éditeurs ou constructeurs, etc.

Par ailleurs, il est important de mettre en place (si la taille des réseaux le justifie) des équipes de surveillance chargées de la supervision (ou pilotage) du réseau, et qui doivent traiter les incidents avant qu'ils ne deviennent des problèmes sensibles pour les utilisateurs. Cette tâche est d'autant plus importante qu'il est plus difficile d'établir un diagnostic à partir des éléments indiqués par les utilisateurs (qui n'ont souvent pas la vision technique mais posent les problèmes par rapport à leurs préoccupations fonctionnelles) qu'à partir des alertes remontées par le réseau.

On a donc affaire à différents métiers adressant chacun une des étapes du processus. Les analystes et ingénieurs réseau font du moyen terme, avec l'analyse des tendances, les prévisions d'évolution de capacité ou d'architecture, etc. Les administrateurs s'occupent de la configuration et de la gestion du parc, et font assez souvent du support de deuxième niveau ; c'est un travail plus opérationnel. Les pilotes ou opérateurs assurent la surveillance du réseau et essaient de résoudre les incidents avant qu'ils n'aient un impact grave. Enfin, les opérateurs de *help desk* sont directement en prise avec les utilisateurs lorsque de simples incidents deviennent des problèmes.

Un autre aspect à prendre en compte est le compromis entre la centralisation et la distribution des activités. La centralisation apporte une réponse à la rareté des experts et permet de capitaliser sur leur expérience. Elle permet aussi d'avoir une vision globale, ce qui est essentiel pour des activités tournant autour des architectures dorsales. La proximité est obligatoire pour un certain nombre d'activités (maintenance *hardware*, déploiements) mais peut présenter des avantages au niveau de l'assistance, essentiellement du fait qu'une équipe locale sera plus proche des préoccupations pratiques des utilisateurs et connaîtra plus précisément les configurations d'un site qu'une équipe centralisée qui a nécessairement plus de systèmes à prendre en compte.

1.2.3 Industrialisation

Pour que l'organisation soit capable de maîtriser les systèmes qu'elle met en place, il faut absolument industrialiser les processus de mise en production, de manière à ce que :

— les installations se fassent selon des configurations homogènes et standard, qui seront connues des équipes de support. Il est beaucoup plus facile de maîtriser le réseau si ce qu'on y trouve est installé de manière rationnelle et homogène ; cela évite de perdre du temps à comprendre comment il est supposé fonctionner ;

— des documentations d'exploitation soient mises au point, permettant à chacun de retrouver les informations qui lui sont nécessaires pour faire évoluer le système ou dépanner. On y trouvera en particulier :

- les documents d'architecture pour les équipes de deuxième niveau qui devront se pencher sur les problèmes pour lesquels des procédures de dépannage n'ont pas été prévues,
- des procédures de dépannage qui éviteront les erreurs et les recherches improvisées lors d'incidents : le déroulement des dépannages est sécurisé, du temps est gagné et cela banalise les interventions, en évitant de faire appel aux spécialistes du domaine,
- des consignes de traitement des alarmes réseau. Il est souhaitable qu'un lien soit fait entre les outils de supervision et la documentation (qui devra dans ce cas être électronique), de manière à accélérer le traitement des alarmes,
- des procédures de passage en mode dégradé lors d'incidents majeurs est un des aspects à ne pas oublier,
- des procédures d'escalade technique pour les problèmes qui ne trouvent pas rapidement de solution,
- des procédures d'escalade hiérarchique pour les problèmes ayant un fort impact sur le métier de l'entreprise, etc.

1.3 Facteur humain

Une organisation n'existe que par les hommes qui la constituent. Pour qu'elle fonctionne correctement, il faut que ceux-ci soient motivés, compétents et aient une vision précise du travail qui leur est demandé et de la manière dont il s'articule avec celui des autres intervenants. Des formations fréquentes sont nécessaires, de même que des mises au point entre les équipes. Des échanges de personnel entre services peuvent permettre de bien appréhender tous les aspects des problèmes et pas uniquement la vision que chacun peut en avoir.

2. SNMP

SNMP (Simple Network Management Protocol) est le protocole standardisé pour l'administration du réseau Internet et il est également très largement déployé dans les réseaux d'entreprise. Trois versions coexistent.

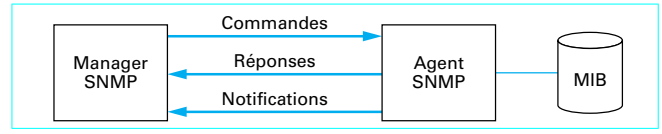

Figure 5 – Modèle agent/manager

SNMP v1 est encore la plus utilisée en 2001, bien que ce soit la plus rudimentaire.

Il y a eu plusieurs versions expérimentales de **SNMPv2**, qui devait remplacer SNMPv1, et en particulier lui apporter les fonctions de sécurité qui lui font défaut. Mais, faute d'un consensus au niveau des groupes de travail de l'IETF, c'est la version intermédiaire et expérimentale connue sous le nom de **SNMPv2C** qui est utilisée par la plupart des éditeurs supportant SNMPv2. La sécurité est encore quasiment nulle car elle reprend le modèle de SNMPv1, à base de « noms de communauté » (d'où le C de SNMPv2C). Malgré tout, elle comble des lacunes de la version 1, en particulier au niveau de la définition des objets, du traitement des notifications et du protocole lui-même (ajoutant une commande GETBULK de manière à minimiser les échanges réseau, particulièrement lourds dans le cas de récupération de tables avec les commandes GETNEXT de SNMPv1). Cette version n'a pas eu un important succès au niveau des déploiements sur les réseaux, faute de progrès conséquents par rapport à la version 1.

SNMPv3 apporte essentiellement des fonctions de sécurité et formalise de façon complète le modèle d'administration SNMP. C'est le standard cible qui reste encore à être éprouvé par le marché.

2.1 Modèle simplifié

Les outils d'administration de réseau sont le plus souvent basés sur une architecture de type client/serveur. Dans celle-ci, le client, appelé « manager », est le logiciel qui centralise les informations en provenance de l'ensemble des équipements de réseau par l'intermédiaire d'« agents » qui peuvent agir pour son compte.

Le « manager » interroge, envoie des commandes de contrôle ou reçoit des notifications de la part des agents qui représentent les composants à administrer (figure **5**). Pour cela, il faut que les deux parties utilisent :

— un **dialogue standardisé** : c'est le rôle du protocole d'administration de réseau qui formalise les échanges possibles. Ce protocole peut être propriétaire (par exemple, les NMVT – Network Management Vector Transport – du monde IBM SNA) ou standard (par exemple, CMIP dans le monde des réseaux OSI et SNMP dans le monde des réseaux TCP/IP). Ce protocole est du niveau application. Le manager peut émettre des commandes de type GET qui permettent de récupérer l'état des objets représentés par l'agent, tandis que les commandes de type SET permettent de le changer. Un agent peut aussi émettre des notifications, pour avertir de changements d'états par exemple ;

— une **représentation commune des composants administrés** : cela est réalisé au moyen d'une base d'information appelée MIB (Management Information Base). Celle-ci constitue la description formelle et commune des objets administrés. Pour formaliser la description des objets, on utilise un langage formel, appelé SMI (Structure of Management Information) dans le cas de SNMP et GDMO (Guidelines for the Definition of Managed Objects) pour CMIP.

2.2 Management Information Base

L'élément le plus important dans le protocole SNMP est probablement qu'il a permis de décrire un grand nombre de composants

(réseaux ou autres) de façon standard. Cette description est faite dans la MIB (Management Information Base), qui regroupe les informations d'intérêt pour les administrateurs. On y retrouve toute l'expérience des spécialistes qui ont établi les modèles concernant les sujets qu'ils maîtrisent, ce qui en fait tout l'intérêt pour les administrateurs réseau.

Le modèle n'est pas un modèle objet : les entités modélisées ne sont pas des objets au sens informatique du terme, mais plutôt un ensemble de variables typées qui peuvent être lues ou mises à jour. Un certain nombre d'astuces sont utilisées ensuite pour permettre de définir des opérations complexes sur ces objets.

2.2.1 Nommage des objets

Chaque objet de la MIB est identifié par un nom et un OID (Object Identifier) qui le décrit de manière unique dans un espace de nommage global et hiérarchique. Ce dernier est représenté par un arbre inversé, dans lequel chaque bifurcation représente un objet. Chaque objet possède un numéro unique qui le situe par rapport au nœud père, ainsi qu'un nom symbolique. Le chemin suivi pour aller de la racine à l'objet constitue l'OID de celui-ci.

Exemple : sur la figure **6**, l'objet décrivant le nom d'une machine s'appelle sysName et a un OID défini de la manière suivante :

```
sysName OBJECT IDENTIFIER ::= { iso org(3) dod(6) internet(1) mgmt(2) mib-2(1)
system(1) sysName(5)}
```

ou encore, s'il est défini à partir de son père :

```
sysName OBJECT IDENTIFIER ::= { system 5 }
```

En abrégé, il est noté .1.3.6.1.2.1.1.5

De fait, l'arbre de nommage de la figure **6** est infini. Dans l'espace de nommage, seule la partie située sous 1.3.6 (dod) est utilisée par SNMP. La partie supérieure est gérée par l'ISO, qui a prévu un espace de nommage pour le DoD (Département de la Défense américain), organisme à l'origine d'Internet.

2.2.2 Modules de MIB

Les objets sont décrits par ensembles qu'on appelle « modules de MIB ». Par abus de langage, on les appelle les MIB. Ils sont classés dans l'espace de nommage sous la branche « internet » :

— « mgmt (2) » contient tous les modules de MIB standard établis par l'IETF ;

— « expérimental (3) » contient les modules de MIB établis de manière expérimentale par l'IETF (sachant qu'on les retrouvera sous leur forme définitive dans des sous-arborescences de « mgmt », plus ou moins modifiés, après aboutissement du standard) ;

— « private (4) » contient les modules de MIB développés en dehors des standards IETF, par exemple par des entreprises ou des centres de recherche. Les modules de MIB développés sous cette arborescence sont communément appelés « MIB privées ».

Les modules de MIB IETF sont standardisés selon un processus décrit dans la RFC 2438 (« Advancement of MIB specifications on the IETF Standards Track »), qui indique les critères qui permettent de savoir si les implémentations sont multiples, indépendantes et inter-opérables, ce qui permet de faire évoluer la RFC vers l'état de « proposed standard ».

2.2.2.1 MIB2

La RFC 1213 (STD 17) décrit le module mib-2 qui recense les objets devant absolument être présents sur un agent SNMP si les protocoles correspondants le sont (tableau **1**).

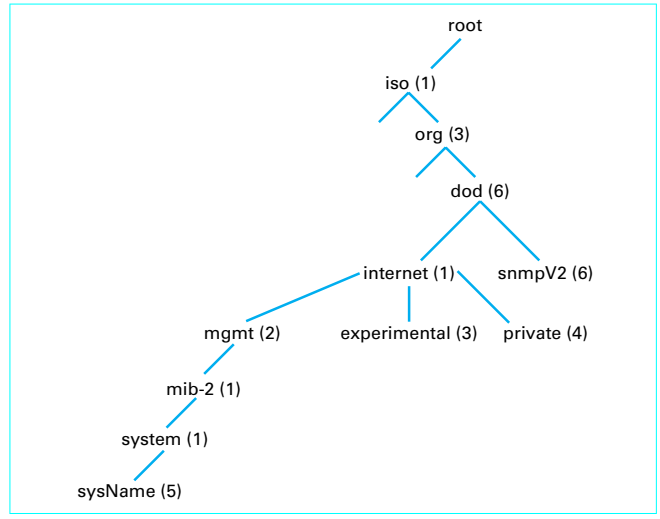

Figure 6 – MIB

Tableau 1 – Groupes mib-2

Groupe	OID	Description
system	{ mib-2 1 }	Système géré : son nom, le type d'équipement, etc.
interface	{ mib-2 2}	Interfaces réseau. La table ifTable contient la description, l'état et les statistiques des interfaces.
at	{ mib-2 3 }	Address Translation : atTable contient la table des adresses MAC (Media Access Control) et réseau. Ce groupe est peu utilisé (*deprecated*) car remplacé par les équivalents spécifiques à un protocole réseau.
ip	{ mib-2 4 }	IP : configuration générale, statistiques, table des adresses (ipTable), de routage (ipRouteTable), table ARP (Address Resolution Protocol) (ipNetToMediaTable). ipRouteTable (21) est en principe remplacée par ipForward (24) (défini dans la RFC 1354) car indexée par l'adresse destination et donc ne supportant pas les routes multiples.
icmp	{ mib-2 5 }	ICMP (Internet Control Message Protocol) : statistiques.
tcp	{ mib-2 6 }	TCP : configuration, statistiques générales et table des connexions (tcpConnTable).
udp	{ mib-2 7 }	UDP : statistiques et table des *listeners* (udpTable).
egp	{ mib-2 8 }	EGP (Exterior Gateway Protocol) : statistiques et table des voisins (egpNeighTable).
cmot	{ mib-2 9 }	CMIP over TCP/IP : obsolète. Seul l'OID est réservé dans mib-2.
transmission	{ mib-2 10 }	Ce groupe est prévu pour « raccrocher » d'autres modules de MIB qui concernent des médias de transmission plus spécifiques qui viennent compléter les informations contenues dans le groupe interface. Par exempte : dot3 (7) décrit le média Ethernet, tandis que dot5 (9) décrit Token-ring.
snmp	{ mib-2 11 }	SNMP : statistiques.

2.2.2.2 Extensions relatives aux interfaces

Média

Les objets spécifiques à un média sont définis dans un module de MIB spécifique, sous la branche de MIB mib-2.transmission (tableau **2**). On retrouve en particulier dans ces modules des tables qui complètent la table des interfaces ifTable avec des objets spécifiques au média (par exemple, collisions pour Ethernet).

Tableau 2 – Exemples de branches de MIB sous mib-2.transmission

OID	RFC	Média
lapb(16)	RFC 1381	X25 trame-LAPB
x25 (5)	RFC 1382	X25 packet
dot3(7)	RFC 2665	ethernet-like
dot5(9)	RFC 1748	token-ring
fddi(15)	RFC 1285	FDDI
ds1(18)	RFC 1406	DS1/E1
isdnMib(20)	RFC 2127	ISDN
ppp(23)	RFC 1471	PPP
ds3(30)	RFC 1407	DS3/E3
sip(31)	RFC 1694	SONET IP
frame-relay(32)	RFC 2115	Frame Relay DTE
rs232(33)	RFC 1659	RS-232-like
para(34)	RFC 1660	Parallel-printer
miox(38)	RFC 1461	IP over X.25

LAPB : Link Access Protocol Balanced
DS : Digital Signal
ISDN : Integrated Services Digital Network
PPP : Point-to-Point Protocol
SONET : Synchronous Optical Network

Interface Group MIB

Le module « Interface Group MIB » (RFC 2233), dont l'OID est {mib-2 31}, étend et reprend les définitions du groupe interface de mib-2. Il ne définit que des objets applicables, quel que soit le média.

Les définitions des objets de la table d'interfaces ifTable sont précisées pour lever les ambiguïtés :

— les interfaces représentent des entités logiques (couches de protocole par exemple) ou physiques ;

— les compteurs comptent les nombres délivrés au niveau supérieur ;

— selon le type d'interface, tous les objets ne sont pas supportés (interfaces orientées caractère, paquet ou bits), ce qui est défini à l'aide de *conformance statements* : ifGeneralInformationGroup, ifPacketGroup, ifFixedLengthGroup ;

— pour ifNumber et ifIndex, il y a possibilité de « trous » dans la numérotation et possibilité de changement de ifIndex après un boot.

La table ifXTable étend la table des interfaces et prend en compte les statistiques et définitions d'interfaces en compteurs 64 bit (High Capacity Counters), pour les médias qui le nécessitent. Elle permet aussi au manager SNMP de donner un alias aux interfaces, qui gardera sa valeur, contrairement aux numéros d'interface. Il peut ainsi plus aisément retrouver les instances d'interface lors des boots. Enfin, l'objet ifLinkUpDownTrapEnable permet de déterminer la remontée ou non de traps lors des changements d'état, interface par interface.

La table ifStackTable permet de définir la manière dont s'empilent les couches définies par les interfaces. Enfin, un type IANAIfType est défini pour permettre de maintenir la liste sans cesse plus grande des types d'interface en dehors de la MIB. C'est l'IANA (Internet Assigned Numbers Authority), organisme dépendant de l'IETF, qui gère cette liste.

2.2.2.3 RMON

RMON (Remote Monitoring, tableau **3**) est le standard d'interopérabilité par excellence pour la supervision et le dépannage des réseaux locaux. Implémenté de manière complète ou partielle sur les équipements de réseau (hubs, switchs, routeurs...), il existe aussi sous forme d'équipements dédiés à la fonction d'analyse (sondes RMON), potentiellement plus fiables en terme de capacité à analyser le réseau dans toutes les conditions de charge. Il est important de disposer de cette fonction sur un maximum d'équipements réseau car cela donne de plus larges possibilités d'analyse et une homogénéité dans le traitement des équipements multiconstructeurs.

Un des avantages de RMON est qu'il permet la distribution de l'intelligence sur l'équipement lui-même. La stratégie est de programmer la sonde pour qu'elle fasse l'acquisition de données de manière autonome : on choisit les intervalles d'échantillonnage, les variables à surveiller, le nombre d'échantillons à conserver dans les tables d'historique, les seuils d'alerte permettant de déclencher des traps, etc. On ne récupère les données historisées que lorsque necéssaire, par exemple, lorsqu'un trap avertit la station d'administration qu'un seuil d'alerte a été dépassé. Cela présente un avantage évident sur les solutions classiques qui impliquent d'interroger par *polling* permanent toutes les variables intéressantes et de comparer des indicateurs par rapport à des seuils au niveau de la station d'administration. Les solutions de *reporting* évoluées peuvent aussi récupérer les données aux heures creuses de trafic.

Par ailleurs, le standard permet de réaliser de l'analyse de protocole en mode client/serveur, ce qui est utile pour la résolution d'incidents et évite dans bien des cas à un spécialiste de se déplacer sur site. L'application d'administration programme sur la sonde les paramètres de capture et de filtrage de paquets et récupère les buffers pour analyse et décodage.

2.2.2.4 RMON et les réseaux commutés : SMON

La mise en place de RMON en environnement commuté est délicate, car les sondes n'ont plus accès à l'ensemble du trafic du réseau : elles ne voient que les paquets qui transitent sur le segment auxquelles elles appartiennent. Il faut donc une sonde par segment, ce qui peut s'avérer coûteux, ou mettre en place des solutions intermédiaires ; par exemple, utiliser des switchs qui supportent les groupes de base de RMON (statistics, history, alarm, event) sur chaque port, de manière à avoir les statistiques de base permettant de suivre l'état de chaque segment. Pour les analyses plus précises, une sonde RMON complète le dispositif, installée sur un port du switch configuré pour recevoir une copie du trafic en provenance d'un ou de plusieurs ports du switch.

SMON (RFC 2613 – Switched RMON) permet de standardiser le contrôle du port de copie, les remontées de statistiques VLAN, etc.

2.2.2.5 RMON 2

RMON permet de faire des analyses de niveau 2, c'est-à-dire que les tables de host et les matrices de trafic se font par rapport à des adresses MAC. RMON2 (RFC 2021) permet de remonter dans les couches et de faire des analyses concernant les couches 3 et supérieures (tableau **4**).

Tableau 3 – Groupes RMON

Groupe	OID	Description
statistics (1)	{ rmon 1 }	Donne l'état en temps réel des segments mis sous surveillance (utilisation, erreurs, distribution du trafic par taille de paquets, broadcasts, multicasts...). Utilisé en dépannage (temps réel) ou par les applications non évoluées pour le *reporting*, ou encore pour définir des seuils d'alarme sur la sonde.
history	{ rmon 2 }	Une table permet de paramétrer l'acquisition des données et une table d'historiques les met à disposition pour interrogation.
alarm	{ rmon 3 }	La table alarmTable permet de mettre en place une surveillance locale sur n'importe quelle variable de la MIB enregistrée par l'agent (alarm Variable), selon une fréquence donnée (alarmInterval), par rapport à des valeurs des seuils d'alarme montant et descendant (alarmFallingThreshold et alarmRisingThreshold), etc. Les deux seuils permettent de définir un cycle d'hystérésis qui limite le nombre d'événements générés. Les événements générés lors d'un dépassement de seuil sont identifiés par leur index dans la table des événements définis dans le groupe event (alarmRisingEventIndex et alarmFallingEventIndex).
host	{rmon 4}	Donne les statistiques des machines du réseau vues par la sonde, avec des rapports sur une période donnée. La table hostControlTable permet de définir les rapports, tandis que les résultats sont donnés par hostTable (indexation par les adresses MAC) et hostTimeTable (indexation par date de création de la ligne).
hostTopN	{rmon 5}	Fournit un rapport sur les *N* plus grands hosts, basé sur les statistiques du groupe statistics (trafic, collisions, etc.). Chaque entrée dans la table de résultats constitue un rapport.
matrix	{rmon 6}	Stocke les erreurs et les statistiques d'utilisation pour les couples paires d'équipements qui communiquent sur le réseau (par exemple : Error, bytes, packets).
filter	{rmon 7}	Moteur de filtre permettant d'isoler un flux de paquets à partir de la combinaison logique d'un filtre de données (correspondance d'un *pattern* de bits) et d'un filtre d'état (correspondance de l'état des paquets : CRC, etc.). Le résultat appliqué à une interface constitue un *channel*. Un *channel* peut être activé par un événement, et sa définition sera utilisée pour la configuration des captures de paquets (cf. groupe capture), ou simplement pour le comptage des paquets ou la génération d'un événement (préalablement défini dans la table eventTable) lors de l'arrivée du premier paquet ou/et des suivants.
capture	{rmon 8}	Permet la capture de paquets qui correspondent à un channel défini dans le groupe filter : BufferControlTable permet de définir le buffer de capture pour un channel, ainsi que la stratégie en cas de buffer plein ; CaptureBufferTable permet au manager SNMP de récupérer les paquets (une ligne pour chaque paquet capturé). Un système de fenêtre sur le buffer de capture permet de faire avec les contraintes du protocole SNMP (taille maximum des PDU).
event	{rmon 9}	La table eventTable permet de définir des événements qui peuvent déclencher un trap SNMP et/ou l'ajout d'une entrée dans un fichier journal (« log ») conservé sur l'agent sous la forme d'une table de nom logTable.
tokenRing	{rmon 10}	Le groupe étend RMON aux réseaux Token Ring, avec l'ajout des spécificités Token Ring (source routing, statistiques MAC...).

Tableau 4 – Groupes RMON2

Groupe	OID	Description
protocolDir	{ rmon 11 }	Protocol Directory Group : table des protocoles (RFC 2074) que l'agent observe et pour lesquels il maintient des statistiques.
protocolDist	{ rmon 12 }	Protocol Distribution Group : statistiques de trafic en paquets et octets pour chaque protocole du directory.
adressMap	{ rmon 13 }	Address Map Group : Liste des équivalences adresse MAC – adresse réseau découvertes par la sonde.
nlHost	{ rmon 14 }	Network Layer Host Group : statistiques de trafic de et vers chaque adresse réseau découverte par la sonde.
nlMatrix	{ rmon 15 }	Network Layer Matrix Group : Statistiques de trafic pour les paires d'adresse réseau, avec la possibilité d'établir des matrices de trafic source-destination et destination-source, ainsi que des rapports de type « top N ».
alHost	{ rmon 16 }	Application Layer Host Group : statistiques par protocole « applicatif » (au sens RMON2 du terme : protocole de niveau supérieur à 3), pour chaque adresse réseau du segment.
alMatrix	{ rmon 17 }	Application Layer Matrix Group : statistiques de trafic par protocole pour chaque paire d'adresses réseau identifiée par la sonde.
usrHistory	{ rmon 18 }	User History Group : il permet de mettre en place des historiques similaires à ceux définis dans RMON, mais sur n'importe quel objet de type INTEGER ou dérivé.
probeConfig	{ rmon 19 }	Probe Configuration Group : définition des groupes supportés par la sonde, liste des machines destinatrices des traps SNMP, définition du fichier de boot TFTP (Trivial File Transfer Protocol), etc.

2.3 Description des objets : SMI

2.3.1 Abstract Syntax Notation : ASN.1

La MIB est décrite au moyen d'un langage formel appelé SMI (Structure of Management Information). Celui-ci utilise un sous-ensemble de la syntaxe OSI, ASN.1, qui permet de représenter des structures de données. Quelques notions concernant ASN.1 sont à connaître pour comprendre SMI et la MIB.

Module

Un module ASN.1 décrit une collection d'objets.

```
NOMDUMODULE DEFINITIONS ::=
BEGIN
      Relations avec les autres modules (clauses
IMPORT et EXPORTS)
      Définition des objets
END
```

Objets

Les objets définis avec ASN.1 peuvent être :

— des types, avec des types simples comme INTEGER ou BOOLEAN, et des types construits permettant de définir des listes (SEQUENCE) et des tableaux (SEQUENCE OF) ;

— des valeurs, c'est-à-dire des objets ayant un type précédemment défini ;

— des macros, qui permettent d'étendre les définitions et définir de nouveaux types.

Par convention, les types commencent par une majuscule, les valeurs par une minuscule et les macros sont tout entières en majuscules. Les commentaires sont précédés de deux tirets.

2.3.2 Structure of Management Information

SMI est la syntaxe utilisée pour représenter les objets de la MIB, décrite au moyen d'ASN.1. Deux versions de SMI coexistent : la première est décrite dans la RFC 1157. La seconde (RFC 2579) est compatible ascendante avec la première et doit être utilisée pour écrire les RFC décrivant de nouveaux modules de MIB.

2.3.2.1 Modules de MIB

Les modules de MIB sont décrits en suivant normalement la syntaxe ASN.1. En SMIv2, la macro MODULE-IDENTITY permet d'identifier plus précisément le module en lui affectant un OID et en décrivant des informations telles que le contact et le nom de l'auteur, l'historique des versions, etc.

2.3.2.2 Objets

SMI permet de décrire les objets de la MIB, avec :

— leur syntaxe ;

— les opérations autorisées ;

— leur état de standardisation.

Exemple : un objet est défini au moyen de la macro OBJECT-TYPE, qui est légèrement différente en SMIv1 et v2.

En SMIv1, on aura (RFC 1516) :

```
rptrMonitorTransmitCollisions OBJECT-TYPE
  SYNTAX Counter
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION "This counter is incremented every
time the repeater state machine enters the TRANSMIT
COLLISION state from any state other than ONE PORT
LEFT (Ref : Fig 9-2, IEEE 802.3 Std). The approximate
minimum time for rollover of this counter is
16 hours."
  REFERENCE "Reference IEEE 802.3 Rptr Mgt,
19.2.3.2, aTransmitCollisions."
: : = {rptrMonitorRptrInfo 1 }
```

Et en SMIv2 (même exemple à partir de la RFC 2108 qui remplace la RFC 1516) :

```
rptrMonitorTransmitCollisions OBJECT-TYPE
  SYNTAX Counter32
  MAX-ACCESS read-only
  STATUS deprecated
  DESCRIPTION "*********THIS OBJECT IS DEPRECATED
**********
  For a clause 9 (10 Mb/s) repeater, this counter
is incremented
  ..."
  REFERENCE "[IEEE 802.3 Mgt], 30.4.1.1.8, aTrans-
mitCollisions."
: : = { rptrMonitorRptrInfo 1 }
```

Dans cet exemple, l'objet de nom `rptrMonitorTransmitCollisions`, dont la description textuelle est donnée sous DESCRIPTION, a pour type Counter (équivalent à Counter32 en SMIv2), n'est accessible qu'en lecture seule. Il devait être implémenté de manière obligatoire avec la RFC 1516, mais avec la RFC 2108, cet objet est devenu *deprecated*, c'est-à-dire qu'il est désuet mais qu'on peut continuer à l'implémenter pour des raisons de compatibilité. Son OID est rptrMonitorRptrInfo.1.

2.3.2.3 Syntaxe des objets

La MIB SNMP est composée de différents objets :

— **objets partiels**, qui servent à construire l'arborescence de la MIB. Ils n'ont pas de type, mais un OID. Par exemple, l'objet `system OBJECT IDENTIFIER ::= { mib-2 1 }` groupe les objets de ma mib-2 qui permettent d'identifier un agent SNMP ;

— **objets scalaires**, c'est-à-dire qu'ils ont une instance unique. Ils auront un type simple (faisant partie de la syntaxe de base ASN.1), ou un type défini dans SMIv1 ou SMIv2 ;

— **objets tabulaires** : une table est décrite comme une séquence de lignes qui sont elles-mêmes une séquence d'objets.

Objets scalaires

Les types utilisables sont des types simples ASN.1 ou des types dérivés de ceux-ci (tableaux **5** et **6**). En SMIv2, certains des types définis en SMIv1 ont été renommés, et de nouveaux types ont été introduits (tableau **6**).

Tableau 5 – Types simples ASN.1

Type	Description
INTEGER	Entier 32 bit
OCTET STRING	Chaîne d'octets
NULL	Vide
OBJECT IDENTIFIER	Séquence d'entiers identifiant un objet à l'aide de son positionnement à partir de la racine de la MIB (ex. : 1.3.6.1)
ENUMERATION OF INTEGER	Permet de représenter une énumération de valeurs par des entiers non nuls (ex. : on(1), off :2))

Tableau 6 – Types simples SMIv1 et SMIv2

Type SMIv1	Type SMIv2	Description
IpAddress	IpAddress	Représente l'adresse IP 32 bit par un OCTET STRING de longueur 4.
Counter	Counter32	Représente un entier non nul dont la valeur croît jusqu'à sa valeur maximum puis repasse à zéro (ex : compteur de trames émises sur une interface). Représenté sur 32 bit.
	Counter64	Compteur 64 bit : créé pour décrire les objets dont la valeur peut boucler à zéro en moins d'une heure s'ils sont représentés par un Counter : par exemple le nombre d'octets transitant sur un FDDI peut faire boucler le compteur correspondant en 5,7 min.
Gauge	Gauge32	Jauge : entier positif dont la valeur peut croître ou diminuer (ex. : % de CPU).
TimeTicks		Permet de compter une durée en centièmes de seconde à partir d'un temps origine.
Opaque	(obsolète en v2)	Permettait de passer une syntaxe ASN.1 arbitraire sous forme d'un OCTET STRING.
	BITS	Ensemble de bits nommés.

Les instances des objets scalaires sont identifiées dans la MIB avec un OID `{ nomobjet 0 }`, où *nomobjet* est la classe de l'objet. Par exemple, sysDescr.0 est l'instance de sysDescr.

Objets tabulaires

Ils sont décrits au moyen des types construits ASN.1 (tableau **7**).

Tableau 7 – Types construits ASN.1

Type	Description
SEQUENCE	Liste ordonnée d'autres types ASN.1
SEQUENCE OF TYPE	Liste ordonnée d'éléments du même type

SEQUENCE permet de définir les objets qui constituent les entêtes de colonne et SEQUENCE OF permet d'instancier ces objets. Les instances auront pour OID, dans ce cas, l'OID résultant de la concaténation des valeurs d'index.

Exemple : table SNMP ifTable

La table des interfaces est décrite dans mib-2 de la manière suivante :

```
ifTable OBJECT-TYPE
   SYNTAX SEQUENCE OF IfEntry
   ACCESS not-accessible
   ::= {interface 2}
ifEntry OBJECT-TYPE
   SYNTAX IfEntry
   ACCESS not-accessible
   STATUS mandatory
   INDEX { ifIndex }
   ::= { ifTable 1 }
IfEntry ::=
  SEQUENCE {
    ifIndex INTEGER,
    ifDescr DisplayString,
    ifType INTEGER,
   ... }
ifIndex OBJECT-TYPE
  SYNTAX INTEGER
  ACCESS read-only
   STATUS mandatory
   ::= {ifEntry 1 }
ifDescr OBJECT-TYPE
   SYNTAX DisplayString (SIZE (0..255))
   ACCESS read-only
   STATUS mandatory
   ::= {ifEntry 2 }
   ...
```

L'objet ifTable est décrit au moyen d'une séquence d'objets ifEntry qui ont pour type IfEntry.

Nota : I majuscule pour le type et i minuscule pour l'objet.

Le type IfEntry est une liste d'objets qu'on définit ensuite, constituée des objets simples ifIndex, ifDescr, etc. Les instances (éléments des lignes du tableau) sont définies à partir des objets identifiés dans la clause INDEX de ifEntry, c'est-à-dire à partir de ifIndex (INTEGER).

La figure **8** permet de visualiser le résultat d'une série d'interrogations SNMP permettant de retrouver les instances et les valeurs des objets « ifDescr » sur une machine dont la table ifTable a la structure décrite figure **7** : ici, les instances sont 1 et 2 et les valeurs des objets instanciés « Ethernet1 » et « Serial1 ».

2.3.2.4 Opérations autorisées sur les objets

En SMIv1, les opérations possibles sont « read-only », « read-write », « write-only » et « not-accessible » ; elles sont décrites dans la clause ACCESS. Il est possible, à l'aide d'une paire d'objets, de faire réaliser des actions par un agent SNMP : un objet en read-only permettra de consulter un état et un autre en read-write permettra de modifier cet état. Par exemple, dans la mib-2, ifAdminStatus permet de modifier l'état d'une interface d'un équipement (up, down, testing) et ifOperStatus permet de le consulter. La comparaison de l'état de ces deux variables permet de savoir si l'interface est dans l'état souhaité.

En SMIv2, les opérations possibles ont été modifiées. Elles sont décrites dans la clause MAX-ACCESS et peuvent prendre les valeurs not-accessible, accessible-for-notify, read-only, read-write, read-create. Cette nouvelle syntaxe permet en particulier de standardiser

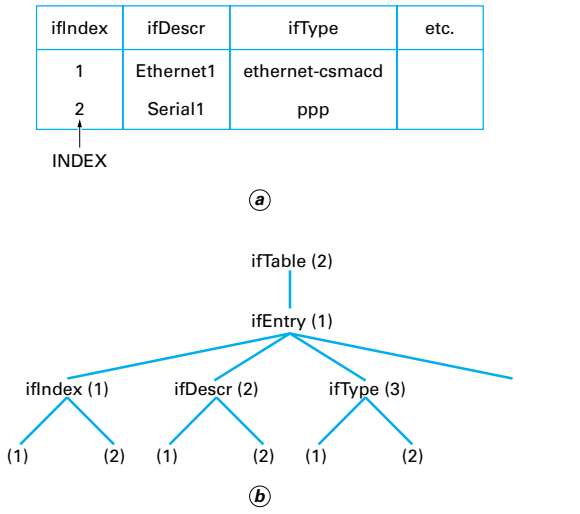

Figure 7 – Exemples de tables SNMP : ifTable

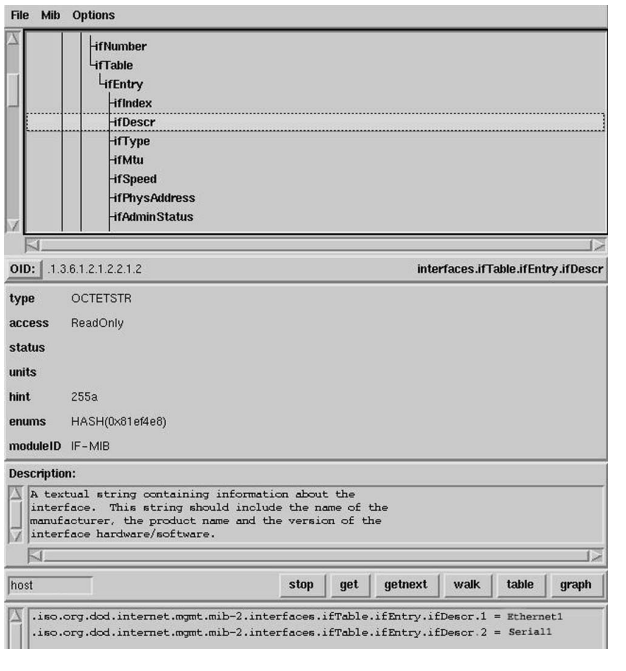

Figure 8 – Walk SNMP sur ifDescr avec le MIB browser NET-SNMP

les opérations d'ajout et de retrait de lignes dans les tableaux, qui n'étaient pas standardisées en SMIv1.

Dans chaque table pour laquelle il est possible d'ajouter ou retrancher des lignes, on trouve un objet de type Row Status, qui permet de créer une ligne (« createAndGo », « createAndWait »), de l'effacer (« destroy »), de la rendre active (« active ») ou inactive (« notInService »), ou encore qui indique son état (« active », « notInService », « notReady »). Un automate à états finis contrôle le passage d'un état à l'autre.

Pour créer une ligne, il faut :

— sélectionner une instance qui n'existe pas (soit au hasard, soit après avoir récupéré la liste des instances de la table) ;

— créer la ligne et éventuellement mettre à jour des objets de la ligne avec les valeurs désirées (il peut éventuellement y avoir des défauts aux valeurs des objets d'une ligne). On doit pour cela choisir la méthode de création : avec « createAndGo », la ligne est créée et activée immédiatement (s'il n'y a pas de problème lors de l'opération, comme des données manquantes ou une instance qui existe déjà, par exemple), alors qu'avec « createAndWait », la ligne est créée mais non activée et le gestionnaire peut modifier les valeurs des objets avant de réaliser l'activation ;

— activer la ligne, ce qui permettra par exemple de déclencher une opération sur l'agent SNMP, comme la mise en place d'une surveillance, la prise en compte d'une configuration, etc.

2.3.2.5 État de définition des objets

Cet état peut, en SMIv1, prendre les valeurs « mandatory », « optional », « deprecated » ou « obsolete ». En SMIv2, il peut prendre les valeurs « current », « deprecated » ou « obsolete », ce qui semble a priori moins riche. En réalité, SMIv2 fait usage de déclarations supplémentaires appelées *conformance statements* qui permettent de définir au sein d'un module les conditions d'implémentation des objets. Par exemple, selon qu'on a affaire à une interface réseau fonctionnant en mode caractère ou en mode bit, les objets appropriés pour calculer le trafic sont différents.

2.3.2.6 Description des notifications

Dans le standard SMIv1, la RFC 1215 (statut *informational*) décrit comment identifier les traps, qui sont les notifications du protocole SNMPv1. Elle indique toutefois que leur usage est vivement déconseillé, ce qui fait sourire quand on sait que beaucoup de produits qui se prétendent SNMP se limitent justement à l'émission de traps !

Un trap est identifié par :

— un OID ;

— un numéro de trap « generic-trap », qui permet d'identifier cinq traps « génériques » définis dans la mib-2, et de les séparer des traps « spécifiques » définis dans d'autres modules de MIB :

- coldStart(0) indique un reset complet de l'agent et probablement de l'équipement,
- warmStart (1) indique une réinitialisation,
- linkDown (2) indique le passage à l'état « down » d'une interface,
- linkUp (3) indique le passage à l'état « up » d'une interface,
- authenticationFailure (4) indique qu'une interrogation SNMP sur l'agent a été faite avec une mauvaise authentification,
- egpNeighborLoss (5) indique la perte d'un voisin EGP, quand ce protocole de routage est supporté par l'équipement,
- (6) indique qu'il s'agit d'un trap spécifique, identifié par un numéro « specific-trap » ;

— un numéro de trap « specific-trap », qui associé à l'OID permet d'identifier un événement spécifique.

Par ailleurs, peuvent être attachés au trap des objets de la MIB et leur valeur, qui permettront de mieux caractériser l'événement.

Exemple : le trap fictif suivant indique un changement d'état down d'une interface dont le numéro est remonté dans le trap (ifIndex). Il a pour OID myEnterprise, comme « generic-trap » 6 et comme « specific-trap » 2 : (RFC 1215, *statut informational*).

```
myEnterprise OBJECT IDENTIFIER ::= { enterprises 9999 }
myLinkDown TRAP-TYPE
        ENTERPRISE myEnterprise
        VARIABLES { ifIndex }
        DESCRIPTION
          « A myLinkDown trap signifies that the sending SNMP application entity recognizes a failure in one of the communications links represented in the agent's configuration. »
        ::= 2
```

La macro NOTIFICATION-TYPE permet de prendre en compte les différents types de notifications standardisées dans le protocole SNMPv2. Une notification est identifiée par un OID (de la forme <OID>.0.<numéro>, de manière à pouvoir faire une correspondance univoque avec les traps SNMPv1 et à assurer une migration plus facile depuis SNMPv1 vers SNMPv2 et v3. De la même manière que pour les traps SNMPv1, les notifications peuvent contenir des objets de la MIB, certains d'entre eux n'étant accessibles d'ailleurs que de cette manière (syntaxe accessible-for-notify) et pas par interrogation directe.

2.4 Protocoles SNMPv1 et SNMPv2

2.4.1 Primitives SNMP

Le protocole SNMP est basé sur des primitives qui permettent de réaliser des actions sur les objets de la MIB :

— les primitives de type GET : Get, GetNext, GetResponse (SNMPv1 et v2) et GetBulk (SNMPv2) permettent de lire des valeurs d'objets dans la MIB ;

— la primitive de type SET : Set (SNMPv1 et v2) permet d'écrire des valeurs d'objets dans la MIB ;

— les primitives de type EVENT : Trap (SNMPv1 et v2), SNMPv2-Trap, InformRequest et Report (SNMPv2) permettent de faire des notifications d'événements.

GET et SET

Pour les primitives de type GET et SET (figure **9**), on a affaire à un échange de type commande/réponse. Le command-generator (le manager SNMP en terminologie SNMPv1) envoie dans sa requête la commande d'interrogation ou de mise à jour, en indiquant la liste des objets concernés, et reçoit en retour une réponse qui donne la valeur de ces objets (après l'opération, en cas de SET), avec des codes d'erreur qui qualifient les problèmes éventuels.

Il est possible de lire ou d'écrire plusieurs valeurs d'objets dans la même requête. Le comportement est différent en cas d'erreur pour SNMPv1 et SNMPv2 : les commandes SNMPv1 sont atomiques, c'est-à-dire qu'il est nécessaire pour qu'elles aboutissent que toutes les sous-commandes puissent être réalisées par l'agent. C'est intéressant dans le cas d'un SET car cela permet de positionner dans une même requête toutes les commandes nécessaires à une action. Ça l'est moins dans le cas d'un GET car une seule valeur posant problème (par exemple, l'objet n'existe pas dans la MIB) empêche de récupérer l'ensemble des valeurs interrogées. SNMPv2 est plus souple et permet, en cas d'impossibilité d'opération GET sur des valeurs, de réaliser l'opération sur les autres et d'indiquer un code d'erreur noSuchObject ou noSuchInstance sur les valeurs posant problème, ce qui rend le protocole beaucoup plus efficace.

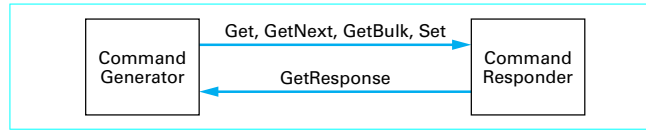

Figure 9 – Primitives GET et SET

GET et **SET** sont les primitives les plus simples car elles permettent de récupérer ou de mettre à jour des objets explicitement nommés avec leur OID. Il n'est possible de réaliser des opérations que sur les instances et non directement sur les objets tels qu'ils apparaissent dans la MIB. Ainsi, on interroge la valeur de sysDescr.0, on obtiendra une valeur, alors que sur sysDescr, le command-responder retournera une erreur de type noSuchName.

GetNextRequest est utilisée pour récupérer des valeurs d'instances dont on ne connaît pas l'index a priori. C'est par exemple le cas des valeurs d'objets contenus dans les tableaux. Une commande GetNext sur un objet de la MIB permet d'obtenir la valeur de l'instance suivante qui le suit dans l'ordre lexicographique des OID : cela permet de faire un parcours récursif des objets instanciés dans la MIB (opération communément appelée « walk »).

Exemple : sur la table ifTable donnée en exemple figures **7** et **8**, la séquence suivante permet de récupérer quelques valeurs de la table :

GetNext(ifTable) renvoie ifIndex.1=1 ; GetNext(ifIndex.1) renvoie ifIndex.2=2 ; GetNext(ifIndex.2) renvoie ifDescr.1=Ethernet1, GetNext(ifDescr.1)=Serial1, etc.

GetNext utilisé de manière judicieuse permet de récupérer des tables de manière relativement efficace (récupération par lignes).

- **GetBulk** est une primitive qui mélange dans une même requête l'équivalent d'un Get et d'un GetNext (figure **10**). Le premier paramètre (nonrepeater) indique le nombre de variables scalaires à récupérer ; le deuxième (max-repetition) le nombre d'instances successives à récupérer pour les autres objets (équivalent du nombre de GetNext qu'il faudrait passer pour avoir le même résultat), puis le dernier (varbind) contient la liste des variables à récupérer.

EVENT

Les primitives Trap et SNMPv2-Trap ne demandent pas d'acquittement : elles sont émises par le Notification-originator et réceptionnées par le Notification-receiver. Comme le protocole SNMP s'appuie en général sur la pile UDP/IP, la réception de ces notifications n'est pas garantie : elles ne sont acquittées ni au niveau applicatif, ni au niveau UDP (User Datagram Protocol). En général, ce type de communication est utilisé pour les notifications agent à manager. La primitive InformRequest, qui demande un acquittement par un GetResponse, est plutôt utilisée dans les communications manager à manager.

2.4.2 Codage et format des PDU

Les primitives sont codées en ASN.1 avec le code BER (Basic Encoding Rules).

Le format des **PDU** (Protocol Data Unit) **SNMP** suit globalement un même schéma, sauf pour les Traps SNMPv1 :

Type	Request ID	*x*	*y*	Variable bindings

« Type » est un octet identifiant la primitive (Get, Set, GetResponse, GetNext, GetBulk, SNMPv2-trap, Inform, Report).

« Request ID » est un entier identifiant la requête, qui permet de corréler commande et réponse.

Le couple (*x*, *y*) est (0, 0) dans le cas des Get, Set, GetNext, SNMPv2-trap, Inform. Dans le cas des GetResponse, *x* est le champ

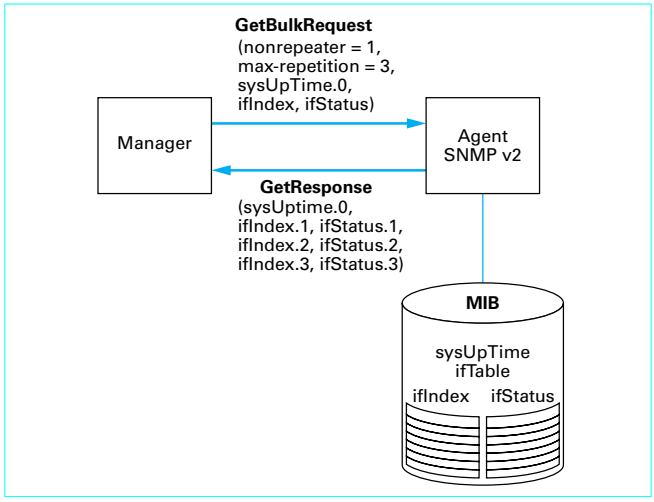

Figure 10 – GetBulk

« error status » qui permet de retourner le code d'erreur concernant la requête correspondante, tandis que *y* est le champ « error index » qui permet d'indiquer quelle est la première variable dans la requête qui est en erreur. Pour GetBulk, *x* représente la valeur « nonrepeater » tandis que y représente la valeur de « maxrepetition ».

« Variable bindings » représente la liste des variables, avec pour chacune un OID et une valeur (ou un code d'erreur dans le cas de SNMPv2).

Le format des **traps SNMPv1** est sensiblement différent, avec un même champ type indiquant qu'il s'agit d'un trap SNMPv1. Les champs « entreprise », « trap spécifique » et « trap générique » identifient le trap, l'adresse IP de l'agent est répétée, le timestamp et « variable bindings » indiquent respectivement la valeur du sysUpTime et la liste des variables avec leur valeur au moment de la génération du trap.

OxA4	Entreprise	Adresse agent	Trap générique	Trap spécifique	Time-stamp	Variable bindings

2.4.3 Format des messages SNMPv1 et v2C

Les messages SNMPv1 et v2C ont le même format :

n° de version	Communauté	PDU SNMP

Le numéro de version est 0 pour SNMPv1 et 1 pour SNMPv2.

Communauté est un OCTET STRING qui représente un mot de passe à valider par l'entité SNMP – qui s'en servira par exemple pour déterminer à quelles branches de la MIB le demandeur peut accéder et avec quels droits : c'est ce qu'on appelle des « MIB views ».

La PDU SNMP code l'opération SNMP.

2.4.4 Transport

Le transport privilégié de SNMP est UDP/IP. D'autres protocoles sont cependant utilisables, tels que OSI CLNS (Connexion-Less Network Service) et CONS (Connexion Oriented Network Service) et les protocoles propriétaires IPX et Appletalk. Dans le cas de TCP/IP, les messages sont envoyés sur le port UDP 161, sauf pour les traps SNMPv1 qui sont émis vers le port 162. Comme UDP n'assure pas la garantie de bonne délivrance des paquets, il est à la charge du manager SNMP (ou de l'entité SNMP en employant la terminologie de SNMPv3) de gérer des séquences de timeout et retry en cas de non réponse. Le choix d'un protocole non connecté est dû au fait qu'en cas de problèmes réseau (ce qui est un des cas d'utilisation d'un outil d'administration réseau), une connexion TCP a de bonnes chances d'être mise à mal, alors que des datagrammes UDP parviendront à remonter des informations, même parcellaires.

2.5 SNMPv3

2.5.1 Architecture

L'architecture de SNMPv1 et SNMPv2 est basée sur un modèle agent/manager : le manager accède aux informations contenues dans la MIB de l'agent en l'interrogeant par *polling* avec des commandes de type GET. Il peut aussi mettre à jour des variables sur l'agent avec des commandes de type SET. L'agent peut reporter des événements au manager avec des commandes de type traps. Dans ce modèle, on appelle *proxy* un agent SNMP qui agit pour le compte d'un ou de plusieurs autres équipements non-SNMP. Il s'agit d'une passerelle de niveau applicatif qui dialogue d'un côté en SNMP et de l'autre avec un autre protocole. Les proxys ont été essentiellement utilisés pour permettre à des équipements préexistants au standard SNMP de s'intégrer dans une solution d'administration SNMP (par exemple, des commutateurs X.25).

SNMPv3 reprend en les englobant SNMPv1 et SNMPv2. Le modèle d'architecture de SNMP est développé dans la RFC 2271, tandis que les RFC suivantes décrivent dans le détail les différents sous-systèmes conceptuels du modèle. Il est important de savoir que cette formalisation ne présume en aucun cas des implémentations qui pourraient être faites, et que beaucoup de modules présentent des options telles que SNMPv1 peut n'être vu que comme une option simple de SNMPv3.

SNMP entity

Ici, la différenciation agent/manager s'efface au profit de la notion plus générale d'entité SNMP (« snmp entity »), qui selon le cas aura un rôle de manager, d'agent ou les deux à la fois, en employant la terminologie SNMPv1. Ce modèle est un modèle de type *peer to peer*. Une entité SNMP est constituée d'un moteur (*engine*) et d'« applications » (figure **11**).

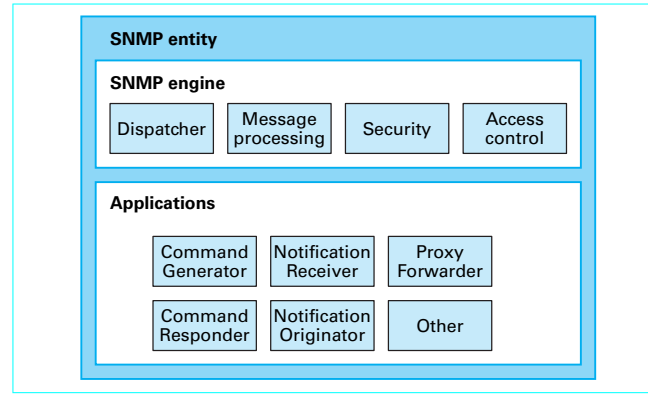

Figure 11 – Entité SNMP

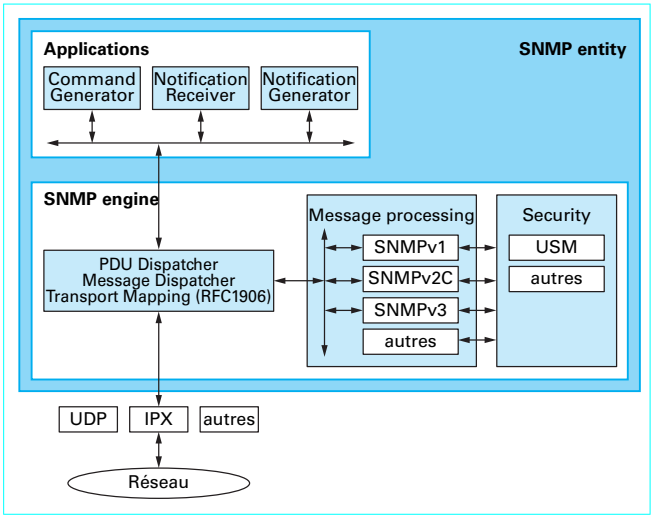

Figure 12 – « Manager » SNMP

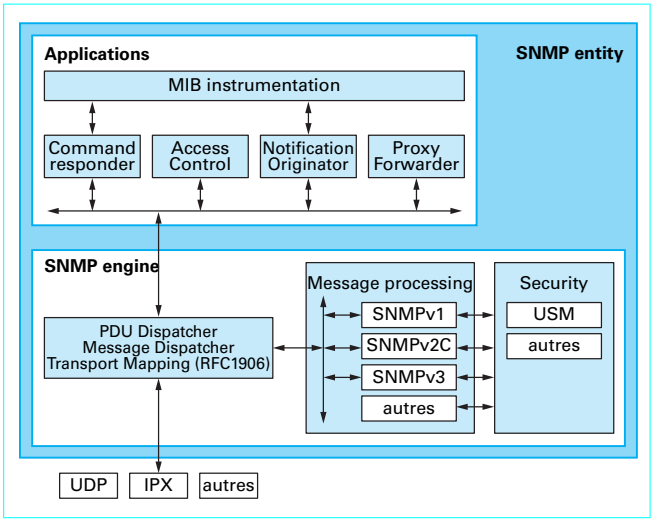

Figure 13 – « Agent » SNMP

- **SNMP Engine**

Comme son nom l'indique, le moteur assure toute la partie technique du travail.

Le *dispatcher* aiguille les messages qui arrivent du réseau vers le bloc de traitement de messages qui sera capable de le traiter (par exemple le sous-système SNMPv1 ou SNMPv3 du module *message processing*). Inversement, il redirige les messages en provenance de ce module vers le réseau, encapsulé dans le protocole adéquat (par exemple, UDP/IP ou IPX).

Le *message processing* s'occupe du décodage et de l'assemblage des messages. Des sous-systèmes gèrent les différents types de messages utilisables par l'entité (SNMPv1, SNMPv2C, SNMPv3).

Le module *security* traite de la sécurité des échanges, si elle est nécessaire : c'est lui qui est chargé de traiter la confidentialité des communications et l'authentification du correspondant. Plusieurs protocoles sont définis et utilisables et l'architecture SNMPv3 est extensible de manière à permettre l'utilisation d'autres modèles de sécurité que ceux prévus d'origine (USM User-based Security Model, est le modèle de sécurité de SNMPv3 qui répond de manière complète au problème, alors que le Community-based Security Model utilisé par SNMPv1 et v2 est plus qu'élémentaire).

Le module *access control* traite quant à lui le contrôle d'accès aux ressources de la contrôlée par l'entité SNMP, en fonction des prérogatives accordées. Un modèle est défini dans SNMPv3 : VACM (View-based Access Control Model) qui permet de définir des vues sur la MIB, vues sur lesquelles les opérations SNMP pourront être limitées. Par exemple, il sera possible d'accéder à des tables de configuration en lecture mais pas en écriture, ou encore il ne sera pas possible du tout d'accéder, même en lecture, à certaines parties de la MIB. Le module de MIB VACM permet de configurer ce contrôle de manière fine mais complexe.

- **Applications**

Une application est un sous-système propre aux fonctions de l'entité SNMP concernée. Selon le cas, les applications implémentées sont :

— des *command generators* : ils font des requêtes SNMP (et bien sûr analysent les réponses). Un manager SNMPv1 en est un exemple ;

— des *command responders* : ils sont susceptibles de répondre à des requêtes générées par des *command generators*. Un agent SNMPv1 en est un exemple ;

— des *notification generators* : ils émettent des notifications ;

— des *notification responder* : ils en reçoivent ;

— des *proxy forwarders* : ils aiguillent les informations en provenance d'un *engine* vers un autre, en fonction du contexte des informations transportées.

Selon les applications implémentées, on aura affaire à un agent, un manager, etc. Par exemple, un manager au sens SNMPv1 du terme implémentera les applications décrites figure **12**. Le fonctionnement de l'agent est décrit figure **13**.

Administration de SNMPv3

Ce modèle d'architecture souple et modulaire demande lui-même à être géré, en particulier pour ce qui est des configurations. Le standard définit donc pour chacune des parties des modules de MIB qui permettront de faire les paramétrages, etc.

2.5.2 Protocole SNMPv3

SNMPv3 reprend largement les spécifications de SNMPv2. En particulier, rien n'est changé au niveau des PDU. La primitive de type Report était définie en SNMPv2 mais son usage n'est explicité qu'avec SNMPv3, pour les communications *engine* à *engine*. Elles permettent en particulier de reporter les erreurs dans le traitement des messages SNMP (traité par « Message Processing Subsystem »), et pour gérer les synchronisations de temps.

Format des messages SNMPv3

msg Version	msgID	msgMax-Size	msg-Flags	msg Security-Model	Para-mètres de sécurité (opt.)	context EngineID	context Name	PDU

Les paramètres sont les suivants :

— le numéro de version, comme en SNMPv1 et v2, permet au *dispatcher* de savoir à quel sous-système du *message processing* envoyer le message ;

— msgID est l'identificant du message SNMP qui permet de corréler commandes et réponses ;

— msgMaxSize indique la taille maximum de message supportée par l'entité SNMP ;

— msgFlags définit la manière dont sont supportés les messages Reports, l'authentification et la privauté ;

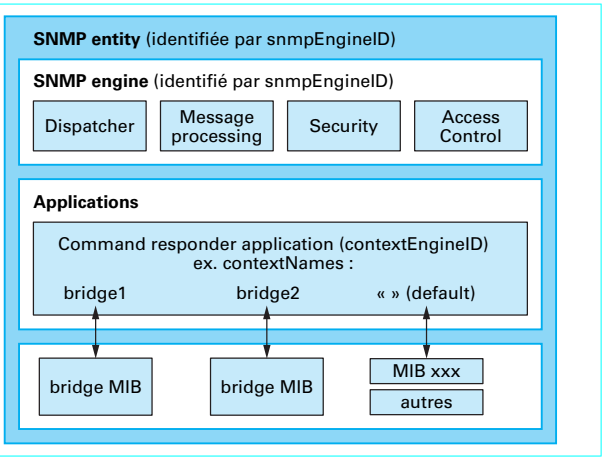

Figure 14 – ContextEngineID et contextName

— msgSecurityModel définit le modèle de sécurité utilisé, ce qui déterminera la signification du bloc suivant (paramètres de sécurité) et déterminera aussi si la suite du message sera cryptée ;

— contextEngineID et contextName permettent de déterminer le contexte de la requête : en SNMPv3, à l'OID de la MIB qui identifie un objet se rajoute la notion de contexte qui permet d'avoir plusieurs instances de MIB sur une entité SNMP, qui seront identifiées par un « contexte ». Par exemple, pour un équipement implémentant deux ponts, le contextName permettra d'identifier la MIB du pont d'où on désire interroger (figure **14**).

Sécurité

C'est le progrès principal de SNMPv3 : SNMPv1 et SNMPv2C offrent un modèle de sécurité réseau à base de noms de communauté, qui n'assure que l'authentification. Il est de plus très facile de le détourner puisque les noms de communauté circulent en clair sur le réseau. Ce modèle reste une option dans SNMPv3.

Un modèle complet est proposé avec USM (User-based Security Model), qui s'appuie sur les protocoles HMAC-MD5-96 (basé sur MD5) et HMAC-SHA-96 (basé sur SHA) pour l'authentification et CBC-DBS (encryptage symétrique) pour la privauté, sachant que d'autres protocoles peuvent être utilisés. Ces protocoles sont basés sur un système de clés et sont configurés à l'aide d'un module de MIB.

3. Perspectives

3.1 SNMP

Le standard SNMPv3 donne un cadre précis d'administration et suffisamment d'ouvertures pour permettre d'intégrer des extensions qui seraient nécessaires, comme par exemple de nouveaux modèles de sécurité.

Les évolutions actuelles de SNMP se font donc plus autour de la MIB ; celle-ci est enrichie progressivement par de nouveaux modules :

— pour prendre en compte les standards Internet (par exemple, DSMON est la MIB RMON pour diffserv) ;

— pour s'étendre au-delà du réseau vers les services applicatifs ou système (RFC 2591 : Definitions of Managed Objects for Scheduling Management Operations, RFC 2287 : Definitions of System-Level Managed Objects for Applications, RFC 2594 : Definitions of Managed Objects for WWW Services, etc.) ;

— un des domaines dans lesquels la définition de nouveaux modules est susceptible d'intéresser la plupart des administrateurs est celui de la gestion de la QoS sur lequel des modules de MIB sont en cours de définition. Par exemple, la MIB APM (Application Performance Measurement MIB) définit des objets pour mesurer les performances applicatives telles que perçues par les utilisateurs finaux, avec disponibilité et temps de réponse, en s'appuyant sur une base RMON2 pour la définition des protocoles ;

— pour standardiser des fonctionnalités utiles et souvent implémentées dans des modules de MIB privées. C'est par exemple le cas des MIB Ping, Traceroute et Lookup qui devront permettre à un manager de demander à un agent de faire des ping, traceroute et requêtes DNS. C'est aussi le cas de la MIB « Physical Topology » PTOPO-MIB) qui permet de décrire la topologie physique d'un réseau (c'est-à-dire les interconnexions des équipements entre eux au niveau des ports) : actuellement, chaque constructeur a sa solution propriétaire qui lui permet de construire des applications permettant de donner de telles vues, dans un environnement monoconstructeur.

Il reste à voir comment le marché réagira et si les standards seront adoptés, en particulier pour SNMPv3 dont la mise en œuvre et l'utilisation restent complexes, comparées à celles de SNMPv1 et SNMPv2C.

3.2 Plates-formes et outils

La plupart des outils s'orientent maintenant vers une interface Web (Java ou HTML), ce qui donne le choix du poste de travail client et permet une intégration relativement aisée des outils.

On retrouve aussi cette tendance sur les équipements eux-mêmes qui embarquent désormais un serveur Web et permettent à un administrateur de se connecter directement sur l'équipement avec une ergonomie meilleure que celle d'une console telnet. Cela présente toutefois des inconvénients si les liens réseau sont de faible débit ou de faible qualité, ce qui est souvent le cas lors de problèmes ; telnet ou SNMP deviennent alors plus intéressants.

3.3 CIM et WBEM

Le besoin d'une base standard de programmation dans le domaine de l'administration est une réalité pour les développeurs. SNMP n'adresse pas le problème des API qui permettent d'écrire des applications « end-user » (ce n'est pas le rôle de l'IETF que de définir des API : il se limite aux standards d'interopérabilité). L'API de l'Open Group XMP a eu un succès mitigé.

Le DMTF (Distributed Management Task Force), consortium d'éditeurs et de constructeurs informatiques, a établi le standard DMI (Desktop Management Interface), qui est au hardware PC ce que SNMP est aux équipements réseaux. Il travaille maintenant sur la finalisation des standards CIM (Common Information Model) et WBEM (Web Based Enterprise Management) dont l'ambition est de donner un cadre de développement aux applications d'administration des réseaux d'entreprise, permettant d'attaquer des *providers* tels que SNMP pour les réseaux, DMI pour les PC, les environnements Microsoft (WMI : Windows Management Interface), etc., à partir d'un même environnement WBEM (figure **15**) :

— CIM a un schéma objet extensible qui permet la représentation de l'environnement administré ;

— il est utilisé par un manager d'objets accessible via API ou échanges XML sur HTTP par les applications de management ;

— il s'appuie sur des fournisseurs d'objets qui prennent en compte les différents protocoles d'administration.

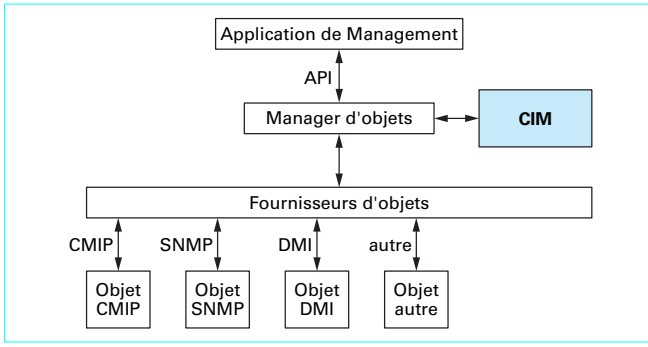

Figure 15 – WBEM

Contrairement à SNMP, on a ici un modèle réellement objet, qui est d'ailleurs représenté par une notation dérivée d'UML. Cet environnement a le soutien de la plupart des grands éditeurs de logiciels d'administration ainsi que de Microsoft. Toutefois, l'environnement est encore récent et n'a pas fait ses preuves.

4. Conclusion

SNMP est le standard incontournable dans le domaine de l'administration de réseaux d'entreprise. Son utilisation s'est étendue au-delà, dans le monde du système et des applications. Les limitations des versions 1 et 2 sont comblées avec la version 3. Ce dernier standard demande toutefois plus de travail d'implémentation et de mise en œuvre, et il restée voir si le marché va l'adopter ou conserver les solutions plus simples de SNMPv1 et SNMTv2C.

Le standard CIM et WBEM, s'il tient ses promesses, permettra de construire des applications d'administration puissantes et interopérables. Il est par contre douteux qu'il remplace à court terme SNMP au niveau des équipements réseau, ce protocole ayant fait ses preuves.

Dans tous les cas, établir une administration de réseau efficace demande un travail non négligeable en terme de choix et de mise en place d'outils, d'organisation et de formation et d'implication des personnels.