Université de Montréal

Système de configuration des politiques de gestion de réseaux

par

Diéfadima DIOUBATÉ

Département d'informatique et de recherche opérationnelle

Faculté des Arts et des Sciences

Mémoire présenté à la faculté des études supérieures

en vue de l'obtention du grade de Maître ès sciences (M.Sc.)

en informatique

décembre 2000

Université de Montréal
Faculté des études supérieures

Ce mémoire de Maîtrise intitulé

Système de configuration des politiques de gestion de réseaux

Présenté par
Diéfadima DIOUBATÉ

Membres de jury
Présidente : Mme Rachida Dssouli
Directeur : Raouf Boutaba
Membre : Mme Brigitte Kerhervé

À mon fils Bai, à mon mari Abdul et à mes parents

Dioubaté-Condé-Koulibaly.

Sommaire

Les systèmes de réseau actuels constituent un ensemble complexe d'interconnexions de réseaux de plusieurs institutions. Le réseau de chaque institution est composé d'une variété de dispositifs, systèmes, services et applications. La structuration, l'utilisation abondante, et l'évolution rapide de ces réseaux, nécessitent leur gestion dynamique, évolutive et intégrée.

- Gestion dynamique pour permettre aux administrateurs de réseau d'effectuer la configuration des éléments gérés en ligne.
- Gestion évolutive pour leur permettre d'adapter les mêmes outils de gestion après une modification de l'architecture des réseaux.
- Gestion intégrée pour leur permettre de gérer la diversité des éléments composant ces réseaux avec un minimum d'outils de gestion.

De plus, avec le développement des nouvelles technologies de l'information, les réseaux s'avèrent complexes, larges et instables. Gérer de tels réseaux, requiert l'utilisation d'un système de gestion séparant les caractéristiques de gestion des unités de gestion devant les appliquer dans le réseau, ainsi qu'une configuration efficace non individuelle de ces caractéristiques dans les composants matériels, logiciels et services interagissant dans le réseau.

Il s'agit en l'occurrence, de doter les administrateurs, d'outils de gestion intégrée. Ce cas est d'importance quand on est en face de réseaux hétérogènes, utilisant plusieurs systèmes de gestion pour surveiller et maintenir l'état de fonctionnement des éléments gérés et la répartition des ressources du réseau.

Les politiques de gestion de réseau représentent les caractéristiques opérationnelles

des composants du réseau, que les administrateurs de réseaux peuvent utiliser pour contrôler les activités des objets gérés. Un des aspects importants de gestion de réseau basée sur les politiques est la séparation du stockage de politiques, des mécanismes de contrôle de politiques.

Ce mémoire de maîtrise décrit l'architecture d'un système de gestion de politiques de gestion de réseaux. Le système de gestion de politiques défini dans ce document, permet dans une structure hiérarchique :

- La gestion des politiques : la gestion de la configuration des paramètres de contrôle et/ou la visualisation des statistiques de fonctionnement des éléments gérés du réseau.
- Le stockage des politiques.
- Le contrôle des politiques de gestion : la reconfiguration automatique des éléments gérés du réseau.

Ces trois fonctions sont accomplies respectivement par un ensemble d'application de gestion de politiques, un système de stockage de politiques et un ensemble de mécanismes de contrôle de politiques.

Aux termes de cette thèse, nous avons implanté une application intégrée de gestion de politiques de gestion de réseaux. Dans cette application intégrée de gestion, nous utilisons un gestionnaire SNMP et un client LDAP. Le gestionnaire SNMP nous permet de collecter des statistiques sur le fonctionnement des éléments gérés du réseau et de configurer la valeur des attributs dans la base d'information de gestion ; et le client LDAP nous permet de configurer, de visualiser et de rechercher les politiques de gestion dans un serveur LDAP.

Remerciements

Je tiens à exprimer ma profonde gratitude à mon directeur, le Pr. Raouf BOUTABA, pour avoir dirigé ce mémoire avec patience. La disponibilité qu'il m'a accordée, a été un grand stimulant pour cette recherche.

De même, je remercie le Programme canadien des Bourses de la Francophonie (PCBF), pour son aide financière durant la période de cette recherche. Je remercie les autorités de l'Université de Conakry, particulièrement, le Pr. Mohamed Lamine Kaba et le Dr Binko M. Touré, pour leur confiance. Je remercie les familles Mané Yaya-Kadiatou, Diabaté Mme Fatou Sidibé (Awah et Ibrahim), pour leur soutient moral et spirituel. Grand merci à Dr Diallo Abdourahamane pour ses commentaires judicieux.

Je ne saurais terminer ce mémoire, sans remercier toute ma famille, particulièrement, mon époux et mes parents Dioubaté-Condé-Koulibaly, qui n'ont ménagé aucun effort, pour rendre cette étude de maîtrise une réalité.

Table des matières

Table des figures

Introduction

Le développement de l'Internet, son intégration dans tous les domaines de la vie sociale et son utilisation comme moyen de communication ont changé la configuration des réseaux (LAN, MAN, WAN, etc.) en systèmes distribués très larges et très complexes. Ces systèmes sont composés d'ensemble de réseaux de plusieurs départements ou institutions, dont chaque réseau peut être à son tour constitué d'une variété de dispositifs, systèmes, services et applications.

La gestion de ces réseaux est une tâche difficile de plusieurs niveaux, qui nécessite une structure d'application de gestion intégrée, dans une architecture de système de gestion extensible et facilement adaptable. Une structure d'application de gestion intégrée combine la gestion de tous les groupes d'éléments (utilisateurs, applications et systèmes d'opération, informations, systèmes de transport et services) présents dans un réseau. La gestion de chacun des groupes d'éléments constitue un niveau de gestion impliquant un ensemble de fonctionnalités de gestion.

La gestion de la configuration du réseau est une tâche fondamentale de gestion sur laquelle est basé le bon fonctionnement du réseau. Un des aspects critiques de la gestion de la configuration est la gestion des données de configuration (le stockage et la distribution).

Pour faciliter la gestion de la configuration, les informations génériques de gestion des composants peuvent être stockées de façon centralisée, puis distribuées dans le réseau lorsque cela est nécessaire. Certaines informations de gestion nécessitant un stockage local peuvent être maintenues localement dans les éléments gérés du réseau.

En utilisant les systèmes standards de gestion de réseau IP (par exemple, SNMP), les administrateurs ne peuvent pas séparer l'administration des informations de gestion,

de celle des mécanismes de contrôle. Le système qui permet cela est celui de gestion basée sur les politiques.

Les politiques représentent les caractéristiques opérationnelles des composants du réseau, que les administrateurs utilisent pour contrôler les activités de leurs réseaux. Elles sont distribuées dans le réseau pour faciliter certaines tâches des composants du réseau (tel que, le routage avec qualité de service) ou certaines tâches des agents de gestion (comme le filtrage des informations de surveillance).

Dans le cadre de notre projet de recherche, nous développons un système de gestion de politiques de gestion de réseaux et implémentons une application de gestion de politiques. Ce système facilite la configuration collective et le suivi des paramètres de contrôle des éléments gérés du réseau et la reconfiguration automatique de ces éléments.

Notre application de gestion de politiques permettra aux administrateurs ayant des systèmes distribués larges et complexes, de configurer les politiques de gestion stockées dans une base de donnée centrale distribuée (par exemple, le service de répertoire Lightweight Directory Access Protocol) ou dans une base de donnée locale (par exemple, la MIB).

L'idée sous-jacente à l'utilisation de politiques est de faciliter l'évolution en ligne du réseau et de son système de gestion intégrant dynamiquement de nouvelles politiques. Le système de gestion de politiques ciblé dans cette thèse est développé sur la base d'une gestion par domaine qui, de plus, intègre la technologie d'agents mobiles.

L'utilisation des agents mobiles, principalement pour la mise à jour des informations de gestion dans certaines conditions d'opération, permettra d'une part de réduire le nombre de messages de gestion transmis dans le réseau et d'autre part de réagir plus rapidement à certains événements qui ont lieu dans le réseau.

Notre application de gestion de politiques manipulera les politiques de gestion à différents niveaux d'abstraction par sous réseaux (exemple: les politiques de gestion propres au réseau du département des finances ou réseau privé virtuel) ou par service (exemple: les politiques de gestion pour assurer la qualité de service dans le réseau ou renforcer la sécurité dans un réseau IP).

Dans notre implantation, nous utilisons deux stockages différents de politiques de gestion: un stockage local appelé Simple Network Management Protocol Management Information Base (SNMP-MIB) pour les politiques dans les éléments gérés du réseau, et un stockage central distribué appelé service de répertoire (LDAP directory) pour les politiques communes.

Notre application de configuration de politiques permet à un administrateur de réseau d'utiliser une interface graphique pour facilement surveiller et contrôler l'état des éléments gérés du réseau avec SNMP, de visualiser et modifier les politiques de gestion du réseau en utilisant LDAP.

Le chapitre 1 de ce mémoire est une introduction à la gestion de réseau. Le chapitre 2 décrit le système de configuration de politiques et le protocole d'accès distribué (LDAP), tandis que dans le chapitre 3, nous détaillons notre architecture. Le chapitre 4 présente l'implémentation de l'application de configuration et le chapitre 5 conclut ce mémoire.

Chapitre 1

Gestion de réseaux et de Systèmes distribués

1.1 Introduction

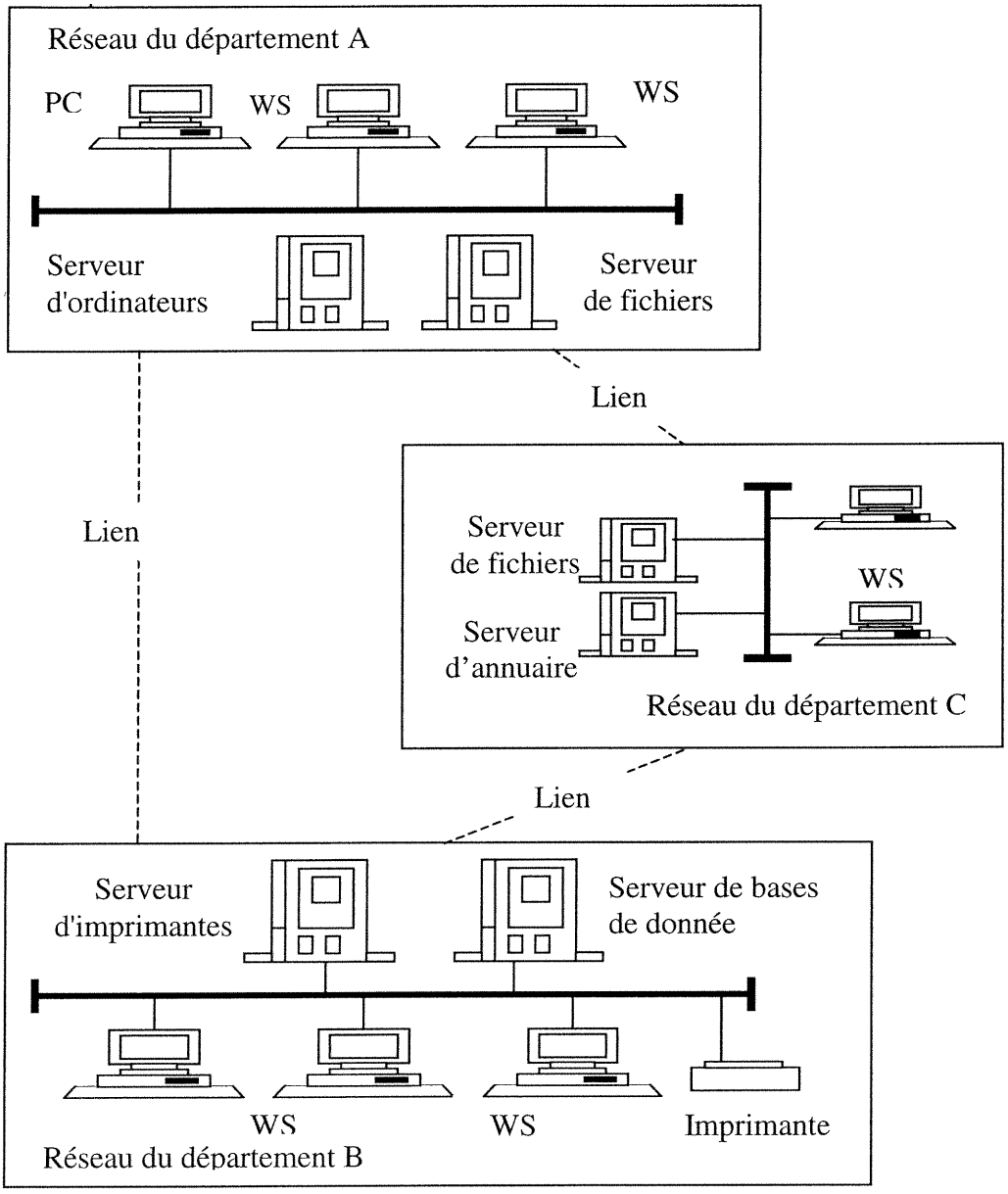
Les administrateurs de réseaux et de systèmes distribués modernes (figure 1.1) sont généralement confrontés à des difficultés de suivi et de contrôle de leurs architectures, qui sont de grande taille et composées de divers objets. Pour bien gérer les objets multiples et multiconstructeurs, locaux et distants de ces architectures, ils ont besoin de systèmes de gestion basés sur des normes et sur l'intégration de ces normes.

L'intégration des normes facilite la gestion intégrée et hiérarchique, dont chaque niveau hiérarchique peut être responsable de la gestion d'un ensemble spécifique d'éléments (applications, données, réseau et ressources du système, système de transport, structure organisationnelle et administrative de l'entreprise) [Hei 99], [Noe 97].

Dans [Hei 99], sont définis cinq niveaux de gestion, qui sont : la gestion de l'entreprise, la gestion des applications, la gestion des informations, la gestion des systèmes et la gestion du réseau, figure 1.2.

- La gestion du réseau s'occupe de la gestion des équipements et des services de communication (pont, routeur, concentrateur, multiplexeurs, etc.).

- La gestion des systèmes est la gestion des logiciels de réseau (serveurs, système de fichiers, utilisateurs, modules de logiciel, processus, etc.).



Légende
WS = Station de travail

Figure 1.1 Système distribué moderne

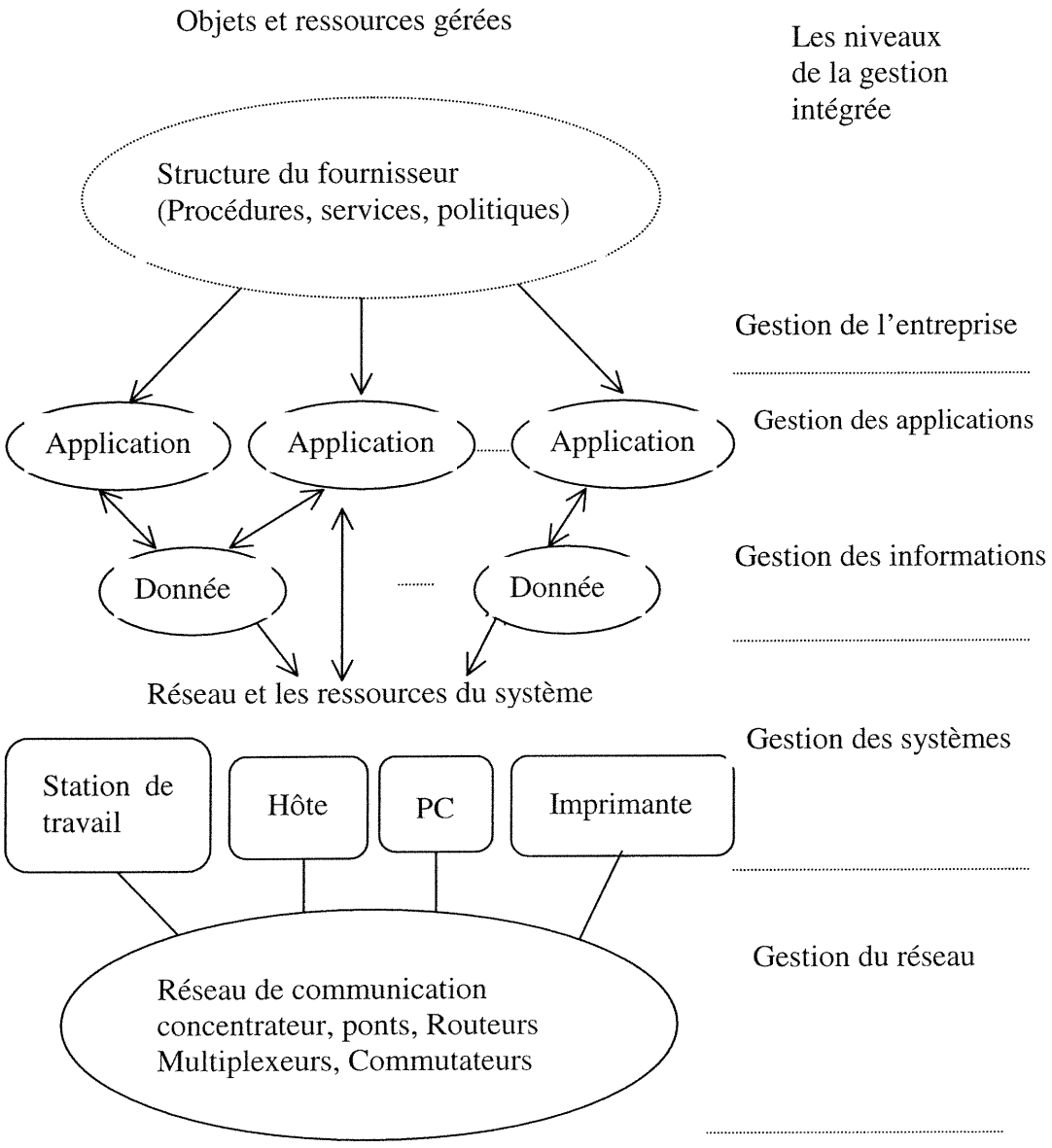


Figure 1.2[1] Niveaux de gestion intégrée

[1] Figure 1.2 de [Hei 99]: Levels of integrated management

- La gestion des informations s'occupe de la conception, la maintenance, et l'assurance de consistance et d'accessibilité des données.

- La gestion des applications consiste en celle des applications distribuées et services disponibles.

- La gestion de l'entreprise établit et gère les politiques de l'entreprise. Elle gère aussi le personnel, les tâches financières, la technologie et la production de l'entreprise.

De manière générale, ces différents niveaux de gestion implémentent les fonctions de gestion normalisées par l'organisation internationale de standardisation (ISO).

Dans ce chapitre, nous introduisons la gestion de réseau et de systèmes distribués. En particulier, nous présentons les différents systèmes de gestion développés pour les différentes architectures de réseau et systèmes distribués existants.

1.2 Modèles et normes de gestion de réseau

Un système de gestion standard, est un système dont l'architecture a une orientation multivendeurs et multisystèmes pour une gestion intégrée dans un environnement distribué et hétérogène. Dans un système de gestion standard, sont généralement définis quatre modèles de référence de gestion, qui sont: le modèle informationnel, le modèle organisationnel, le modèle de communication, et le modèle fonctionnel.

Dans les sections qui suivent, nous décrivons de manière générique ces différents modèles, puis chaque modèle d'un système de gestion spécifique sera décrit dans la section dédiée à ce système de gestion.

1.2.1 Le modèle fonctionnel

Quelle que soit la complexité d'un système distribué, quel que soit son type ou sa taille, le besoin de l'administrer existe toujours. La difficulté reste à déterminer ce que doit faire une application de gestion, l'objectif étant de pouvoir surveiller et contrôler toutes les ressources et tous les composants du système.

Dans le modèle fonctionnel de la norme de gestion, sont définies cinq fonctions de gestion, qui sont : la gestion des fautes, la gestion de la configuration, la gestion des informations comptables, la gestion de la performance et la gestion de la sécurité, communément abrégées par FCAPS.

Pour le développement d'une application de gestion, on peut se baser sur ces cinq fonctions de gestion. Cependant, le besoin en administration étant étroitement lié aux caractéristiques du réseau à gérer ou de l'organisation qui l'emploie, tous les administrateurs n'ont pas nécessairement besoin de toutes ces fonctions. Pour certains administrateurs, c'est la sécurité qui prime alors que pour d'autres c'est la comptabilité. Une application de gestion doit pouvoir effectuer une de ces fonctions ou une intégration de deux ou plusieurs.

1.2.1.1 La gestion des fautes

La gestion des fautes consiste en la localisation, l'isolation et la réparation des fautes pour maintenir un fonctionnement normal du système. Dans un environnement complexe, large, et hétérogène, les fautes peuvent provenir de n'importe quelle source : du matériel (câble, routeur, concentrateur, etc.) et/ou du logiciel.

1.2.1.2 La gestion de la configuration

La gestion de la configuration est une fonction de contrôle fondamentale et critique. Elle est un processus englobant l'obtention des informations d'initialisation des objets gérés du système et l'utilisation de ces informations pour initialiser et contrôler l'installation de tous les composants [Lei 93].

Les informations d'initialisation des objets gérés peuvent être l'état de fonctionnement d'un système, des paramètres d'autorisation, des entrées de table de routage, des noms de serveurs et répertoires, des aspects d'installation physique de logiciel ou matériel, des paramètres de filtrage d'un pont, etc.

Les mécanismes impliqués dans la configuration des composants sont [Sta 93]:

- Collecter et propager les données de l'état courant des ressources. Communiquer les changements (localement initiés ou survenus par suite d'événements imprévisibles) aux services de gestion en utilisant les protocoles normalisés.

- Initialiser ou interrompre le fonctionnement des objets gérés.

- Configurer et modifier la configuration des paramètres liés aux composants et logiciels des couches du réseau.

- Affecter des noms aux objets et groupes d'objets.

On distingue deux types de configuration : la configuration statique et la configuration dynamique.

- La configuration statique est responsable de la mise en place initiale du système. Pour ce faire, l'administrateur a besoin d'interrompre l'opération du système pour entrer les nouvelles valeurs. Les valeurs nouvellement entrées sont validées après un redémarrage ou un rechargement du système.

- La configuration dynamique (reconfiguration) est l'ensemble des modifications qui surviennent pour réorganiser un système (par suite d'une erreur, d'un ajout ou retrait d'éléments). Dans la configuration dynamique, les mises à jour s'effectuent en ligne ou sans interruption du système.

La gestion de la configuration permet à l'administrateur d'accéder rapidement aux données de configuration des composants, de comparer la configuration en vigueur à celle stockée dans le système, de les reconfigurer et de faire l'inventaire.

L'inventaire maintenu dans une base de donnée décrit pour chaque composant, le fabriquant et la personne de contact en cas de problème, figure 1.3 [Lei 93]. Dans l'inventaire, l'administrateur peut trouver la réponse à certaines questions (par exemple, l'identificateur du composant, les différentes versions d'un système d'exploitation, etc.).

Parmi les outils de base de configuration, il y a les outils d'édition et de stockage des informations de configuration du système, comme les comptes des utilisateurs, les adresses assignées aux composants du réseau, les numéros de série, l'emplacement physique des composants et autres informations pertinentes.

Certains outils plus sophistiqués permettent d'effectuer la collecte et le stockage automatique d'informations de gestion des composants, la recherche et modification des informations stockées, la comparaison des configurations courantes à celles stockées dans le système.

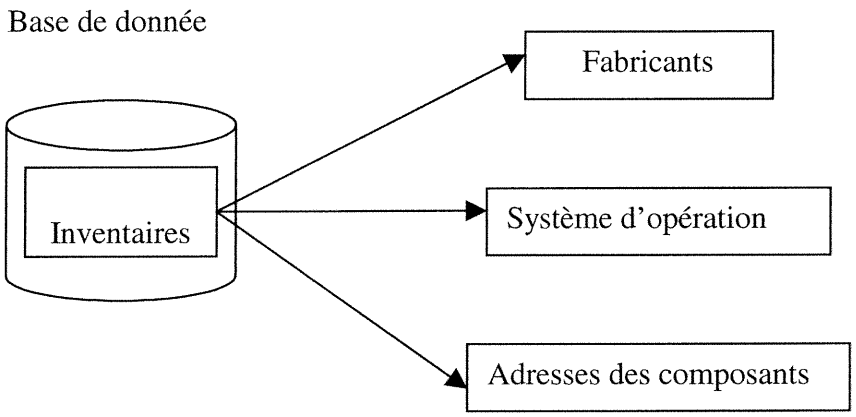


Figure 1.3 Base de donnée d'inventaires

L'utilisation des systèmes de gestion de base de données relationnelle pour le stockage des informations de gestion des composants offre plusieurs avantages entre autres, l'économie d'espace de stockage et la création de relations complexes entre les informations de gestion.

La création des relations entre les informations est utile, par exemple pour retrouver les informations du fabriquant à partir de celles des composants. L'inconvénient des systèmes de gestion de base de donnée relationnelle est leur complexité, et surtout leur caractère propriétaire.

Ces dernières années ont connu l'émergence d'outils plus sophistiqués et mieux adaptés à la gestion de la configuration. Ces outils, dont en particulier les systèmes à base de politiques et les services de répertoire font aujourd'hui l'objet de normalisation au sein d'organismes internationaux comme l'IETF. Ils permettent de faciliter les tâches de configuration et surtout fournissent un plus haut degré d'automatisation.

1.2.1.3 La gestion des informations comptables

La gestion des informations comptables concerne l'analyse et l'exploitation des informations relatives à l'utilisation des ressources et à l'utilisation des services supportés par le réseau. La gestion des informations comptables est très importante dans les réseaux de télécommunications.

Elle fournit les procédures permettant [Hei 99]:

- D'informer les utilisateurs, des coûts encourus.
- De limiter les comptes d'usage des ressources gérées.
- De combiner des coûts si plusieurs ressources sont utilisées pour offrir le service de communication.

1.2.1.4 La gestion de la performance

La gestion de la performance permet à l'administrateur d'évaluer régulièrement et éventuellement d'améliorer l'utilisation globale du réseau. Elle comprend toutes les mesures nécessaires pour assurer que la qualité de service (QoS) est conforme au contrat de service entre le fournisseur et l'utilisateur de service.

Elle comprend en particulier [Hei 99]:

- L'établissement des paramètres et métriques de la qualité de service.
- Le suivi de toutes les ressources pour une bonne optimisation des performances.

- La réalisation des mesures et l'orientation des analyses pour la prévision des échecs.

- L'évaluation d'historique.

- Le traitement des données de mesure et la compilation des rapports de performance.

- La planification de la performance et de la capacité. Cette planification comporte la fourniture des modèles analytiques ou des méthodes de simulation et de prévision pour le déploiement de nouvelles applications, les mesures de réglage et les changements de configuration.

1.2.1.5 La gestion de la sécurité

Les ressources et les informations d'une entreprise doivent être protégées contre les attaques extérieures à l'entreprise et les mauvaises utilisations de l'intérieur. La gestion de la sécurité intervient pour contrôler les accès et vérifier les droits d'accès au réseau et services, pour détecter les violations des règles de sécurité et pour accomplir les tâches de gestion de manière sécurisée.

Ainsi, elle doit supporter l'autorisation, le contrôle d'accès, le cryptage des données, l'authentification de l'utilisateur et l'intégrité des données.

1.2.2 Modèle organisationnel

Le modèle organisationnel définit les différentes entités du système de gestion, leurs rôles et les interactions entre elles. Il y a généralement deux entités principales dans ce modèles: l'agent et le gestionnaire (Manager), figure 1.4, [Sta 99] [Slo 94].

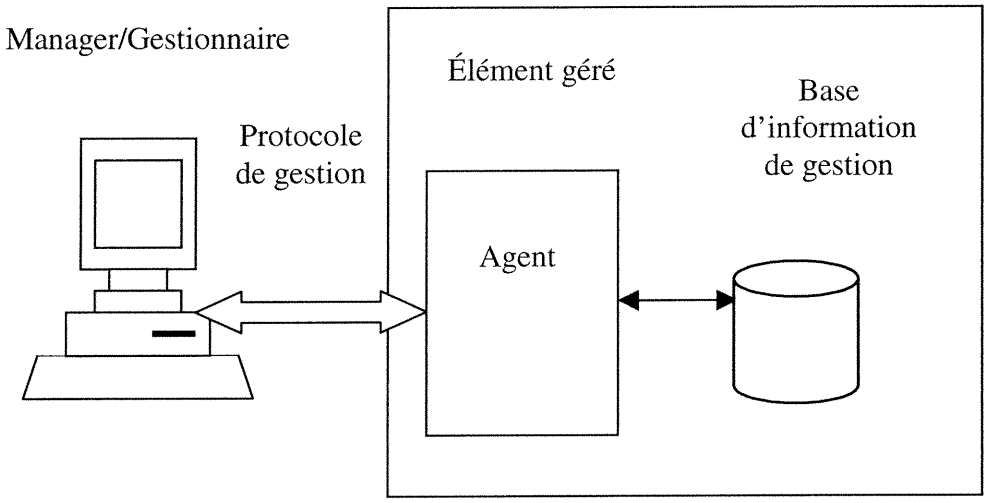


Figure 1.4 Architecture du système de gestion

- Le gestionnaire est un processus responsable de la réalisation d'une ou plusieurs fonctionnalités de gestion. Le gestionnaire s'occupe de l'initialisation des requêtes de gestion pour leur acheminement vers les agents de gestion, de la réception des réponses d'opérations de gestion ou de notifications d'événements.

- L'agent est un processus qui s'exécute sur les éléments gérés du système (par exemple, serveur, routeur, concentrateur ou service). L'agent est considéré dans le système de gestion comme le collecteur de statistiques de communication et d'activités relatives à son objet géré. Il enregistre localement les statistiques qu'il collecte de l'élément géré, reçoit les requêtes de gestion du gestionnaire, traite ces requêtes et retourne les résultats au gestionnaire.

Les agents et gestionnaires sont organisés dans le système de gestion sous différentes formes. Généralement, il existe deux formes d'organisation de ces deux entités: une

organisation centralisée (architecture de gestion centralisée) et une organisation distribuée (architecture de gestion distribuée).

1.2.2.1 Architecture de gestion centralisée

Une architecture de gestion centralisée est une architecture dans laquelle il n'y a qu'un seul gestionnaire et un ou plusieurs agents. Le gestionnaire communique directement avec les agents pour contrôler et surveiller les activités du réseau. Ce genre d'architecture est aussi appelé architecture à deux tiers, figure 1.5.

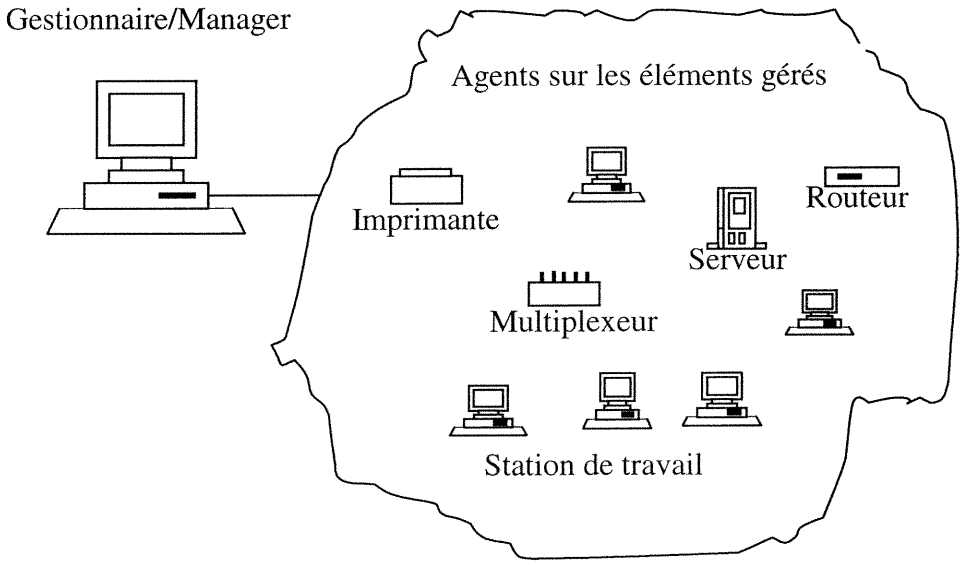


Figure 1.5 Gestion centralisée

Dans un réseau de grande taille, l'architecture centralisée induit un coût élevé des communications agents/gestionnaire. Elle est aussi moins tolérante aux pannes. En effet, si la station de gestion à un problème de fonctionnement, tout le système de gestion est affecté. De plus, dans une telle situation, la station de gestion constitue un goulot d'étranglement.

1.2.2.2 Architecture de gestion distribuée

Une architecture distribuée de gestion est une architecture dans laquelle il existe plusieurs gestionnaires permettant la hiérarchisation des gestionnaires, et/ou leur coopération.

Dans la gestion hiérarchique, figure 1.6(a), il y a un gestionnaire principal et un gestionnaire intermédiaire pour chaque niveau ou domaine de gestion, qui ne s'occupe que de la gestion de son niveau ou domaine ; les informations pertinentes du réseau global étant acheminées vers le gestionnaire principal.

Dans la gestion coopérative, figure 1.6(b), les gestionnaires dirigent la gestion en coopération, non pas de façon hiérarchique, mais plutôt d'égal à égal (par exemple, entre gestionnaires du même niveau hiérarchique).

Dans l'architecture distribuée, certains sont gestionnaires et agents à la fois. La gestion par délégation [Ger 95] est aussi très présente dans cette architecture ; un gestionnaire peut déléguer un autre pour effectuer une ou plusieurs opérations de gestion. Les architectures distribuées sont généralement implémentées selon un modèle à 3 ou 4 tiers.

Les architectures distribuées de gestion ont l'avantage d'accroître la disponibilité des gestionnaires dans le réseau ; elles étendent les fonctionnalités des agents en leur permettant d'exécuter ou d'interpréter les scripts qui leur sont envoyés par le gestionnaire [Ade 98]. Dans ces architectures, le coût de communication agents/gestionnaires est réduit par rapport à l'architecture centralisée de gestion.

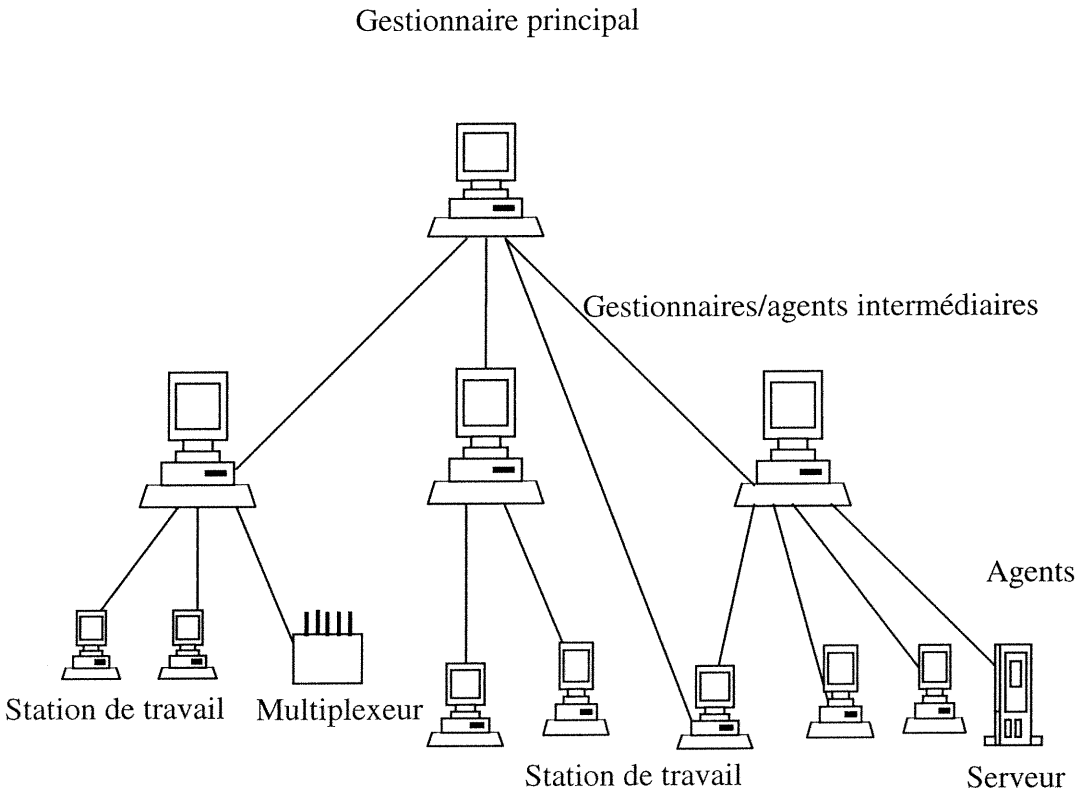
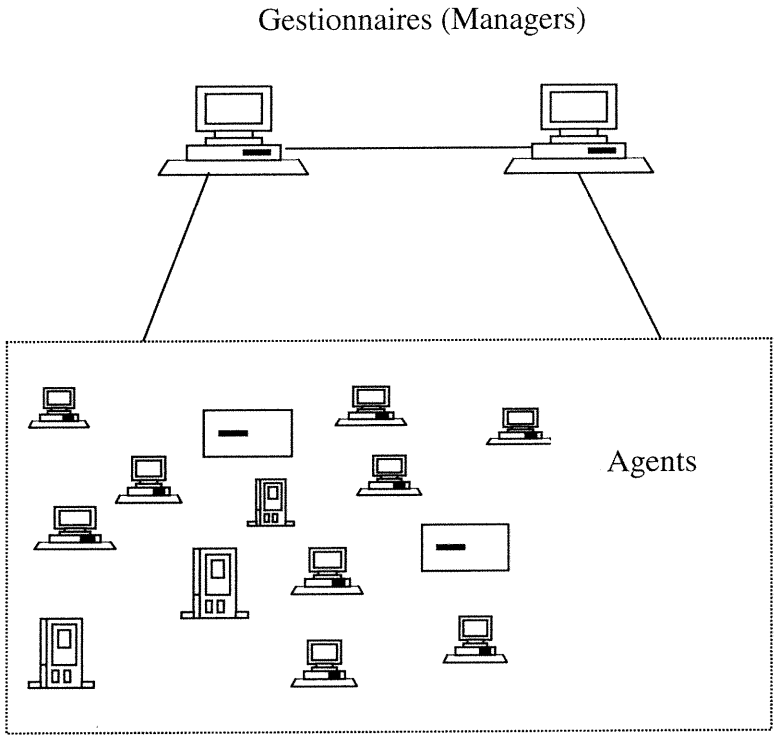


Figure 1.6(a) Gestion hiérarchique



Légende :

Serveur

PC ou Station de travail

Routeur

Figure 1.6(b) Gestion coopérative

1.2.3 Modèle informationnel

Pour gérer un objet, il faut tout d'abord l'identifier de façon unique ; pour ce faire, un modèle informationnel est nécessaire. Le modèle d'information est très important dans

le système de gestion. Il spécifie la description pour la représentation des objets gérés, fournit la possibilité pour les identifier, manipuler, créer des relations entre eux et les traiter en utilisant des protocoles de gestion.

Les informations de gestion d'un même élément ou service sont stockées dans différentes bases de donnée (par exemple, QoS LDAP schema [Str 99], QoS PIB). Ceci est un obstacle d'interopérabilité.

1.2.4 Modèle de communication

Le modèle de communication est la description des processus d'échange d'informations de surveillance et de contrôle du réseau. Il spécifie les mécanismes de transmission des messages de gestion entre les entités, tout en définissant le format de donnée à échanger entre elles.

1.3 Systèmes de gestion standards

Il existe plusieurs systèmes de gestion pour différentes architectures de réseau. Dans cette section, nous présentons quelques-uns d'entre eux.

1.3.1 Gestion de l'interconnexion de systèmes ouverts (OSI)

L'organisation internationale de normalisation (ISO) a développé en collaboration avec l'Union Internationale des Télécommunications (UIT) un système de référence pour la gestion des réseaux dont l'architecture est basée sur la norme ISO. La projection de ce système de référence sur les quatre modèles décrits précédemment (i.e. le modèle organisationnel, le modèle fonctionnel, le modèle d'information et le modèle de communication) est comme suit :

- Le modèle fonctionnel de l'interconnexion des systèmes ouverts comprend

cinq fonctions : la gestion des fautes, la gestion des informations comptables, la gestion de la configuration, la gestion de la performance et la gestion de la sécurité, [Hei 99] [Noe97] [Slo 94] [Sta 93].

- Le modèle organisationnel est composé de deux entités : le gestionnaire et l'agent.

- Le modèle informationnel de l'interconnexion des systèmes ouverts est appelé Structure of Management Information (SMI). Le SMI du système de gestion de l'interconnexion des systèmes ouverts utilise le concept orienté-objet pour définir la base d'information de gestion (MIB). L'utilisation du concept orienté-objet pour la définition des objets gérés permet leur réutilisation et dérivation.

- Le modèle de communication de l'interconnexion des systèmes ouverts est un ensemble d'éléments appelé Common Management Information Service Element (CMISE) [Sta 93]. Il est composé du protocole Common Management Information Protocol (CMIP) et des services Common Management Information Service (CMIS). Le CMIS est utilisé pour la communication des fonctions de gestion entre le gestionnaire et l'agent, tandis qu'à travers le protocole CMIP est effectuée la communication de gestion.

Le système de référence de l'OSI est surtout utilisé dans la gestion des réseaux de télécommunication (TMN) et celle des architectures distribuées de réseaux de télécommunication. Son modèle d'information de gestion fournit des options efficaces pour la modélisation des objets gérés du réseau, mais, sa complexité ne facilite pas son utilisation dans les réseaux simples et de petite taille.

1.3.2 Le protocole SNMP de l'IETF

Simple Network Management Protocol [RFC 1157] est un protocole très simple développé par la communauté d'Internet (IETF) pour effectuer la gestion des réseaux dont l'architecture est basée sur Transmission Control Protocol/Internet Protocol (TCP/IP). Il est une collection de spécifications qui sont : le protocole, les informations de gestion et l'organisation, ainsi que les mécanismes d'échange d'informations entre les différents composants.

Ces trois spécifications correspondent aux modèles organisationnel, informationnel et de communication.

- Le modèle organisationnel est composé de gérant et géré, sur lesquels s'exécutent respectivement le gestionnaire (Manager) et l'agent. Le gestionnaire est un ensemble d'outils de gestion qui permet aux administrateurs du réseau de contrôler le réseau, d'analyser les données de surveillance, de réparer les erreurs et d'enregistrer les informations de gestion, quant à l'agent, il répond aux requêtes du gestionnaire et lui envoie des notifications. L'agent peut s'exécuter sur tous les composants du réseau (tels que : le routeur, le hôte, le pont, le concentrateur).

- Le modèle d'informations de SNMP est écrit dans une structure d'informations de gestion (SMI) [RFC 1155]. Les informations de gestion définies sont stockées dans la base d'information de gestion (MIB). La MIB est une spécification, une collection de spécifications ou les valeurs actuelles de l'information dans un système. Les objets sont organisés dans une MIB sous une forme hiérarchique arborescente, figure 1.7. Ces objets peuvent être des attributs ou tables. Chaque objet a les caractéristiques suivantes: un nom, un

identificateur d'objet (OID), une contrainte optionnelle, une syntaxe ASN.1 (Abstract Syntaxe Notation One), un mode d'accès, un état et une description.

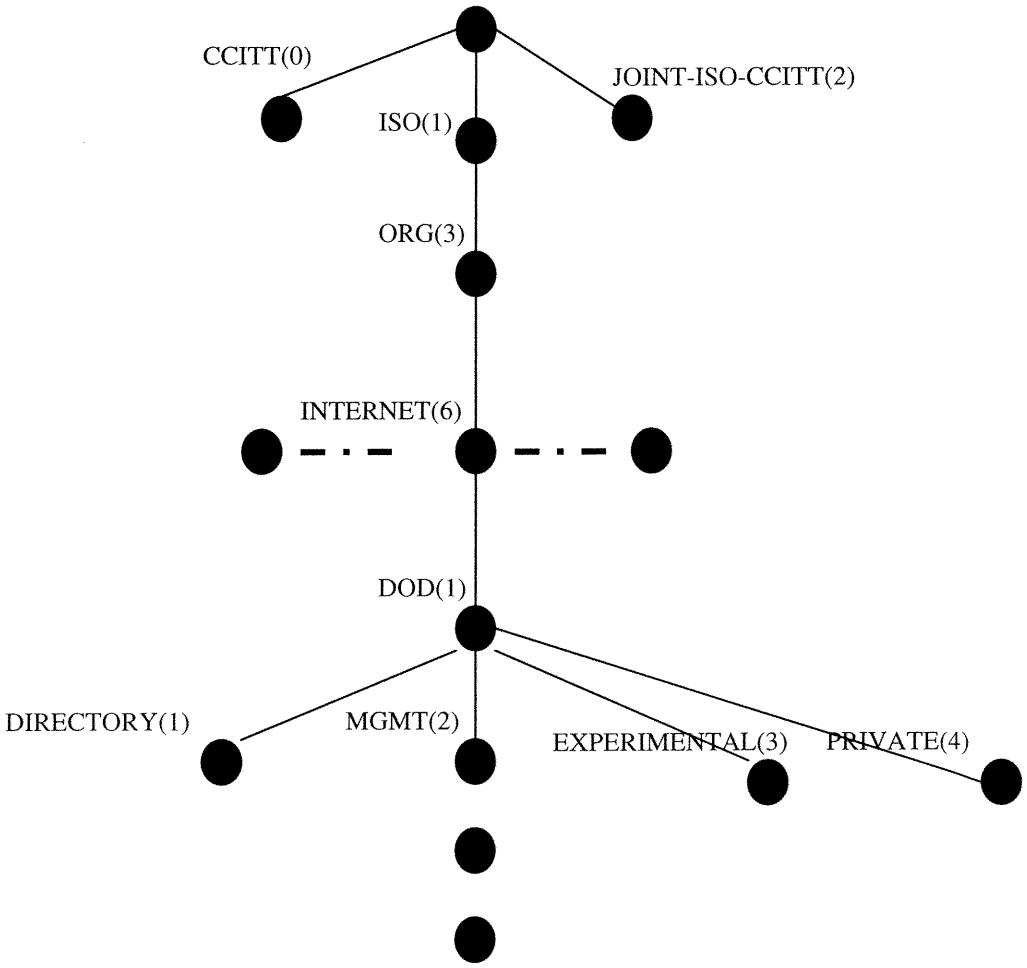


Figure 1.7 Structure d'une MIB

- Le modèle de communication de SNMP est un protocole qui permet aux gestionnaires et agents de se communiquer les messages de gestion (GetRequest, GetNextRequest, SetRequest, GetResponse, et Trap).

Le système de gestion SNMP est le plus déployé dans les réseaux de communication de donnée. Ce large déploiement est du : (1) au développement de l'Internet, ce qui favorise l'utilisation repandue des protocoles basés sur IP ; (2) à la simplicité de SNMP, ce qui permet de l'utiliser à des coûts raisonnables.

Présentement, il y a plusieurs versions du protocole SNMP (SNMPv1 [RFC 1157], SNMPv2 [RFC 1441] et SNMPv3 [RFC 2570]) et plusieurs MIB normalisées (MIB-II [RFC 1213], RMON) [RFC 1757]. Le SNMP fût conçu à l'origine pour fonctionner sur User Datagram Protocol (UDP), mais présentement il existe des implémentations pour TCP [Sch 00].

Les fonctionnalités de SNMP sont limitées (par exemple, son modèle d'information de gestion est moins efficace que celui du système de l'ISO, la sécurité et le volume de donnée à transporter entre les entités ne sont pas identiques dans les différentes versions) et les besoins en administrations ne font que s'accroître. Ces dernières années, nous assistons à l'émergence de plusieurs tendances (par exemple, améliorer SNMP, utiliser un nouveau système de gestion ou combiner les systèmes existants [San 99]).

Parmi les nouvelles solutions, figure le système de gestion de politiques. Il est développé pour résoudre les problèmes de gestion causés par le développement des nouvelles technologies de l'information.

1.3.3 Standards pour les systèmes distribués

Une architecture distribuée est basée sur l'un ou plusieurs des concepts suivants : client/serveur, Remote Procedural Call (RPC) [RFC 1831], et Object Request Broker (ORB). Pour la gestion des architectures distribuées les standards suivants sont utilisés : Distributed Computing Environment (DCE), Open Distributed Processing

(ODP), Telecommunication Information Network Architecture (TINA) et Common Object Request Broker Architecture (CORBA) [CORBA].

CORBA est un standard en matière d'interopérabilité dont la promotion est faite par plus de huit cents organisations et entreprises. Ces organisations et entreprises sont regroupés sous le nom de Object Management Group (OMG) [CORBA]. Le concurrent éminent de CORBA sur le marché est DCOM (Distributed Component Object Model) [Cha 97] de Microsoft, mais, le fait que l'OMG ne fournisse que des spécifications (pas de code) assure son avancement de CORBA vers une position de standard. C'est pourquoi nous nous limiterons ici à décrire CORBA.

1.3.3.1 Common Object Request Broker Architecture (CORBA)

CORBA [Cha 97][Ste 96] est le composant principal de Object Management Architecture (OMA). Elle a été spécifiée par Object Management Group (OMG) et est la norme de fait pour le développement d'applications distribuées, dont notamment les applications de gestion intégrée.

Il a été implémenté pour aider les constructeurs de systèmes distribués à résoudre les problèmes d'interopérabilité, ainsi, il permet aux différentes applications de communiquer dans un environnement hétérogène.

Plus précisément, la norme CORBA combine les technologies objet et client/serveur, aide à la construction d'applications orientées-objet distribuées. Dans son architecture sont définis les quatre modèles de référence, à savoir : le modèle informationnel, le modèle fonctionnel, le modèle organisationnel et le modèle de communication.

- CORBAdomains constitue le modèle organisationnel de CORBA. Il est un ensemble de services spécifiques à des domaines particuliers d'application. Les entités ou composants de ce modèle sont des objets (unités logiques) qui

peuvent assurer le rôle de client et/ou de serveur.

- Le modèle fonctionnel de CORBA est composé de CORBAfacilities, et CORBAservices. Le CORBAfacilities est un ensemble de services communs aux applications. Ces services sont des fonctions de haut niveau (tels que, les interfaces usagers, les documents composites, l'administration d'informations, l'administration de système, et l'administration des tâches). Le CORBAservices fournit des services basiques et communs aux objets (tels que, le nomage, la notification des événements, la sécurité, la messagerie, le stockage et l'archivage, les relations entre objets, la transaction et la concurrence).

- Le modèle d'informations de CORBA est le langage orienté-objet appelé Interface Definition Language (IDL). IDL cache l'implémentation de l'objet et ne définit que les propriétés fondamentales communes des objets. Quand les applications et les données ont une interface IDL, la communication entre elles est indépendante de leur emplacement physique, de la plate-forme qu'elles utilisent, et du protocole de communication.

- Le modèle de communication de CORBA est l'ORB, qui permet le routage des requêtes entre le client et le serveur. Le client invoque le nom de l'objet, et grâce à l'ORB, il accède à l'implémentation de l'objet pour effectuer les opérations. Cette communication entre objets est indépendante des langages, systèmes d'exploitation et réseaux qu'utilisent ces objets. Ceci permet une flexibilité et une distribution de l'application.

Dans CORBA, la principale communication entre les processus (inter-process) est prise en charge par le General Inter Object Request Broker Protocol (GIOP). GIOP permet l'interopérabilité entre les différentes implémentations de l'ORB. Il définit une syntaxe de transfert connue sous le nom de Common

Data Representation (CDR), et sept types de message de base. La transcription de GIOP à TCP/IP est référée par Internet Inter-ORB Protocol (IIOP). Le mécanisme par lequel les objets sont accédés à travers IIOP et entre les ORBs de différents vendeurs est Interoperable Object Reference (IOR).

CORBA est une architecture fortement liée aux concepts orientés-objet, qui sont: l'objet, la classe, l'encapsulation, l'héritage et le polymorphisme. En se basant sur ces concepts, le programmeur peut créer des entités logicielles composites, réutilisables et extensibles. Dans le modèle orienté objet de CORBA, il y a l'objet, la requête, la création et destruction d'objet, le type, l'interface et les opérations.

Le deuxième concept sur lequel est basé CORBA est le modèle client/serveur. CORBA ajoute une dimension à l'interaction client/serveur en insérant un broker (ORB) entre le client et le serveur, ce qui réduit la complexité de l'implémentation.

Dans le modèle client/serveur de CORBA, une requête peut être statique ou dynamique. Une requête statique est toute requête définie pendant la compilation, elle est effectuée à travers un stub IDL au skeleton (Static Invocation Interface ou SII) sur le serveur. Une requête dynamique est toute requête ayant la possibilité d'ajouter des objets et interfaces nouveaux sans changement dans le code du client. La requête dynamique utilise l'interface dynamique d'invocation (Dynamic Invocation Interface ou DII).

L'introduction de CORBA dans la gestion intégrée permet à ses utilisateurs de se servir d'une même technologie pour différents besoins (par exemple, la communication normale et celle de gestion). CORBA facilitera la transformation de plusieurs interfaces propriétaires en interfaces indépendantes de constructeurs. Il facilitera aussi l'utilisation des browsers internets pour la gestion et la coordinnation des agents dans une machine,

1.3.4 Système de gestion basé sur le WEB

Les demandes d'extensibilité et de haute performance des réseaux basés sur la technologie d'Internet, changent progressivement les architectures client/serveur en architectures multi-tierces. Les architectures multi-tierces sont des architectures composées de trois ensembles, qui sont : l'ensemble des clients, l'ensemble des stockages de donnée et l'ensemble des serveurs.

Les clients sont des applications Java, Java applets, scripts ou clients réseaux, qui permettent aux administrateurs d'accéder aux données du réseau à distance au travers de n'importe quel navigateur Web standard [Jam 97] [Ade 98]. Ceci facilite la collecte des données, rend flexible leur compte-rendu.

Les stockages de donnée sont des bases de donnée relationnelles, non relationnelles ou des services de répertoire, qui permettent le stockage des données du système.

Les serveurs permettent la distribution du traitement des requêtes des clients. L'architecture multi-tierce est organisée en interposant les serveurs entre les clients et les stockages de donnée.

Ces architectures ont non seulement l'avantage de faciliter l'extension du réseau selon les besoins mais aussi, elles augmentent la disponibilité des serveurs dans le réseau et l'accès aux applications de gestion à distance. Plusieurs outils permettent le développement d'applications de gestion basés sur le Web (tels que, Java, Java Management API [JAVA-MAPI], WBEM [WBM]).

1.3.5 Technologie d'agent mobile

Dans le modèle traditionnel client/serveur, la communication s'effectue par RPC (Remote Procedure Calling) [RFC 1831]. Le client invoque un service au serveur en

lui envoyant un message; le serveur traite la requête du client et lui renvoie la réponse. Dans chaque communication client/serveur, il y a au minimum deux messages transportés sur la ligne de communication.

Avec les agents mobiles, la communication s'effectue par la programmation à distance (Remote Programming) [Coc 97]. La programmation à distance a l'avantage de fournir le programme et les arguments devant être exécutés dans une machine à distance. Ainsi, une fois le programme transporté dans la machine de destination, elle s'exécute localement.

La mobilité et l'intelligence des agents peuvent permettre aux administrateurs de systèmes de rapidement et efficacement surveiller et contrôler leur système à distance. Avec les agents mobiles, la résolution des erreurs, la configuration et l'optimisation du réseau n'attendront plus l'intervention des administrateurs que très exceptionnellement.

L'utilisation des agents mobiles apporte les avantages suivants [Coc 97] :

- La performance : l'appel de procédure à distance demande plus de ressource que la programmation à distance, le fait que l'agent mobile se déplace dans la machine à distance pour s'exécuter limite l'utilisation du CPU.

- La Connexion par intermittence : un utilisateur n'a pas besoin de garder sa machine connectée pour attendre le retour d'un agent mobile.

- L'économie de stockage : la consommation des ressources est limitée. Un agent mobile réside dans un seul nœud à la fois, et il n'est pas nécessaire de dupliquer les fonctionnalités sur chaque nœud, car, l'agent apporte sa fonctionnalité et peut aller chercher l'information désirée.

- La réduction du trafic : l'agent se déplace avec ses informations, ce qui lui permet de s'exécuter ou continuer son exécution après une interruption.

- L'interaction autonome asynchrone : l'agent mobile peut être délégué pour exécuter certaines tâches, même s'il n'est pas actif.

- L'utilisation moyenne de la bande passante: l'information transférée est minime.

- La possibilité d'interaction avec les systèmes à temps réel : installer un agent mobile proche des systèmes à temps réel prévient les délais causés par la congestion du réseau. Dans le système de gestion de réseau l'agent de gestion de réseau réside habituellement près du matériel.

- La robustesse et la tolérance aux erreurs : si le système distribué a un mauvais fonctionnement, l'agent mobile peut être utilisé pour augmenter la disponibilité de certains services dans le domaine concerné.

- L'hétérogénéité : les agents mobiles sont séparés des serveurs par la mobilité du Framework. Si le cadre mobile est en place, les agents peuvent cibler n'importe quel système.

- L'extensibilité des services en ligne : les agents peuvent être utilisés pour dynamiquement étendre la capacité des applications.

- La commodité dans le développement : la création d'un système distribué basé sur les agents mobiles peut être relativement facile après la mise en place du cadre mobile.

1.4 Conclusion

Le développement des systèmes de réseaux suit de près celui des nouvelles technologies de l'information. Autrement dit, nous assistons présentement à l'explosion continue des services et à l'insertion ininterrompue de nouveaux composants dans les réseaux. L'intégration de la gestion du réseau et des services devient non seulement une nécessité, mais aussi, une priorité.

Deux fonctions fondamentales de cette gestion intégrée sont:

- La gestion efficace de la configuration des paramètres de contrôle des éléments gérés et services du réseau dont les aspects critiques sont le stockage et la distribution des données.

- Le contrôle automatique de la reconfiguration des éléments et services du réseau, ce qui nécessite l'utilisation de mécanismes adaptés en temps réel.

Bien que largement déployé dans la gestion des réseaux de l'Internet, SNMP ne fournit pas d'option pour la réutilisation des informations de gestion, l'unification de la gestion des composants gérés et des services, et l'automatisation de la configuration. CORBA facilite l'interaction entre les différentes applications de gestion mais, manque de modèle d'information de pour décrire les paramètres de contrôle des composants et services. Le système de gestion normalisé par l'ISO complexe ce qui rend son implémentation difficile et coûteuse.

Ces dernières années ont connu l'émergence d'outils plus sophistiqués et mieux adaptés à la gestion de la configuration et au contrôle automatique de la reconfiguration des éléments gérés et services des réseaux de l'Internet. Ces outils, dont en particulier les systèmes à base de politiques et les services répertoires, font

aujourd'hui l'objet de normalisation au sein d'organismes internationaux comme l'IETF. Ils permettent de faciliter les tâches de configuration et, surtout, ils offrent un plus haut degré d'automatisation. Nous réservons le chapitre suivant à la description de ces outils et techniques émergents pour la gestion de réseau.

Chapitre 2

Techniques émergentes pour la gestion basée sur les politiques

2.1 Introduction

L'intégration de la voix et de la vidéo dans les données à transporter et le besoin différencié et prévisible des utilisateurs à acheminer ces données ont mis en cause le service classique de transport "best-effort" jusqu'ici utilisé sur IP. Ceci a nécessité la fourniture d'autres services de transport (tels que, les services différenciés et intégrés [RFC 1633]).

Garantir la qualité de ces services de transport requiert un système de gestion intégré sophistiqué et mieux adapté pour dynamiquement régulariser les différents besoins des utilisateurs dans le réseau. Les systèmes standards de gestion (par exemple, SNMP) manquent les concepts facilitant l'unification de la gestion des éléments et services et la régularisation dynamique des nouveaux besoins. Dans ce chapitre, nous décrivons la technique émergente pour cette gestion automatique de réseau et des services. Plus particulièrement, nous détaillons le système de gestion basé sur les politiques, ses différentes entités et les principes sous-jacents à ces entités.

2.2 Architecture du système de gestion de politiques

Le système de gestion de politiques (figure 2.1) est un ensemble de modèles de protocole, d'informations et de services que les administrateurs peuvent utiliser pour

contrôler tous les composants et services du réseau. Il permet la configuration en ligne des multiples services de transport grâce à des unités de contrôle de politiques.

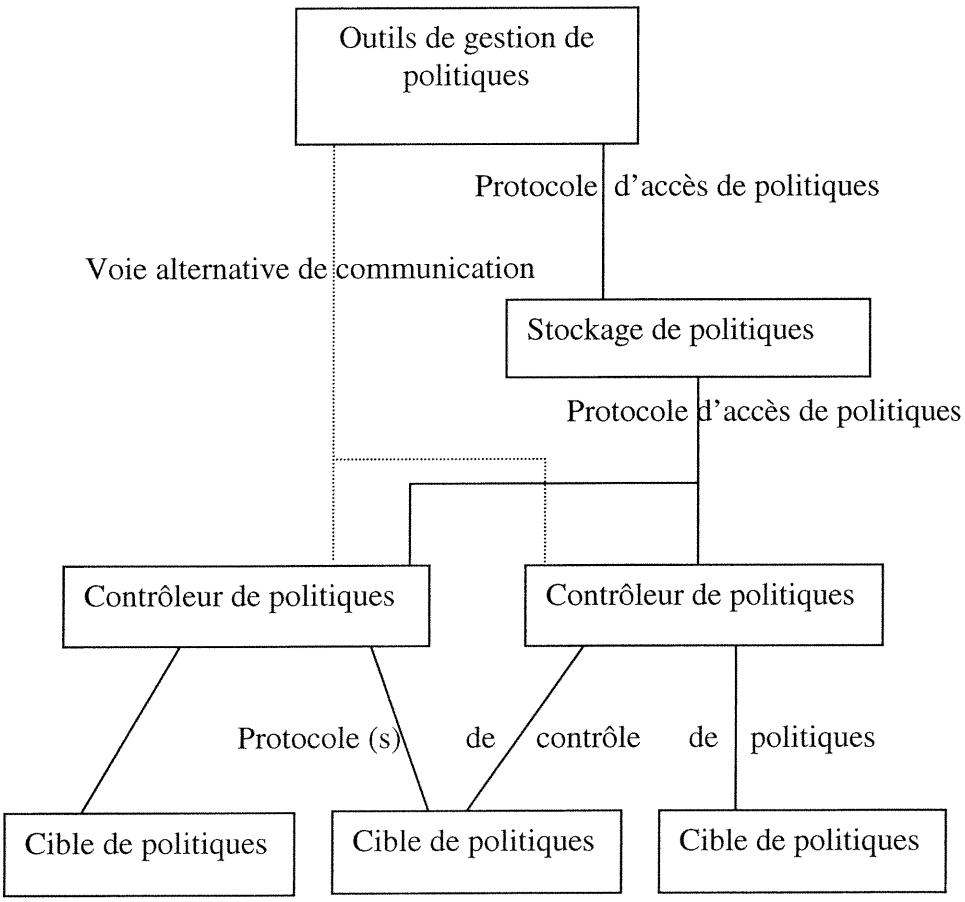


Figure 2.1 Système de gestion de politiques

L'objectif du système de gestion de politiques est de gérer et de maintenir un réseau et ses services, de manière à rendre les opérations du réseau conformes aux objectifs de l'organisation qui l'exploite [Ste 99].

Dans le système de gestion de politiques, sont définis les modèles suivants: le modèle d'information de gestion [Moo 00], le modèle organisationnel et les fonctionnalités de

chaque unité organisationnelle [Hug 99] [Ste 99].

2.3 Organisation du système de gestion de politiques

Dans l'architecture du système de gestion de politiques, il y a les composants suivants: l'application de gestion de politique, le dépositaire ou stockage de politiques, le contrôleur de politique et la cible de politique [Ste 99].

- Outil de gestion de politiques : un outil ou ensemble d'outils de gestion de politiques ayant une ou plusieurs fonctionnalités. Les fonctionnalités d'un outil de gestion sont: d'une part l'édition, le stockage, la recherche, la modification de politiques et, de l'autre, l'association d'une politique à un composant du réseau et la notification de disponibilité ou de modification d'une politique.

- Le stockage de politiques est un fichier, une base de donnée ou un service de répertoire permettant le stockage et la recherche des politiques.

- La cible de politique est un élément géré qui évalue les conditions de l'application des politiques et exécute les actions de politiques.

- Le contrôleur de politiques est une entité qui fait l'analyse des politiques, décide quelles politiques doivent être implantées et initie leur déploiement dans les cibles de politiques.

Le contrôleur de politiques opère selon deux modes : il décide de l'application de la politique à la cible à chaque fois que l'événement nécessitant l'implantation de cette politique se produit ou bien il configure la cible de politiques pour que celle-ci exécute la politique à chaque fois que cela est nécessaire. Dans une architecture simple de système de gestion de politiques, le contrôleur et la cible de politiques peuvent être dans un même composant physique.

Les différentes entités (outil de gestion, stockage, contrôleur et cible de politiques) du modèle organisationnel du système de gestion de politiques peuvent être structurées pour obtenir trois différentes abstractions [Ste 99]. Ces trois abstractions sont : l'abstraction administrative de politiques, les règles de politiques, et les mécanismes de contrôle de politiques, figure 2.2.

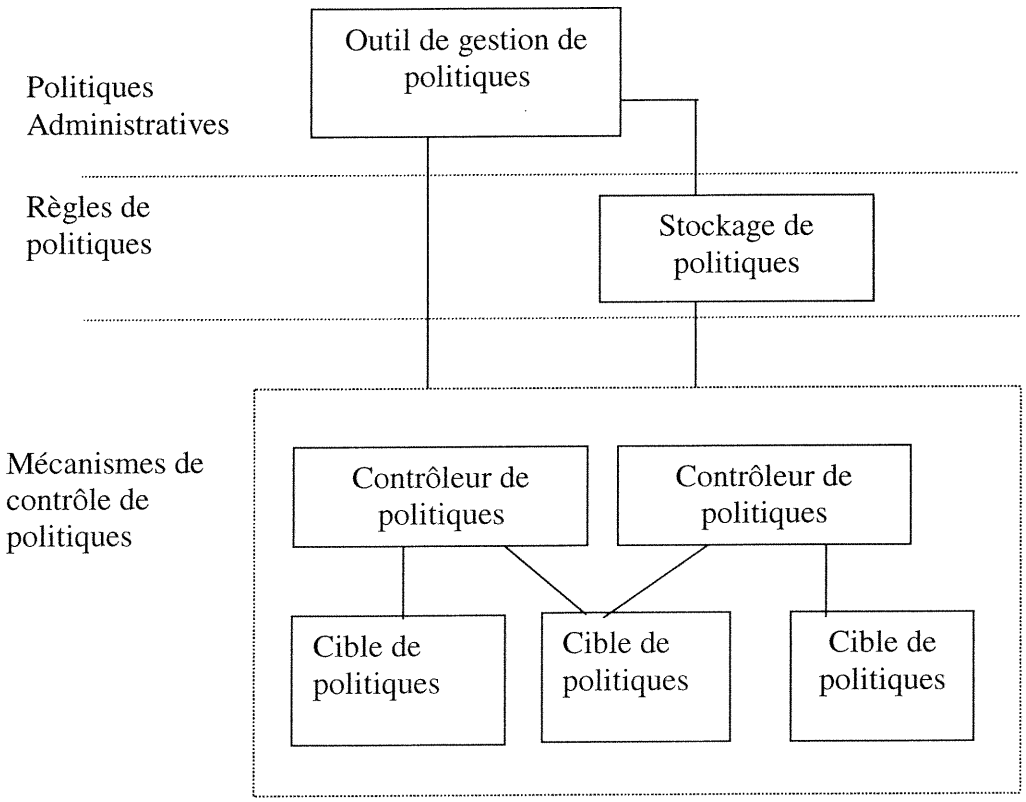


Figure 2.2 Organisation d'un système de politiques

2.3.1 Politiques Administratives

Ce niveau d'abstraction est le niveau où s'exécute l'outil de gestion de politiques. Les administrateurs de réseau interviennent directement à ce niveau pour définir, éditer, stocker, rechercher, et modifier les politiques de gestion de toutes les ressources et

tous les services du réseau. Certains outils de gestion peuvent aussi vérifier la consistance des politiques avant leur stockage.

Les politiques définies par les administrateurs à ce niveau doivent être non seulement compréhensives et applicables par les contrôleurs de politiques, mais aussi, elles doivent indirectement l'être par ses utilisateurs finaux qui sont les cibles de politiques.

2.3.2 Règles de politiques

La règle de politiques est un élément fondamental de politiques. Elle est un ensemble d'actions ou une séquence d'actions devant être initialisées quand un ensemble spécifié de conditions sont satisfaites. Une règle de politiques peut être une expression d'utilisation d'un service se trouvant dans le réseau (connexion, transmission, sécurité, etc.), ou une expression de configuration d'une cible (bande passante entre une source A et une destination B, adresse de serveurs, etc.). Elle est exprimée sous la forme [Moo 00] [Ste 99]:

IF <ensemble de conditions à satisfaire> THEN <ensemble ou séquence d'actions à entreprendre>.

Dans l'expression if <conditions> then <actions>, la partie <conditions> peut être une expression composée et dépendre de l'état des composants, tels que les hôtes, les applications, les protocoles, les utilisateurs ou n'importe quel autre sous-composant du système. La partie <actions> spécifie les services autorisés ou refusés ou d'autres paramètres à utiliser dans l'approvisionnement d'un ou plusieurs services.

Exemple:

if sourceIPAddress == 172.3.128.0/15, && DSCP (Differentiated Service Code Point) == 101110 THEN mark voice traffic with Expedited Forwarding [Sap 00].

L'outil de gestion de politiques édite les règles de politiques qui doivent être utilisées par les contrôleurs de politiques pour influencer le comportement des cibles de politiques. Les règles de politiques spécifient la logique utilisée pour accomplir les services prescrits dans les accords et objectifs de service (Service Level Agreements et Service Level Objectives).

2.3.3 Mécanismes d'implantation de politiques

L'implantation de politiques est assurée à travers des méthodes, protocoles, et outils sous-jacents permettant l'implémentation (évaluation et exécution) des règles de politiques. Dans la plupart des cas, le contrôleur de politiques traduit les règles de politiques pour générer les instructions appropriées pour la cible de politiques.

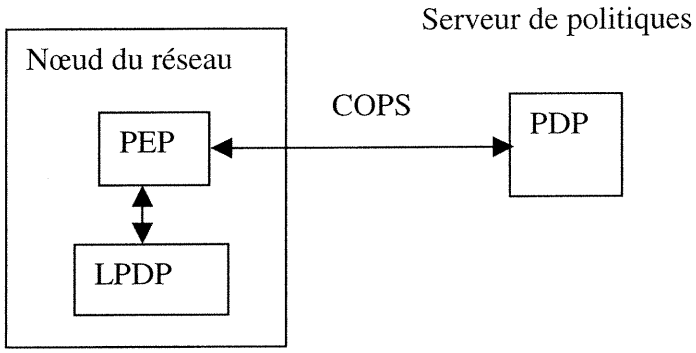
Dans certains cas spécifiques, il identifie tout simplement les règles de politiques appropriées pour un flux ou un environnement donné, puis les transmet aux cibles correspondantes. Dans l'un ou l'autre des cas, ces politiques sont évaluées et implantées par les cibles de politiques.

Les mécanismes d'implantation de politiques peuvent être des protocoles de gestion déjà déployés dans le réseau, (par exemple, SNMP), des protocoles spécialement développés pour l'implantation de politiques dans le réseau (par exemple, COPS [RFC 2748], COPS-RVSP [RFC 2749] et COPS-PR [Fra 99]).

2.3.3.1 Common Open Policy Service (COPS)

Common Open Policy Service (COPS) [2748] est un protocole normalisé de contrôle de politiques. Dans l'architecture de COPS (figure 2.3), le contrôleur de politiques, qui décide l'application de politiques dans les composants du réseau (cibles) est le serveur de politiques, il est appelé "Policy Decision Point" (PDP). La cible de politiques est le

client de politiques qui implante la décision du PDP, elle est appelée "Policy Enforcement Point" (PEP). La communication entre PDP et PEP est assurée par un protocole, ce protocole est appelé "Common Open Policy Service Protocol" (COPS). Il permet à PDP d'envoyer les messages de contrôle de politiques à PEP et de recevoir les messages de notifications d'implantation de politiques de PEP.



Légende

COPS: Common Open Policy Service

PEP: Policy Enforcement Point

LPDP: Local Policy Decision Point

PDP: Policy Decision Point

Figure 2.3 Architecture de COPS

COPS assure l'implantation des politiques dans le réseau grâce à la coopération entre ces trois entités (le PDP, le PEP et le protocole COPS) et un stockage externe de politiques. Lors de la décision d'implantation de politiques, PDP recherche la politique à implanter dans le stockage externe (voir figure 2.4).

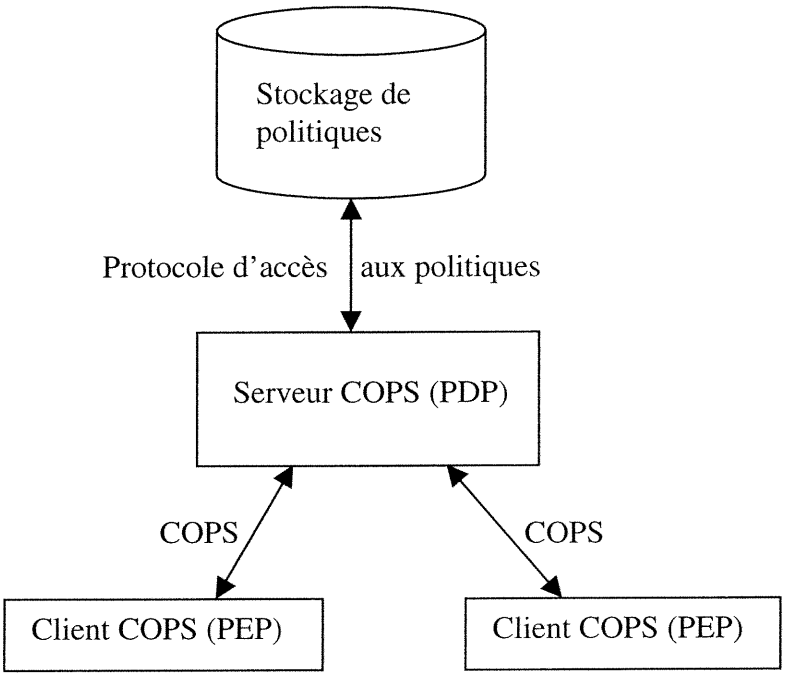


Figure 2.4 Interaction PDP/PEP et PDP/stockage de politiques

COPS est normalisé et ouvert. Il existe des extensions de COPS pour des utilisations spécifiques, tels que, COPS usage for RSVP [RFC 2749] et COPS Provisioning (COPS-PR) [Fra 99]).

2.4 Modèle d'informations de gestion (Policy CIM)

Dans le système de gestion de politiques, le modèle informationnel est basé sur le concept orienté-objet [Moo 00]. Il est une extension de Common Information Model

(CIM) normalisé par Distributed Management Task Force (DMTF) pour la gestion des différents éléments physiques et logiques du réseau (Directory-Enabled Network) [DMTF].

Le modèle d'information du système de gestion de politiques décrit les caractéristiques structurelles des objets gérés (par exemple, les utilisateurs, les dispositifs du réseau, et les services) et les relations existantes entre eux dans deux types de classes hiérarchiques, qui sont: la classe structurelle et la classe relationnelle.

- La classe structurelle représente les règles et contrôles de politiques;
- La classe relationnelle indique comment les classes structurelles sont reliées entre elles.

La règle de politiques est au centre de ces deux classes, qui peuvent correspondrent à plusieurs classes si les politiques sont représentées dans les données d'un autre stockage. Par exemple, Lightweight Directory Access Protocol (LDAP) [Str 99], Simple Network Management Protocole Management Information Base (SNMP-MIB) ou Policy Information Base (PIB) [McC 00].

2.4.1 Policy Information Base (PIB)

La base d'information de politiques (Policy Information Base) [McC 00] est une base de donnée utilisée par le protocole d'approvisionnement de politiques (COPS-PR) [Fra 99], pour configurer les éléments gérés du réseau. La PIB est définie en se basant sur la structure d'information d'approvisionnement de politiques (Structure of Policy Provisioning Information) [Smi 00]. La structure d'information d'approvisionnement de politiques (SPPI) utilise un sous-ensemble adapté de la structure d'information de gestion de SNMP (SNMP-SMI) [RFC 1155].

Une PIB peut être considérée comme une arborescence de donnée, dont les branches représentent des structures de donnée ou classes d'approvisionnement de politiques (PRC), alors que les feuilles représentent diverses instantiations d'approvisionnement de politiques (PRI). La figure 2.5 illustre la structure d'une base d'information de politiques.

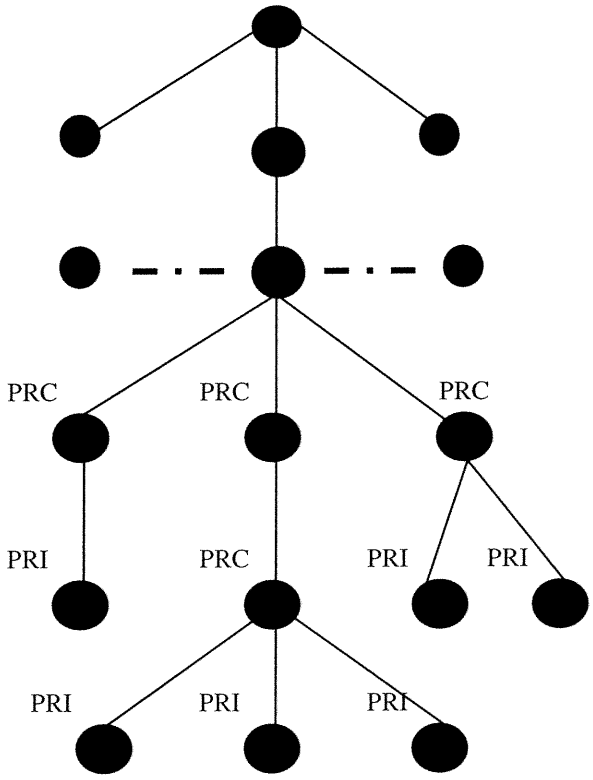


Figure 2.5 Structure d'une PIB

2.4.2 Common Information Model (CIM)

Le modèle commun d'information (CIM) est un modèle de donnée basé sur les concepts orienté-objet, qui sont indépendants de tout langage d'implémentation. Le

CIM permet de décrire l'information de gestion globale d'un réseau d'entreprise pour une gestion intégrée. Le modèle commun d'information est composé d'un cahier de charges et d'un schéma.

- Le schéma fournit les descriptions du modèle.
- Le cahier de charges définit les détails pour l'intégration avec d'autres modèles de donnée (i.e., la base d'information de gestion de SNMP ou le format d'information de gestion de DMTF).

Dans le schéma sont définis les éléments suivants [Hei 99]:

- La classe qui caractérise un ensemble d'objets partageant les mêmes propriétés.
- La méthode qui définit une opération exécutable sur les objets d'une classe.
- Le qualificatif qui indique une caractéristique supplémentaire d'une classe.
- L'association (qui est une classe) pour représenter une relation entre deux ou plusieurs objets.
- La référence (qui est une propriété spéciale) pour faire une référence sur d'autres objets.
- Les classes d'événements pour définir les différents types d 'événements de notification.
- Le schéma pour regrouper des groupes d'éléments pour des buts administratifs.

2.4.3 Service d'Annuaire LDAP

Lightweight Directory Access Protocol (LDAP) [RFC 1777] est un protocole normalisé d'accès aux annuaires fonctionnant sur TCP/IP. Il est l'adaptation du protocole DAP (Directory Access Protocol) de l'annuaire X500 de l'interconnexion des systèmes ouverts (OSI). Un annuaire électronique permet aux utilisateurs d'Internet et aux applications de réseau de retrouver un ou plusieurs attributs d'un objet grâce à des fonctions de recherche multicritères. Il permet de partager des bases d'informations sur le réseau interne ou externe.

Les annuaires électroniques peuvent contenir différents types d'information (par exemple, les coordonnées de personnes ou des données de configurations de systèmes). Leur fonction principale est de servir d'entrepôt pour centraliser des informations et les rendre disponibles, via le réseau, à des applications, des systèmes d'exploitation ou des utilisateurs.

Le protocole d'accès LDAP est basé sur le modèle d'interaction client/serveur (figure 2.6). Il a évolué de passerelle d'accès à des annuaires X500, en serveur LDAP autonome avec sa propre base de donnée. Actuellement, il est de plus en plus utilisé pour l'accès aux données stockées dans le répertoire, dans le cadre de nombreuses applications distribuées y compris la gestion de réseaux.

Dans l'architecture de base de LDAP, il existe les éléments suivants: le protocole LDAP, un modèle d'information, un modèle de nomage, un modèle fonctionnel et un modèle de sécurité.

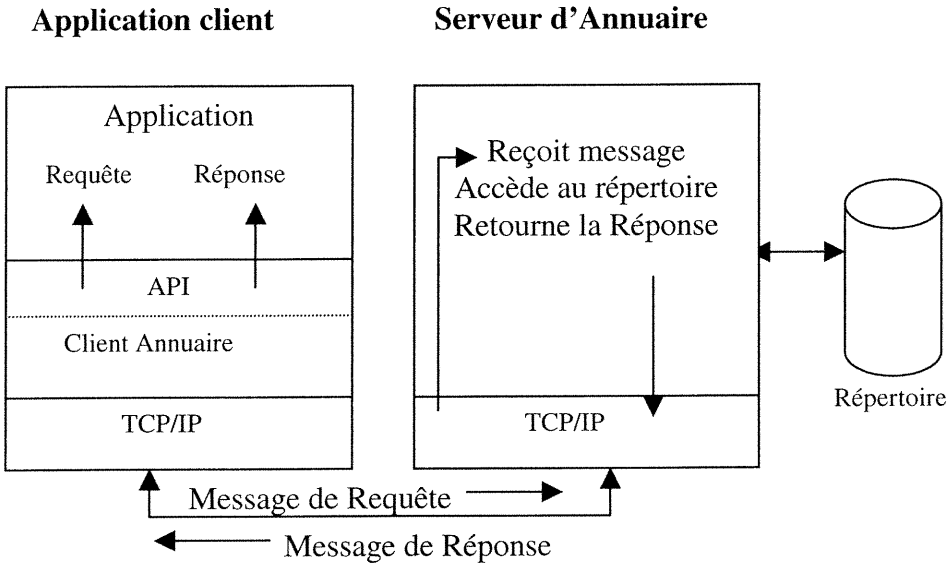


Figure 2.6 Interaction Client/Serveur Annuaire

2.4.3.1 Protocole LDAP

Le protocole LDAP définit la structure des messages échangés entre un client et un serveur LDAP. Dans cette structure sont spécifiées les opérations pour se connecter (ou se déconnecter) au serveur LDAP ; pour rechercher, comparer, créer, modifier ou effacer des objets dans le serveur.

Des interfaces sont disponibles aux programmeurs pour développer des applications comprenant des interactions avec des serveurs LDAP. Ces interfaces sont appelées LDAP APIs [Chr 99] (LDAP Application Programming Interfaces).

2.4.3.2 Modèle d'information de LDAP

L'unité de base des informations stockées dans un annuaire est l'entrée. Les entrées représentent des objets, qui peuvent être abstraits ou issus du monde réel, tels que

personnes, serveurs, organisations, etc. Les entrées sont composées d'une collection d'attributs qui contiennent les informations de l'objet (exemple d'entrée : dc=qosPolicy, dc=policy; dans cette entrée qosPolicy et policy sont des attributs).

Chaque attribut d'une entrée a les caractéristiques suivantes [RFC 1778]:

- Un nom.
- Un identificateur d'objet (OID).
- Une contrainte spécifiant qu'il a une ou plusieurs valeurs.
- Une syntaxe qui spécifie le genre de valeur stockée dans l'attribut et les règles de comparaison qui spécifient les manières de comparaison des valeurs pendant une recherche.
- Un mode d'accès spécifiant qui peut l'accéder et comment l'accéder.
- Une contrainte spécifiant la limite du nombre de valeurs ou la grandeur de la taille d'une valeur.

Le schéma LDAP définit les objets qui peuvent être stockés dans un annuaire, tout en spécifiant leur héritage, dérivation et position dans l'arborescence des types d'objets. Il liste aussi les attributs de chaque type d'objet et spécifie si ces attributs sont optionnels ou obligatoires.

Pour définir un objet dans le schéma, la classe d'objet (*objectclass)* est utilisée. Dans le modèle d'information de LDAP, il y a trois types de classes, qui sont : la classe structurelle, la classe auxiliaire et la classe abstraite.

- La classe structurelle décrit les caractéristiques de base de l'objet.

- La classe abstraite désigne les objets de base de LDAP.

- La classe auxiliaire permet de rajouter des informations complémentaires à des classes structurelles.

2.4.3.3 Le modèle de nomage

Les données LDAP sont structurées dans une arborescence hiérarchique appelé Directory Information Tree (DIT), figure 2.7. La racine de la DIT est le suffixe et ses nœuds sont des entrées de l'annuaire (Directory Service Entries). Les entrées sont arrangées dans le DIT sur la base de leur distinguished name (DN) [RFC 1779]. Un DN est un nom qui identifie l'entrée de façon unique. Les distinguished names sont composés d'une séquence de relative distinguished names (RDNs). Chaque RDN d'un DN correspond à une branche dans le DIT allant de la racine à l'entrée de l'annuaire.

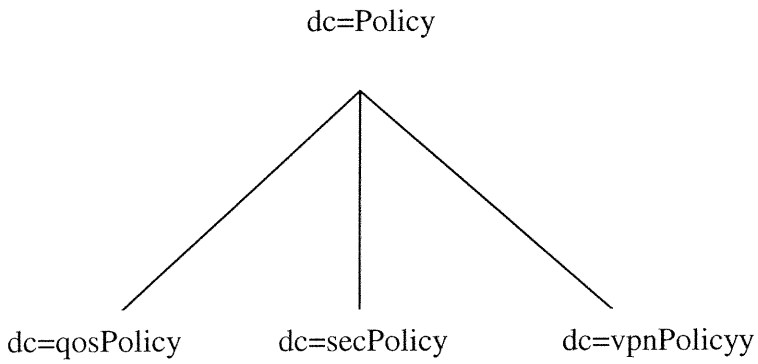


Figure 2.7 DIT de politiques

Chaque RDN est dérivé des attributs de l'entrée. Dans le cas simple et commun, un RDN a la forme <nom attribut>=<valeur>.

Un serveur individuel LDAP peut ne pas stocker toute l'arborescence de l'information

de l'annuaire ; il peut stocker le suffixe du DIT, et un autre serveur stocke les sous-arbres. Ces deux serveurs doivent être reliés pour former un annuaire distribué contenant toute l'arborescence, cette liaison s'accomplissant avec les referrals (figure 2.8).

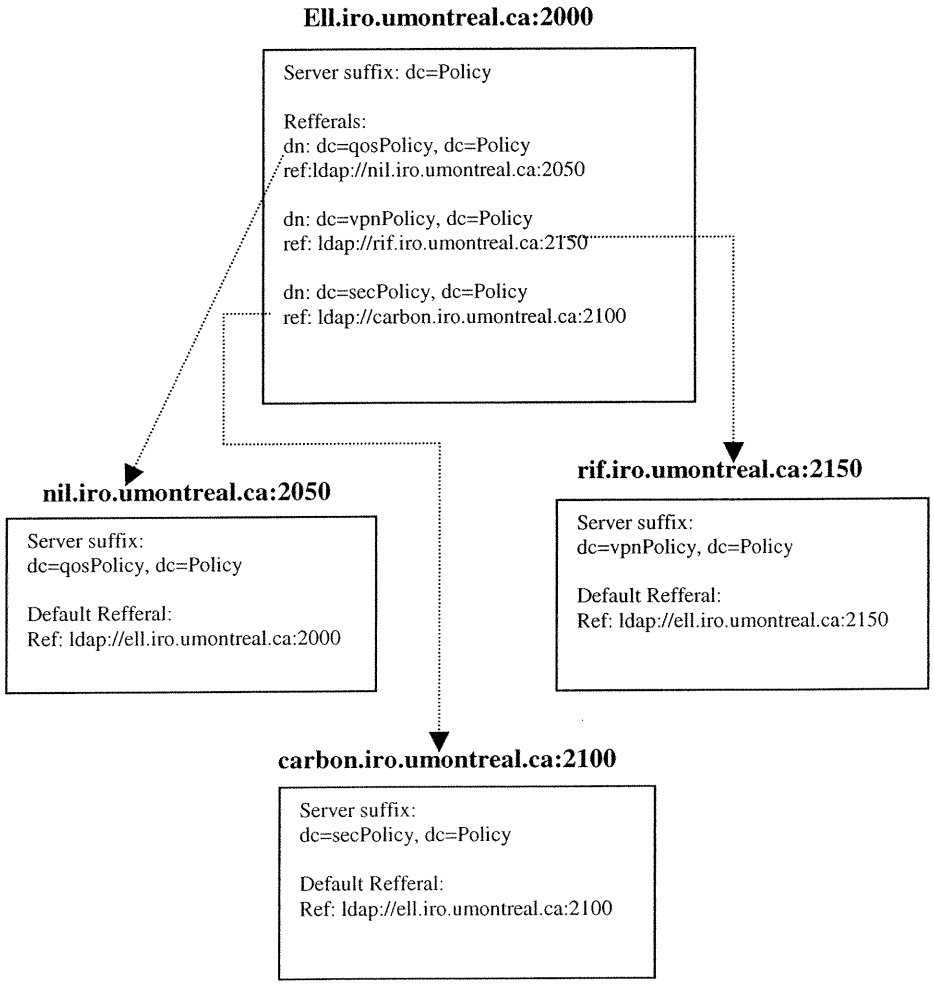


Figure 2.8 Répartition des Politiques de gestion

Le referral est une entrée de *objectclass referral*. Il a un attribut, *ref*, dont la valeur est l'URL LDAP de l'entrée référencée dans un autre serveur LDAP.

2.4.3.4 Le modèle fonctionnel de LDAP

Le modèle fonctionnel définit les opérations qu'un client LDAP peut demander à un serveur. Ces opérations sont les suivantes:

- Search: pour rechercher dans l'annuaire à partir des critères.
- Compare: pour comparer le contenu de deux objets.
- Add: pour ajouter une entrée.
- Modify: pour modifier le contenu d'une entrée.
- Delete: pour supprimer un objet.
- Rename: pour modifier le DN d'une entrée.
- Bind: pour connecter un client au serveur.
- Unbind: pour déconnecter le client du serveur.
- Abandon: pour abandonner une opération en cours.
- Autres opérations propres à la dernière version LDAP v3 [RFC 2251].

La recherche s'effectue avec des paramètres qui spécifient l'endroit de recherche, la profondeur de recherche, les liens à suivre si nécessaire, le nombre de réponses souhaitées, le temps maximal pour la recherche, la liste des attributs à retourner, et le filtre de recherche.

2.4.3.5 Modèle de sécurité

LDAP est un annuaire électronique utilisé pour rendre les informations accéssibles à travers le réseau ou l'Internet. Authentifier qui a émis la requête de l'information et qui l'envoie est une nécessité pour éviter les modifications malveillantes de cette information. Le modèle de sécurité du service de répertoire LDAP est basé sur l'opération de connexion (Bind). Cette opération utilise plusieurs mécanismes de sécurité, parmi lesquels il y a:

- Mot de passe fixe spécifié en texte clair (clair-text password). Quand le client fait la requête d'accès, il utilise un DN en texte clair pour s'identifier, si aucun texte n'est fourni une session anonyme lui est accordée par le serveur. Le mécanisme de connexion par texte clair n'est pas très sécuritaire.

- Utilisation de kerberos dans LDAP v2, ou Simple Authentication and Security Layer dans LDAP v3 [RFC 2251]. SASL est considéré comme sécurisé au niveau des connexions client/serveur.

2.5 Conclusion

Dans les années à venir, le système de gestion de politiques qui intègre plusieures techniques émergentes (par exemple, LDAP), jouera un rôle prépondérant dans la gestion des réseaux. Il permettra d'unifier les différents niveaux de gestion, particulièrement ceux des services et des équipements, et de faciliter l'automatisation de la reconfiguration des ressources.

Toutefois, pour que cela se produise, il faudra clairement spécifier les mécanismes de contrôle de politiques qui peuvent permettre de diminuer la complexité de la gestion des réseaux de l'Internet (gestion inter et intra-domaine). Dans ce but, nous proposons, dans le prochain chapitre, une architecture de gestion de politiques.

Chapitre 3

Système de gestion et de contrôle de politiques de gestion

3.1 Motivation et Objectifs

Le réseau mondial (l'Internet) est constitué de différents réseaux qui appartiennent à des institutions différentes. Le réseau de chaque institution contient une multitude d'objets provenant de fabricants différents. De plus, le développement des nouvelles technologies de l'information influence constamment la stabilité de la structure de ces réseaux. Cela crée de nouveaux besoins de gestion et la nécessité de changement d'approche de gestion.

Parmi les nouveaux besoins de gestion, il y' a:

- La gestion dynamique pour faciliter la reconfiguration automatique des composants.
- La gestion intégrée pour combiner les différents niveaux de gestion.
- La gestion évolutive pour mettre à la disposition des administrateurs des architectures de systèmes de gestion ouvertes et extensibles.

Dans cette thèse de maîtrise, nous voulons réaliser un système de gestion permettant de répondre à ces trois besoins. En effet, les systèmes de gestion standards existants dont SNMP, le plus largement déployé, ont montre leurs limites, en particulier pour assurer une gestion dynamique essentielle pour les réseaux d'aujourd'hui. C'est pour

cette raison que nous avons choisi un système à base de politiques. L'idée sous jacente a la gestion à base de politiques est de séparer les politiques de gestion des applications de gestion permettant ainsi de modifier les politiques en vigueur ou d'introduire de nouvelles politiques sans pour autant avoir a arrêter le processus de gestion et recompiler l'application de gestion. Dans les approches traditionnelles les politiques étaient codées en "hard" dans l'application de gestion.

Bien que la gestion à base de politiques reçoit une attention grandissante, elle demeure toutefois encore en phase de standardisation; plusieurs aspects ne sont pas encore clairs ou maîtrisés.

Cependant, la gestion à base de politiques peut être combinée à d'autres standards de gestion (tel que, SNMP qui est très bien avancé dans la collecte des statistiques) pour accomplir une gestion intégrée et efficace [San 99]. Dans notre recherche, nous avons combiné le système de gestion de politiques et le système standard de gestion SNMP. Ceci permettra à l'administrateur de :

- Centraliser les informations de gestion pour une application uniforme dans le réseau et/ou les stocker localement dans les composants pour une application spécifique et individuelle.
- Faire la configuration dynamique des composants du réseau.
- Distribuer les informations de gestion dans le réseau, grâce à l'utilisation de protocoles de stockage spécialisés.
- Evaluer la performance des ressources du réseau à partir des statistiques de fonctionnement collectées.
- Faire une gestion évolutive des composants du réseau.

Dans SNMP, les interactions gestionnaires/agents SNMP obéissent au modèle client/serveur où le gestionnaire est considéré comme client et l'agent comme serveur répondant aux requêtes du gestionnaire. Dans une telle architecture, le gestionnaire, faisant les requêtes, peut être un goulot d'étranglement. Pour éviter cela, nous utilisons la technologie d'agent mobile à la place du client serveur.

Notre objectif est qu'au terme de cette recherche, nous ayons un système de gestion de politiques permettant de :

(1) Editer et modifier les politiques de gestion à travers d'une interface graphique facile d'utilisation.

(2) Configurer dynamiquement les politiques au niveau des éléments gérés auxquels ces politiques s'appliquent. Ceci à travers de protocoles standards d'accès a distance et de distribution de l'information (ici les politiques de gestion). Les agents mobiles sont aussi utilisés ici pour optimiser la distribution de politiques de la console de l'administrateur ou la base de stockage des politiques vers les composants du réseau géré.

(3) Fournir a l'administrateur (l'utilisateur de notre système de configuration de politiques) un accès intégré convivial aux statistiques sur le fonctionnement du réseau. Ce qui permettra a celui ci d'avoir une vue global sur l'état du réseau géré et donc de décider l'évolution de l'application de gestion, en introduisant de nouvelles politiques par exemple. Cet objectif est assure à travers une intégration au niveau de l'interface d'administration de la gestion SNMP et celle à base de politiques.

3.2 Architecture du système

L'architecture de tout réseau évolue avec le développement de l'entreprise qui l'exploite, amenant ainsi de nouveaux besoins tout au long du cycle de vie du réseau. Dans un réseau d'entreprise, peut s'effectuer plusieurs opérations différentes, nécessitant parfois l'utilisation de toutes les fonctions de gestion pour le suivi et le contrôle de tous les aspects du réseau (tels que, la sécurité, les configurations personnalisées des utilisateurs, les coûts d'utilisation des ressources pour une facturation régulière, la performance des ressources, etc.). Pour permettre aux administrateurs d'étendre la gestion de leurs réseaux selon l'évolution de leurs architectures ou l'évolution des besoins de leur entreprise, nous basons l'architecture de notre système de gestion et de contôle sur une architecture de système de gestion de politiques, illustrée par la figure 3.1.

L'architecture proposée est composée d'une ou plusieurs applications de gestion de politiques, d'un ensemble de mécanismes de contrôle de politiques, et d'un système de stockage de politiques.

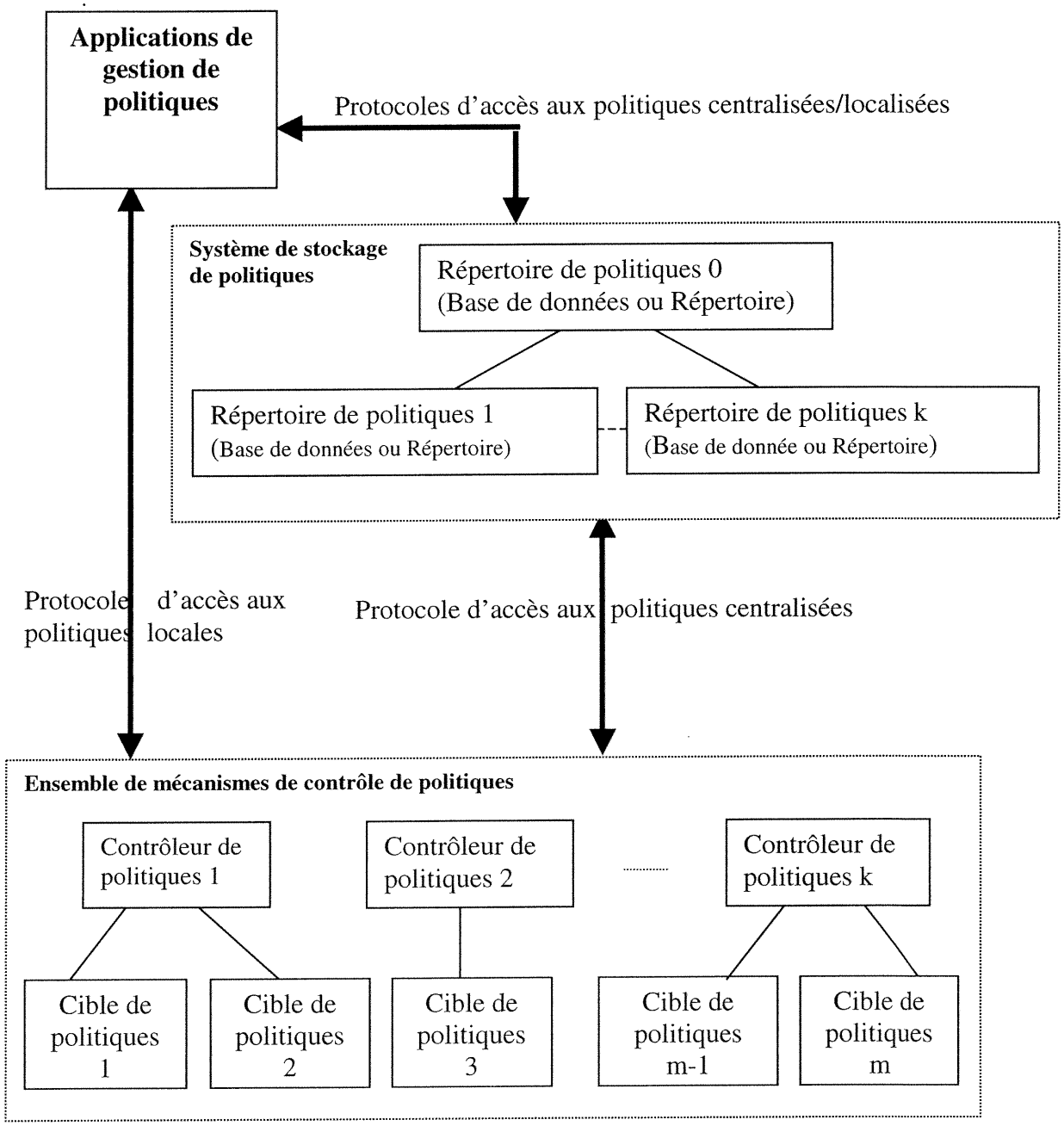


Figure 3.1 Architecture du système de gestion et de contrôle de politiques

- L'ensemble des applications de gestion de politiques permet, non seulement la configuration (visualisation, édition, modification et recherche) des politiques dans l'ensemble de stockage de politiques, mais aussi le contrôle (collecte de statistiques et optimisation) des objets gérés.

- Le système de stockage de politiques permet la centralisation logique et la distribution physique des politiques dans le réseau.

- L'ensemble des mécanismes de contrôle (par domaine ou niveau) de politiques permet la prise de décision et l'exécution d'implantation de politiques dans le réseau.

Dans notre implémentation nous développons un ensemble d'outils de gestion qui nous permettent de configurer les politiques dans un serveur LDAP et de contrôler les ressources ayant des agents SNMP. Le mécanisme de contrôle de politiques que nous proposons pour notre implémentation est le protocole normalisé de contrôle de politiques COPS [RFC 2748].

Autrement dit, nous configurons les informations de gestion stockées localement dans les objets gérés du réseau en utilisant SNMP et nous configurons les politiques de gestion stockées dans une base de donnée centrale (serveur LDAP) en utilisant le protocole LDAP. Les politiques sont distribuées aux ressources gérés avec le protocole COPS (figure 3.2).

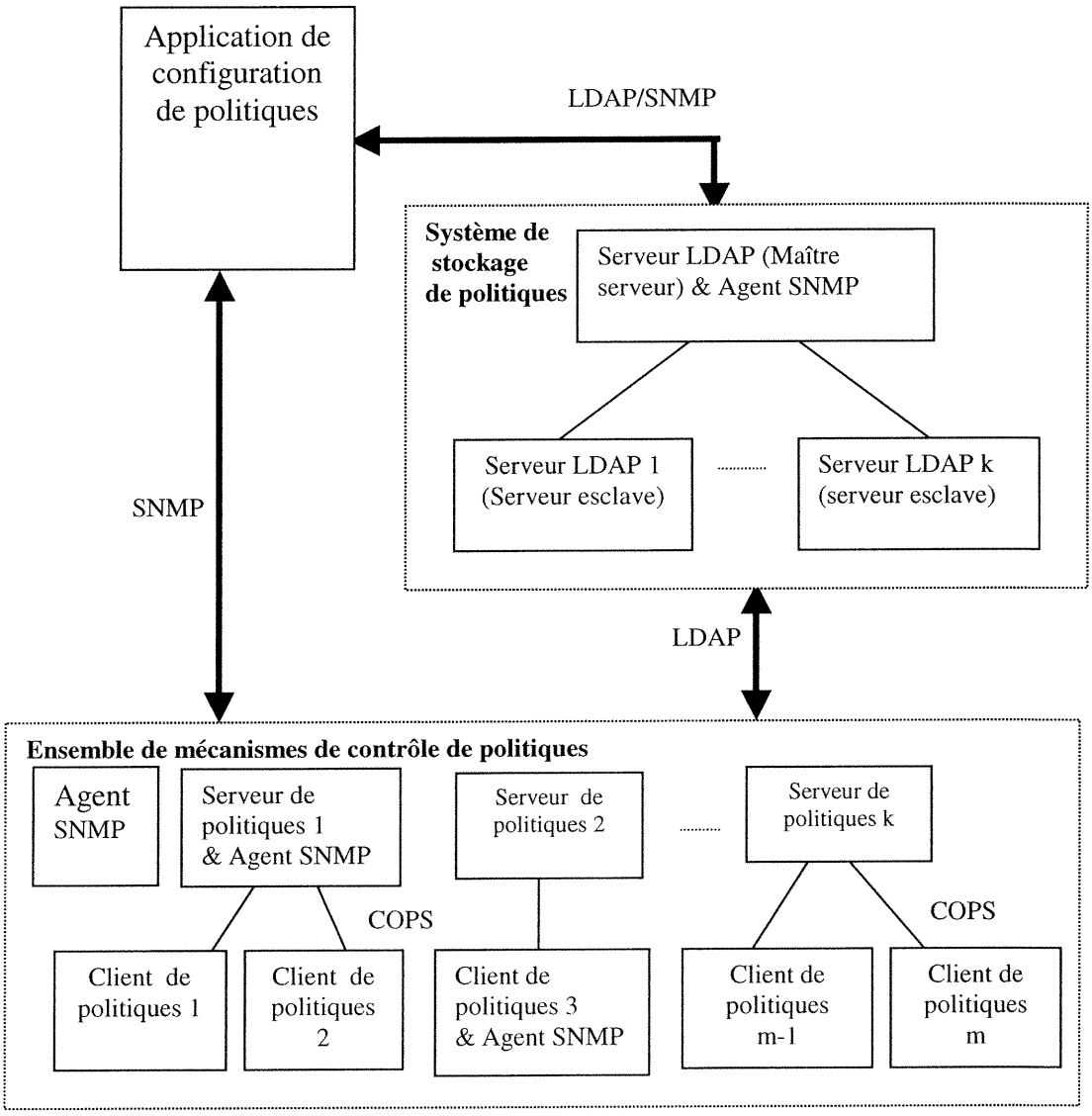


Figure 3.2 Architecture détaillée du système

3.2.1 Application de configuration

La gestion de la configuration s'occupe de la configuration statique, qui est une construction initiale des caractéristiques opérationnelles des éléments gérés du réseau, et de la configuration dynamique ou reconfiguration, qui concerne les modifications

subséquentes des caractéristiques des éléments gérés pendant leur cycle de vie dans le réseau.

La configuration initiale du réseau est la base du bon fonctionnement du réseau, elle est mise en place dès l'installation du réseau. La configuration dynamique est nécessaire pour adapter le réseau au changement opérationnel et évolutif du réseau. Cette configuration dynamique peut s'effectuer à chaque fois que cela est nécessaire.

L'un des objectifs principaux du système de gestion de politiques est d'effectuer la configuration dynamique. Ceci est réalisé en deux phases. La première phase consiste à stocker dans une base de donnée toutes les informations nécessaires (quand est ce qu'il faut faire la modification, la valeur de la modification, l'élément ou les éléments concernés par la modification, etc.). La deuxième phase consiste a rendre ces informations accessibles à travers le réseau pour permettre au mécanisme d'implantation d'effectuer la modification sur les composants concernés.

Pour cela, les deux outils de configuration développés au niveau de l'interface d'administration intégrée: un client LDAP et un gestionnaire SNMP, figure 3.3.

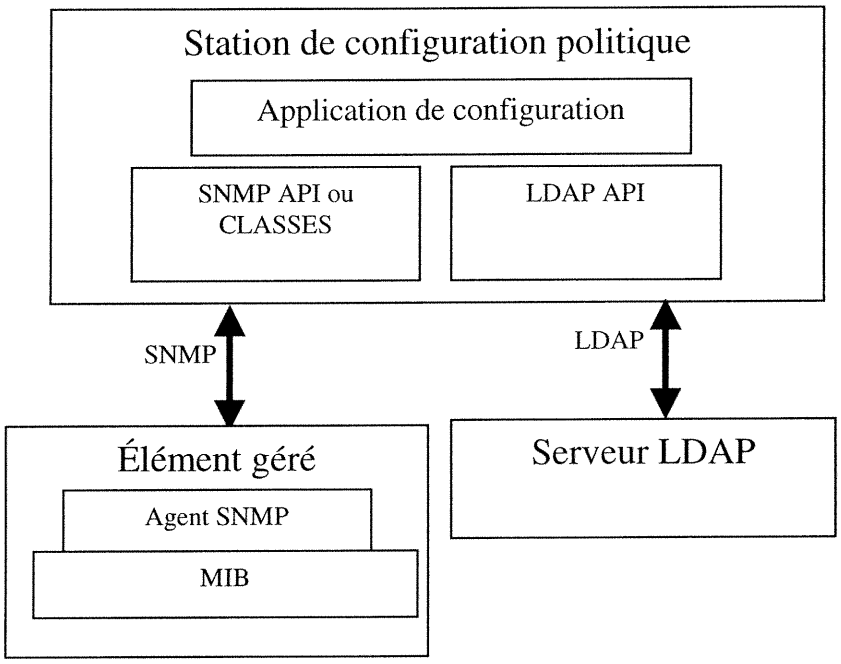


Figure 3.3 Interactions client/serveur LDAP & gestionnaire/agent SNMP

3.2.1.1 Manager SNMP

Le gestionnaire SNMP permet à l'administrateur de suivre et de configurer les éléments utilisant SNMP comme protocole de gestion. Dans le gestionnaire SNMP, nous utilisons la notion d'agent mobile pour diminuer les messages transmis entre le manager et les agents.

Pour réaliser cet outil avec les agents mobiles, les éléments sur lesquels s'exécutent les agents SNMP doivent être capables de recevoir les agents mobiles. Pour cela, ils doivent comprendre une plate-forme d'accueil d'agents mobiles pouvant permettre à ces derniers de continuer leurs activités.

Ainsi, le manager SNMP déploie un agent mobile sur les éléments gérés où se trouvent les agents SNMP pour interagir avec eux. Si l'agent mobile doit interagir avec plusieurs agents SNMP, il ira d'un agent à un autre jusqu'à ce que toutes les requêtes soient finies, et retourne avec le résultat des requêtes à la station où se trouve le manager SNMP (figure 3.4).

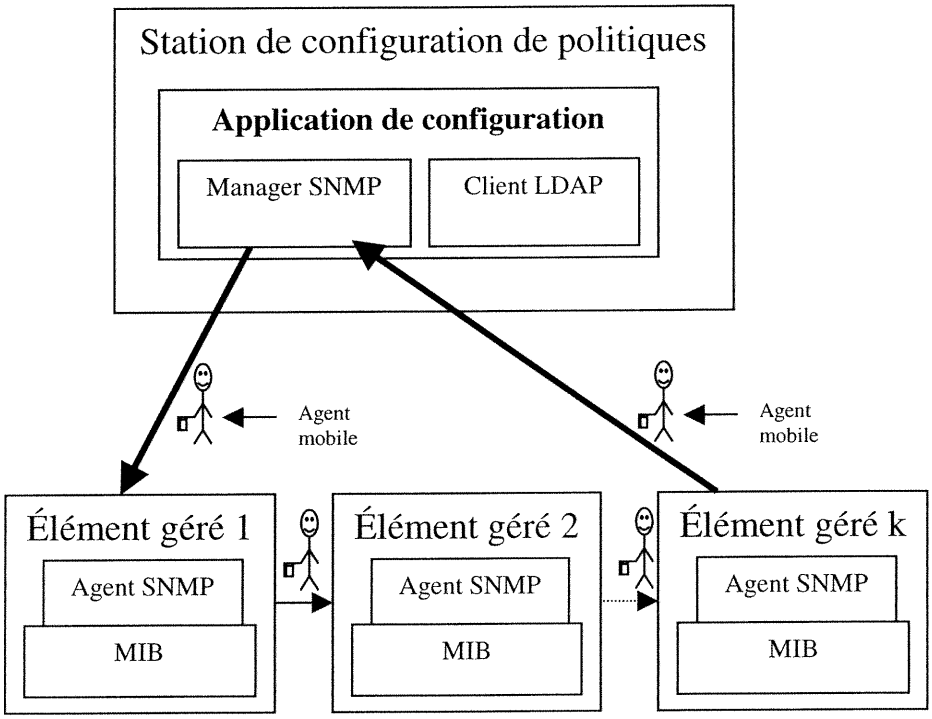


Figure 3.4 Interaction gestionnaire/agents SNMP

3.2.1.2 Client LDAP

Le client LDAP permet à l'administrateur de visualiser et rechercher les politiques stockées dans un serveur LDAP, tout en permettant à un administrateur autorisé de modifier (ajouter, replacer ou effacer) une ou plusieurs politiques dans un serveur

LDAP, figure 3.5.

Le client LDAP développé dans cette thèse, peut se connecter à n'importe quel serveur LDAP sur l'Internet, permettant ainsi à l'administrateur de configurer les données à partir de n'importe où sur le réseau.

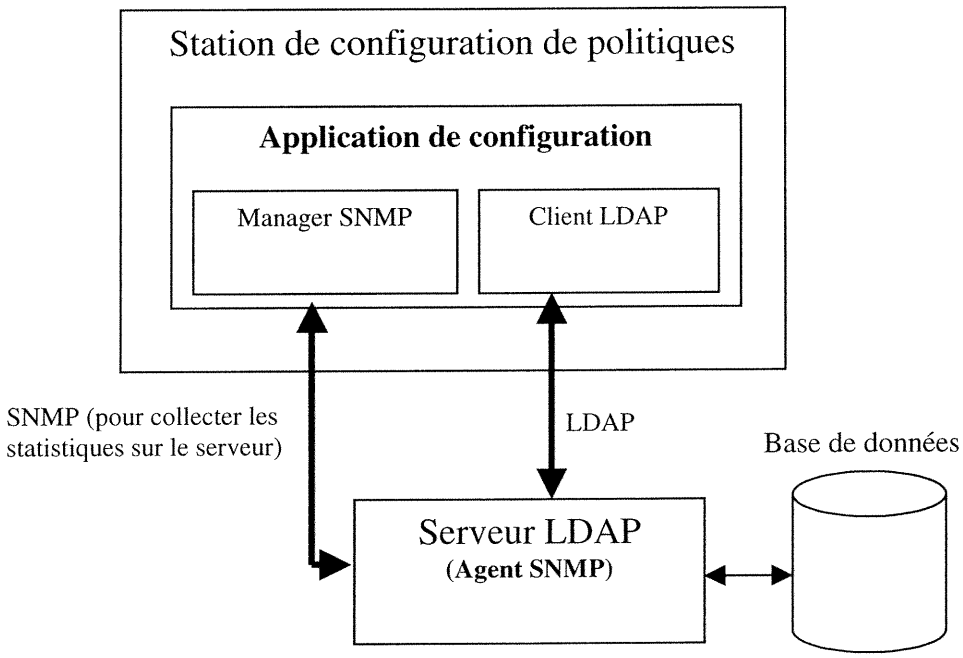


Figure 3.5 Interaction client/serveur LDAP

3.2.2 Stockages de politiques

Dans un système de réseau large, il peut exister une variété d'objets à gérer, rendant ainsi la gestion de la configuration difficile pour les administrateurs. Un moyen d'alléger cette difficulté est de regrouper les objets par domaine. Cette organisation diminue la complexité de la gestion du réseau, augmente sa flexible et sa performance.

Nous structurons l'arborescence de stockage de politiques selon le domaine de gestion

ou les services à gérer dans le réseau (QdS, IPSec, VPN, etc.). Bien entendu, d'autres critères et niveaux de structuration peuvent être utilisés aussi.

L'utilisation de Lightweight Directory Access Protocol (LDAP) facilite cette structuration, figure 3.6. Le service de répertoire LDAP permet la distribution et la reproduction des modifications d'informations dans le stockage. Aussi, il permet aux mécanismes de contrôle de politiques de rechercher les politiques appropriées pour les éléments gérés dont ils supervisent la gestion.

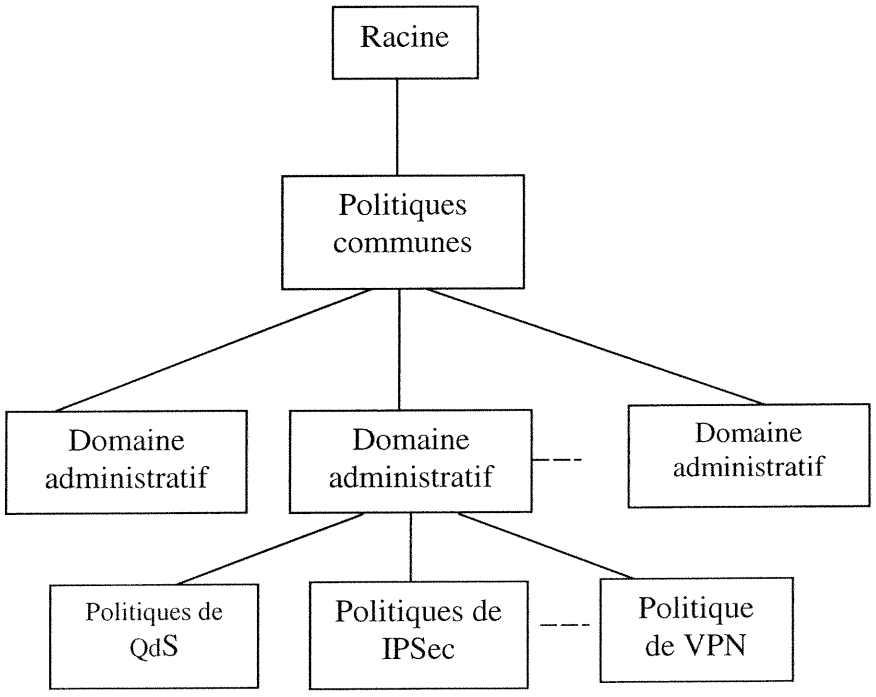


Figure 3.6 Structure du Stockage de politiques

Ainsi, les serveurs LDAP contiennent des politiques de gestion communes à plusieurs domaines et services réseau (ou sommet de l'arborescence de politiques) ainsi que celles spécifiques à un domaine de gestion ou un type de service à gérer dans le

réseau, par exemple, la qualité de service (QdS Policy), la sécurité du réseau IP (IPSec Policy), et le réseau privé virtuel (VPN). L'extension à d'autres politiques est possible et facile grâce aux caractéristiques de LDAP.

3.2.3 Mécanismes d'application de politiques

Plusieurs mécanismes peuvent être utilisés pour appliquer dans le réseau les politiques stockées dans des serveurs LDAP. Il suffit simplement, pour ce faire, que le mécanisme choisi ait la capacité de rechercher les politiques dans les serveurs LDAP, et la capacité de les adapter aux cibles de politiques (les composants gérés). Notre système se base sur deux mécanismes principaux qui sont le ″Common Open Policy Service (COPS) ″ et le ″Simple Network Management Protocol (SNMP)″.

Un client LDAP est utilisé au niveau du serveur COPS pour rechercher les politiques de gestion dans les serveurs LDAP.

Tel que mentionné auparavant, les réseaux actuels sont constitués par une interconnexion de réseaux parfois sous différentes autorités administratives. Dans de tels réseaux, les besoins en matière de gestion ne sont pas toujours les mêmes ; les interactions des clients de politiques dans les sous-réseaux avec un seul serveur de politiques diminuent la disponibilité de celui-ci, affectant ainsi la performance de l'application des politiques dans le réseau. Organiser la gestion par domaine et l'utilisation de serveurs de politiques multiples est un moyen pour alléger la charge du serveur de politiques et améliorer le temps de réponse des requêtes de politiques.

Dans notre système, plusieurs serveurs COPS sont utilisés. Chaque serveur COPS interagit avec le serveur LDAP, qui est physiquement distribué, pour obtenir les politiques qui le concernent. Cela nous permet de décentraliser les prises de décision d'application de politiques de gestion, et d'augmenter la disponibilité des serveurs

d'application de politiques.

Pour une gestion intégrée, l'idéal est d'utiliser un seul mécanisme d'application de politiques pouvant satisfaire tous les domaines de gestion et pouvant se servir d'une même spécification de politiques. Ce besoin est parmi les causes qui ont conduit au développement de normes pour l'implémentation de systèmes de gestion de politiques pour l'opération et la gestion des réseaux IP. Pour l'instant, ce processus rencontre les difficultés suivantes :

- La gestion basée sur les politiques est encore en développement ; plusieurs aspects des mécanismes développés pour appliquer les politiques de gestion dans les réseaux ne sont pas encore clairs ou maîtrisés par les développeurs.

- Difficulté d'interprétation des mécanismes en fournissant une seule spécification d'information de politiques. Le modèle d'information du système de gestion de politiques (Policy CIM) peut être traduit dans plusieurs spécifications (LDAP, PIB, MIB), et ces spécifications ne sont pas inter-opérables pour le moment.

- Le protocole SNMP reste pour le moment le standard de gestion pour IP ; il y est largement déployé et est utilisé dans le suivi et la collecte de statistiques des composants IP. Cependant, SNMP n'est pas très utilisé pour la gestion de la configuration [McC 99] [San 99].

Pour toutes ces raisons, nous avons jugé nécessaire d'inclure SNMP dans les mécanismes d'application de politiques. SNMP permettra non seulement de gérer les éléments impliqués dans l'application de politiques et dans le stockage des informations, mais aussi de surveiller leur performance.

3.3 Conclusion

L'architecture des réseaux actuels est dynamique. Une gestion adaptée est donc requise. Le système de gestion de politiques est un bon candidat pour réaliser une gestion dynamique. Il permet le regroupement des informations de gestion des ressources gérées, facilite leur distribution grâce à l'utilisation des services de répertoire, et surtout, il possède une structure qui dissocie les informations de gestion (les politiques) des applications de gestion d'une composant particulier.

La dissociation des différentes abstractions du système de gestion de politiques facilite son utilisation pour une gestion par domaine. Une architecture de système de gestion basée sur de tels concepts (i.e., les politiques et les domaines) est extensible et adaptable à l'évolution rapide des réseaux d'aujourd'hui.

Ces avantages du système de gestion de politiques nous ont motivé à le choisir comme composant principale de notre architecture. Notre objectif étant d'avoir un système de gestion permettant à tout administrateur de réseau, de configurer les paramètres de contrôle du réseau, de configurer et reconfigurer dynamiquement le réseau et de suivre l'état de fonctionnement du réseau tout en permettant une gestion évolutive, nous avons organisé les politiques par domaine et utilisé SNMP dans l'architecture pour surveiller le fonctionnement du réseau.

Dans cette architecture, nous avons les applications de gestion des paramètres de contrôle du réseau, le système de stockage et les mécanismes de contrôle de ces paramètres. L'architecture de système de gestion de politiques développée dans ce chapitre peut être utilisée aussi bien pour la gestion des réseaux simples que pour celle des réseaux complexes.

Chapitre 4

Application de configuration de politiques

Dans ce chapitre, nous faisons une brève description de la programmation orienté-objet. Puis nous, introduisons le langage de programmation Java, et le framework Voyager utilisés pour réaliser cette application. Nous terminons le chapitre par l'implantation et des exemples d'exécution de l'application.

4.1 Langage de programmation Java

Java est un langage de programmation développé par Sun Microsystems [SunMS]. Il est généralement mentionné dans le cadre du World Wide Web (WWW), au sein duquel opèrent des navigateurs "compatibles Java", comme Netscape Navigator de Sun Microsystems ou Internet Explorer de Microsoft. Un navigateur compatible Java se distingue par sa capacité à télécharger et exécuter des applets sur le système de l'utilisateur.

Java a plusieurs avantages qui incitent un programmeur à l'utiliser comme outil de programmation.

- Il est un langage qui ne dépend d'aucune plate-forme et système d'exploitation ; ce qui signifie que le programmeur peut utiliser le même code dans différentes plate-formes, tels que Solaris, Unix, Macintosh, Windows 95/98/NT, etc. Ce qui est nécessaire pour la programmation Internet qui est hétérogène en plate-formes et systèmes d'exploitation.

- Il est complètement orienté-objet, sauf quelques types basiques.

- Il est un langage dont la syntaxe est plus simple mais similaire à celle des langages de programmation C et C++. La simplicité de Java n'a d'effet négatif ni sur sa puissance ni sur sa souplesse.

- Il possède une importante librairie de classes qui permet d'accomplir la distribution, l'accès à des bases de données à travers le réseau, la sécurité, etc.

Pour le développement de notre application, nous utilisons Java Naming and Directory Interface (JNDI) pour accéder au serveur LDAP. JNDI est une API (Application Programming Interface) qui fournit la fonctionnalité annuaire et le nomage aux applications Java. Elle est définie pour être indépendante de toute implémentation spécifique (figure 4.1).

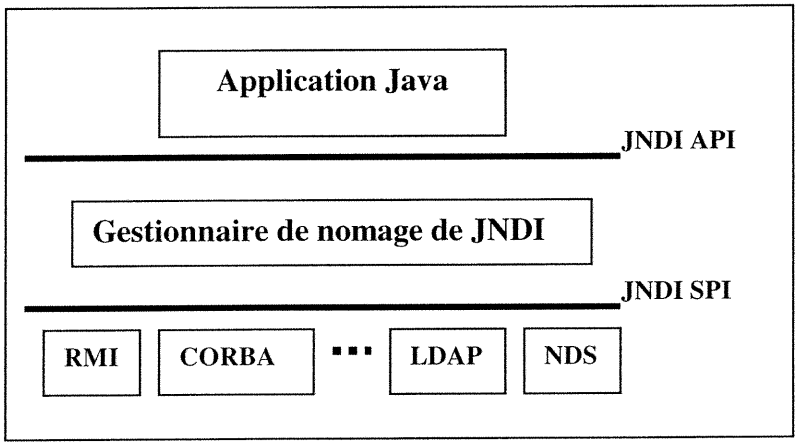


Figure 4.1 Architecture de JNDI

JNDI API permet aux applications java d'accéder à une variété de services de répertoire et de nomage d'une manière commune. JNDI Service Provider Interface permet à un grand nombre de services annuaires d'être insérés de façon transparente dans les applications Java.

4.3 Voyager de l'Objectspace

Voyager (Voyager Core Technology) est un système d'agent mobile développé par Objectspace [VOY]. Il est complètement implémenté en Java et utilise le modèle objet de Java. Il permet la construction des objets à distance, la communication par messagerie entre les objets, et la migration d'objets entre les programmes.

Voyager a la capacité de distribuer des événements, de déplacer des objets dans le réseau et de permettre aux objets de communiquer pendant qu'ils se déplacent. Les programmeurs peuvent se servir de Voyager pour concevoir des interfaces, et les faire communiquer par passerelle (proxy). Voyager garde dans la mémoire cache la référence des objets, une référence qui est mise à jour dès que l'objet se déplace vers une nouvelle machine. Ces capacités sont étendues aux agents qui peuvent communiquer avec d'autres agents en leur envoyant des messages pour s'auto-détruire une fois la tâche terminée.

De plus, il supporte et intègre d'aspects liés à la programmation Java. Parmi eux, il y a:

- CORBA : Voyager peut obtenir une référence virtuelle d'un objet CORBA, CORBA aussi peut obtenir à distance une référence d'un objet Voyager.

- JavaBeans : les utilisateurs peuvent utiliser n'importe quel auditeur de JavaBeans à distance.

- Sécurité : Voyager est fourni avec un gestionnaire de sécurité dont l'utilisation est facultative, ce gestionnaire limite les exécutions des objets étrangers au système.

- Applets : Voyager n'impose pas de restriction de communication entre les objets dans le browser et ceux hors du browser.

- Startup : le Framework de lancement de Voyager permet aux objets d'être persistants et d'être automatiquement réactivés au cas où la machine virtuelle (Java Virtual Machine) serait relancée.

- Extensibilité : Voyager permet aux programmeurs d'établir une liaison entre les modules sous-espaces pour former un espace virtuel. Ainsi, les utilisateurs peuvent communiquer avec l'espace virtuel comme si c'était un objet simple. Quand des messages sont envoyés à l'espace virtuel, ils sont copiés dans les modules en parallèle.

- Service de répertoire et de nomage: ces services permettent aux programmeurs d'établir et joindre des objets dans un système réparti pour leur gestion. En utilisant les stratégies de nomage, les programmeurs peuvent accéder à un objet existant sans connaître son emplacement physique.

4.4 Classes SNMP de Adventnet

Nous utilisons aussi dans cette application Adventnet SNMP API. Adventnet SNMP API est totalement écrite en Java. Cette API permet aux développeurs d'applications de gestion SNMP et aux concepteurs de MIB SNMP de rapidement implanter des applications de gestion basées sur le Web, d'implanter des agents SNMP en Java, et de facilement compiler les MIB SNMP.

4.5 Implantation de l'application

Aux termes de notre architecture développée dans le chapitre précédent, nous avons implanté une application de gestion permettant de configurer les politiques de gestion du réseau (paramètres de contrôle des éléments gérés et services) et de collecter les statistiques de gestion. Cette application permet à tout utilisateur de se servir d'une interface graphique pour éditer, modifier et rechercher les politiques de gestion du réseau dans un serveur LDAP. Elle permet aussi d'utiliser SNMP pour modifier la valeurs des attributs des objets gérés du réseau et collecter les statistiques de fonctionnement du réseau.

Nous avons les sous-applications suivantes dans cette implantation : un client LDAP et un gestionnaire SNMP. Nous avons intégré ces deux sous-applications de gestion (LDAP client et SNMP manager) au niveau de l'interface graphique, pour permettre à l'administrateur de manipuler les données de l'annuaire et celles des MIBs SNMP parallèlement. Une telle intégration rend les deux outils indépendants l'un de l'autre, augmentant ainsi la flexibilité de la gestion à partir de l'application. Un administrateur peut se servir de l'un ou/et l'autre selon son besoin. Ainsi, il peut surveiller les serveurs LDAP, les serveurs et clients COPS avec SNMP manager.

Pour réaliser cette intégration au niveau de l'interface graphique, l'application a besoin d'une conception séparée du client LDAP et manager SNMP. Ainsi, nous avons mis toutes les classes dont a besoin le manager SNMP dans le module snmp, toutes les classes dont a besoin le client LDAP dans le module ldap. Nous avons par la suite, intégré ces deux modules dans le module manager. La figure 4.2 montre cette conception en utilisant UML.

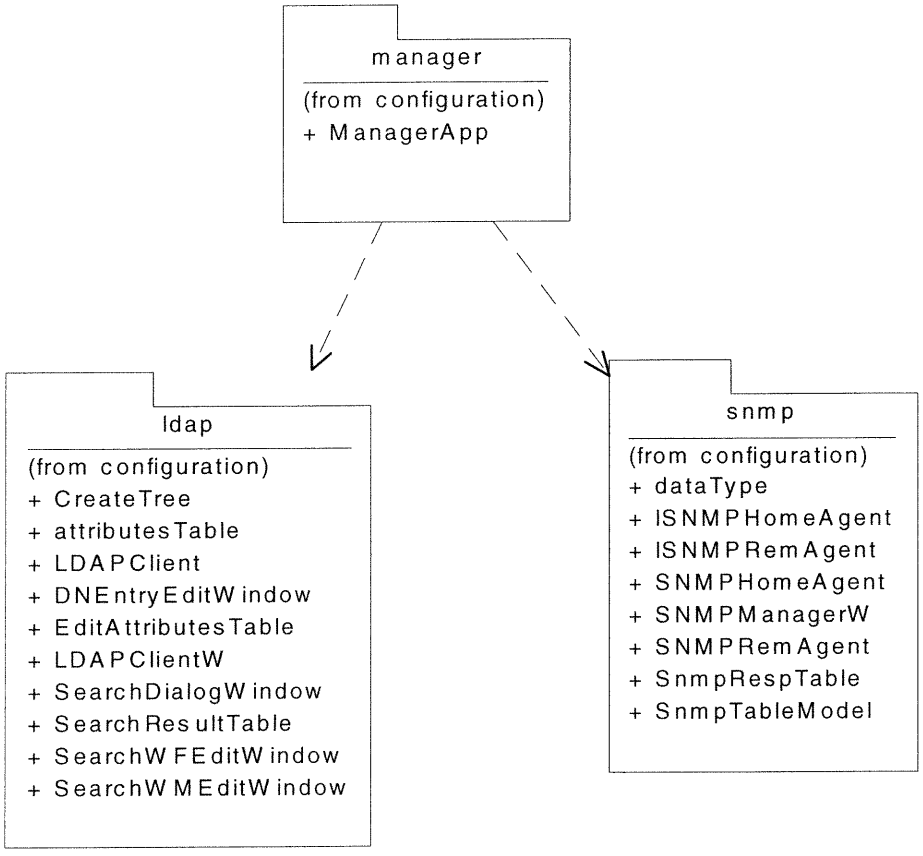


Figure 4.2 Modules UML de l'application

Les relations entre les différentes classes de chaque module montrent comment les classes dépendent les unes des autres. La figure 4.3 montre l'interdépendance entre les classes du module ldap.

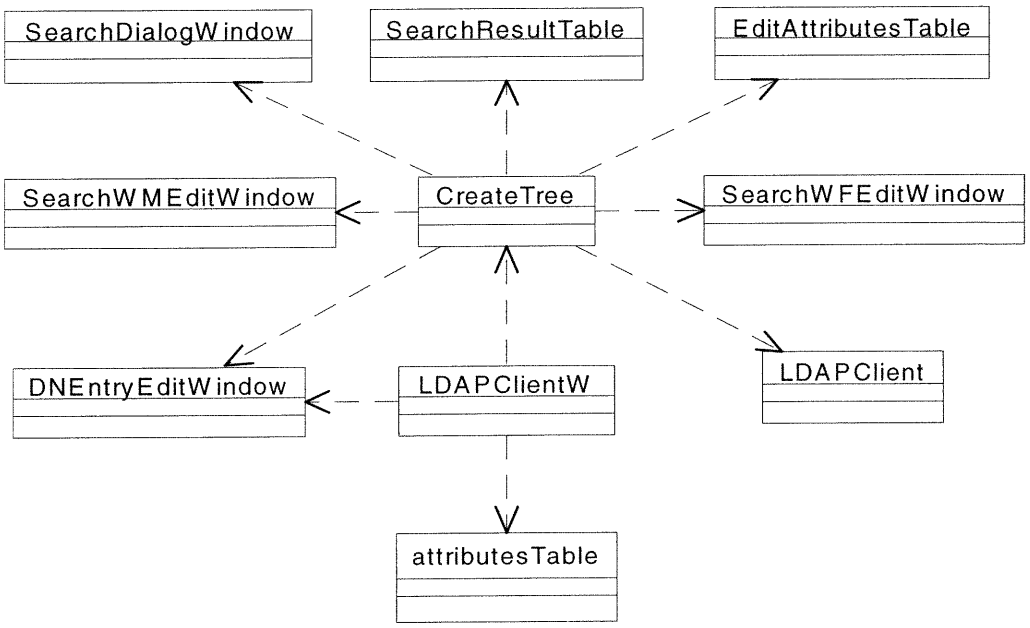


Figure 4.3 Classes UML du module ldap

Nous avons au total dix classes dans le module ldap. Ces dix classes sont :

LDAPClientW.java : pour concevoir l'interface graphique du client LDAP. Toutes les demandes de l'administrateur s'effectuent à partir de cette interface. La classe **LDAPClientW.java** utilise les classes **attributesTable.java, CreateTree.java** et **DNEntryEditWindow.java**.

CreateTree.java : est une classe qui permet la construction graphique de l'arborescence de données (LDAP Directory Information Tree). Elle utilise toutes les classes sauf **attributesTable.java** et **LDAPClientW.java**.

LDAPClient.java : est la classe qui utilise Java Naming and Directory Interface (JNDI) pour se connecter au serveur LDAP. Les opérations de recherche, d'entrée, de

suppression et de modification de politiques s'effectuent dans cette classe. Elle utilise une connexion sécurisée (plain texte) et une connexion anonyme.

AttributesTable.java : permet l'affichage des attributs et leurs valeurs correspondantes dans un tableau. Ainsi, quand l'administrateur choisit une entrée dans l'arborescence (Directory Information Tree), les attributs et leurs valeurs correspondantes affichées sont affichés dans un tableau. Cette classe utilise la classe **LDAPClientW.java**.

SearchDialogWindow.java,SearchResultTable.java, SearchWMEditWindow.java et SearchWFEditWindow.java: sont des classes qui permettent les différentes recherches de donnée dans l'arborescence de donnée.

DNEntryEditWindow.java : permet l'édition des entrées du DIT.

EditAttributesTable.java : aide à éditer les attributs utilisés pour les recherches.

La figure 4.4 présente l'ensemble des classes et interfaces contenues dans le module snmp. Dans ce module, nous avons **ISNMPHomeAgent.java, SNMPHomeAgent.java, ISNMPRemAgent.java, SNMPRemAgent.java, SNMPManagerW.java, SnmpRespTable.java et dataType.java**.

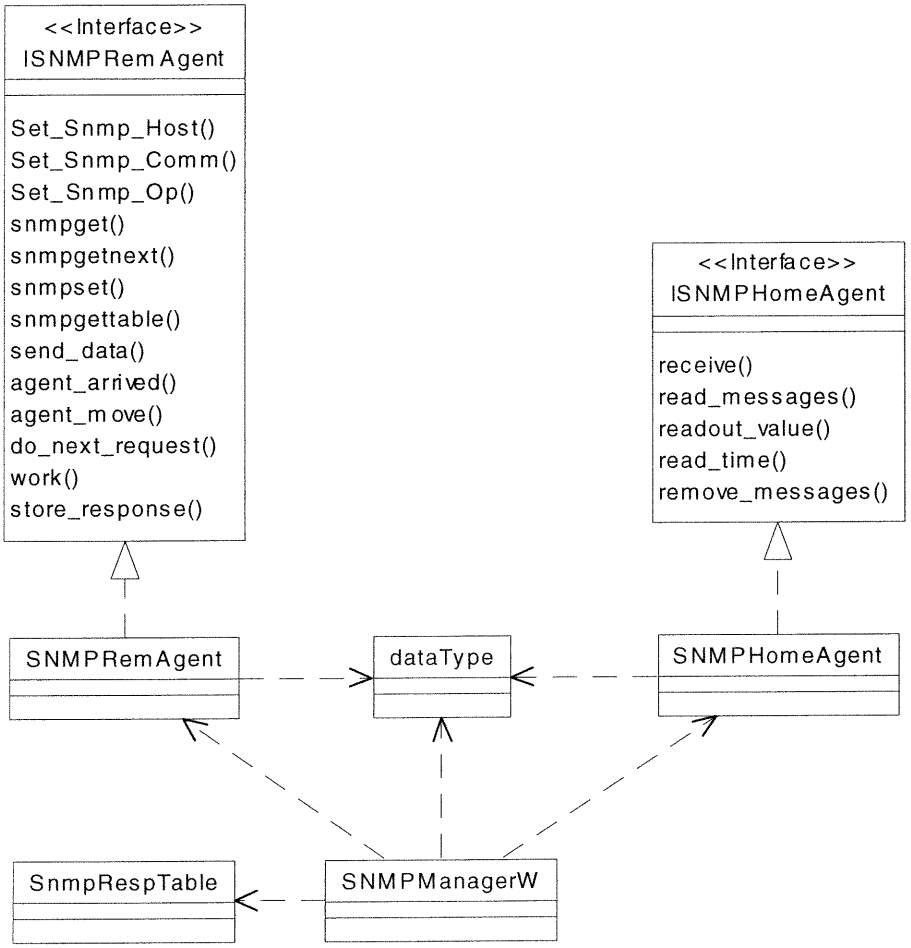


Figure 4.4 Classes UML du module snmp

ISNMPHomeAgent.java : est une interface dans laquelle sont définies les signatures des opérations locales.

SNMPHomeAgent.java: est une classe qui implémente **ISNMPHomeAgent.java**. Dans cette classe sont implémentées les opérations dont les signatures sont dans **ISNMPHomeAgent.java**.

ISNMPRemAgent.java: est une interface qui définit les signatures des opérations distantes.

SNMPRemAgent.java : implémente **ISNMPRemAgent.java**, **java.io.Serializable**. Elle définit les opérations distantes. Ces opérations peuvent être invoquées par l'agent mobile.

SNMPManagerW.java : permet la construction de l'interface graphique dans laquelle s'effectuent les différentes interactions manager/agents SNMP.

SnmpRespTable.java : permet l'affichage des résultats des requêtes dans un tableau.

dataType.java : permet de stocker le format de donnée que nous voulons avoir des requêtes (Nom de l'opération, Nom du hôte où se trouve l'agent SNMP, Liste d'identificateurs d'objet, Liste de valeurs).

Nous écrivons dans un fichier texte, le nom des serveurs voyager (au cas où l'agent mobile serait utilisé), le nom des hôtes (agents SNMP), le fichier MIB à charger (si cela est nécessaire), les différentes opérations et la liste des identificateurs d'objet sur lesquels doivent s'effectuer les opérations. Cela nous permet d'effectuer plusieurs opérations sur plusieurs éléments gérés à la fois.

Un exemple de fichier que nous utilisons est le suivant :

Ucdavis1.txt

TravelTo	***mot clé pour destination (serveur d'agent mobile)***
//ucd-snmp.ucdavis.edu:8000	***nom et port du serveur***
SNMPHost	***mot clé pour l'agent SNMP***
ucd-snmp.ucdavis.edu	***nom de l'agent SNMP***
SNMPCOMM	***mot clé pour community string***
Demopublic	***community string***
MIBFile	***mot clé pour la mib à charger***

nom de la mib
SNMPOper *mot clé pour l'opération à effectuer*
SNMP Get Next *nom de l'opération*
OID *mot clé pour l'identificateur d'objet*
1.2.0 *identificateur d'objet*
OID *mot clé pour l'identificateur d'objet*
1.3.0 *identificateur d'objet*

TravelTo
//ucd-snmp.ucdavis.edu:8000
SNMPHost
ucd-snmp.ucdavis.edu
SNMPCOMM
demopublic
MIBFile

SNMPOper
SNMP Get
OID
1.4.0
OID
1.5.0

Il est possible d'effectuer les opérations **SNMP Get**, **SNMP Get Next**, **SNMP Get Table** et **SNMP Set**.

Les résultats des opérations de requêtes sont affichés par la classe **SNMPManagerW.java**. Cette même classe permet de surveiller à temps réel un objet géré (par exemple, **sysUpTime**). Elle permet aussi de faire le graphe (en utilisant le résultat des opérations) d'un objet de plusieurs valeurs (par exemple, pour évaluer la valeur d'un objet pour des périodes différentes).

4.6 Exécution de l'application

La figure 4.5 est la principale GUI, qui permet au manager du réseau d'utiliser le client LDAP et le manager SNMP.

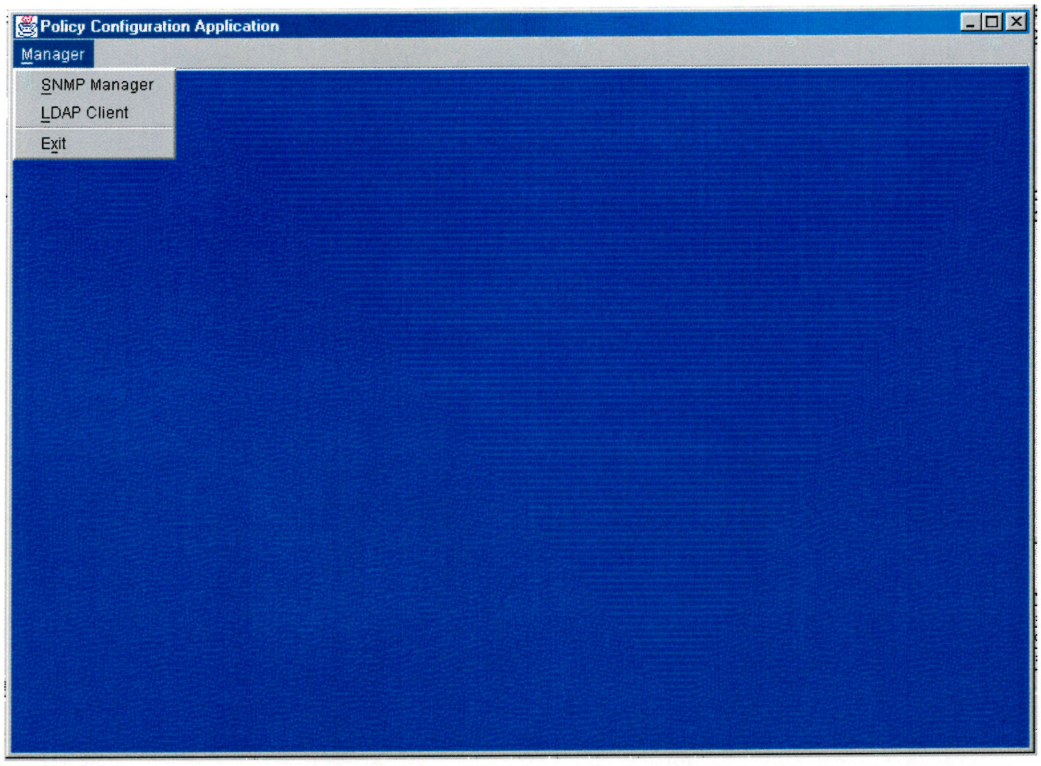


Figure 4.5 Principale GUI de l'application de configuration de politiques

Les figures 4.6(a) et 4.6(b) montrent la connexion du client LDAP que nous avons développé au serveur LDAP de l'Université de Michigan. A gauche, dans la figure, il y a le DIT (Directory Information Tree) du serveur; en haut à droite, il y a les informations du serveur (pour pouvoir se connecter); et à droite, en bas, il y a la table des attributs et les valeurs correspondantes du nœud courant de l'arbre.

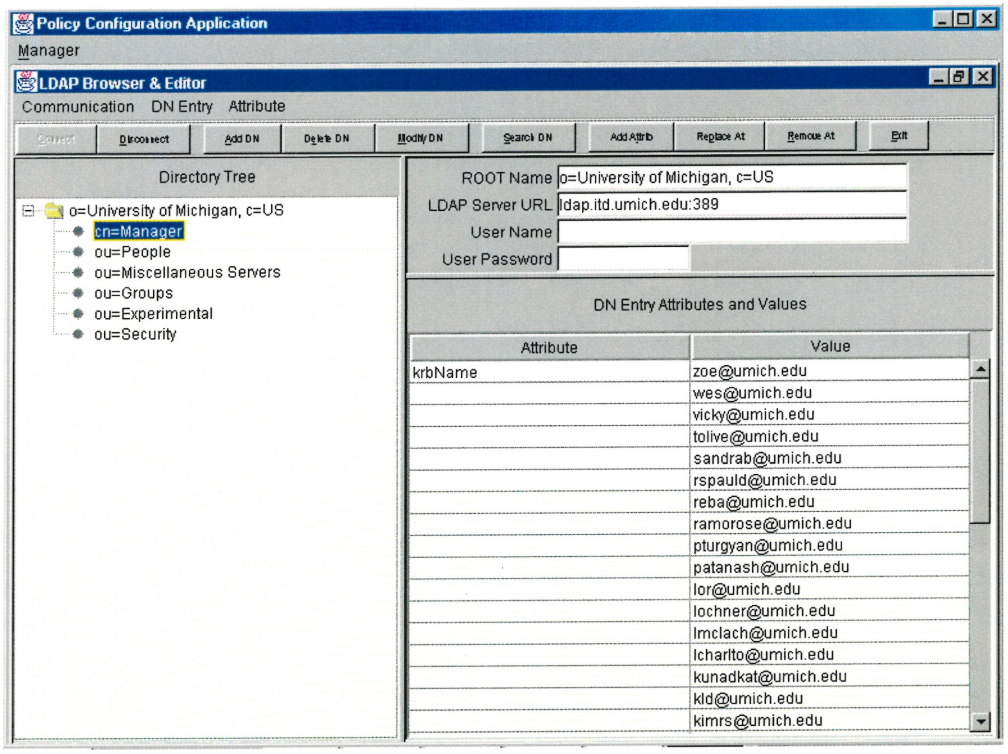


Figure 4.6(a) Connexion du client LDAP au serveur de Univ. de Michigan

La figure 4.6(b) est le résultat de l'extension de "ou=People", puis "ou=Students".

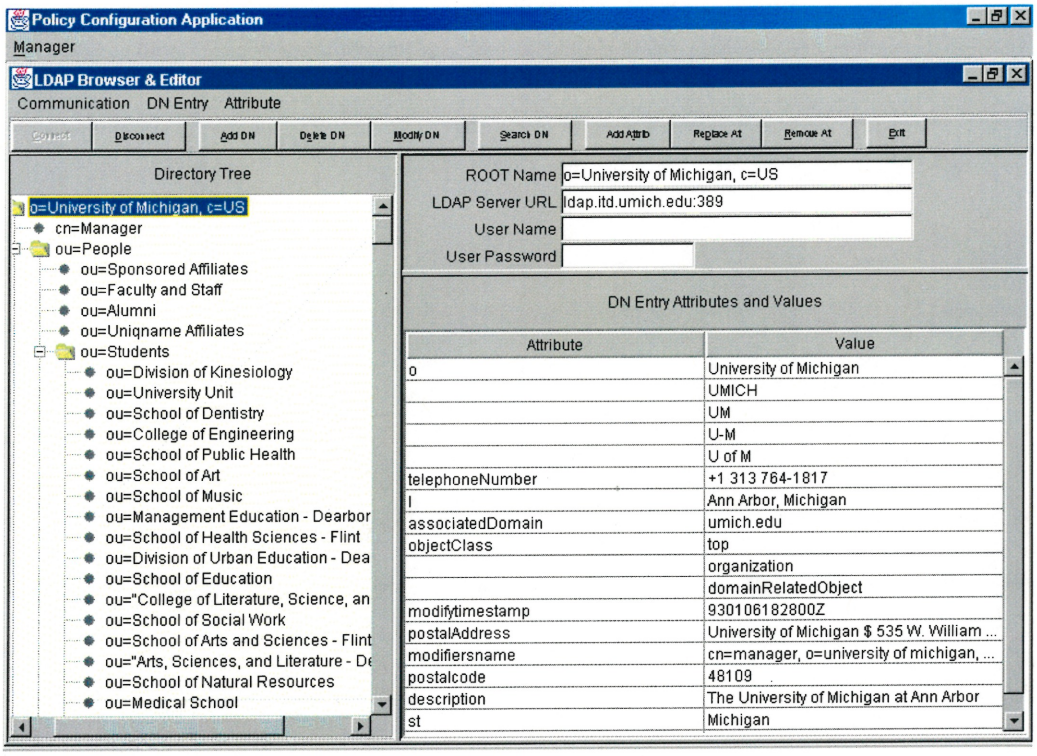


Figure 4.6(b) Extension de la branche ou=People

La figure 4.7 montre la connexion au serveur de Netherland.

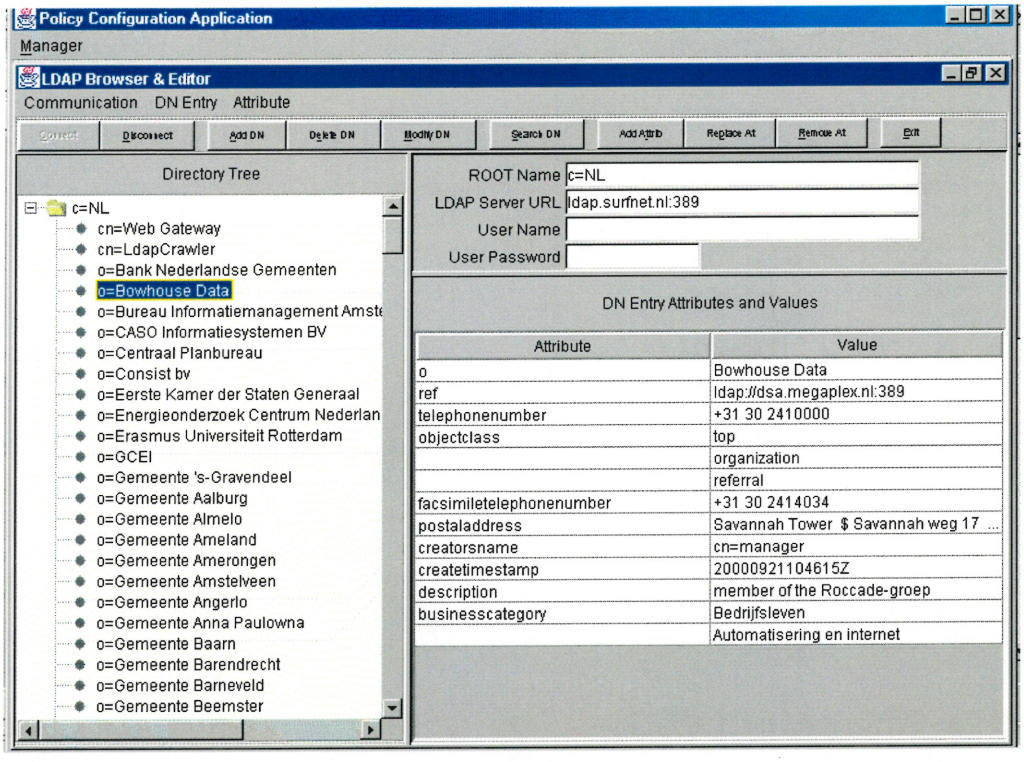


Figure 4.7 Connexion du client LDAP au serveur des Pays-Bas (Netherlands)

La figure 4.8 est la boîte de dialogue pour effectuer les différentes recherches dans le serveur.

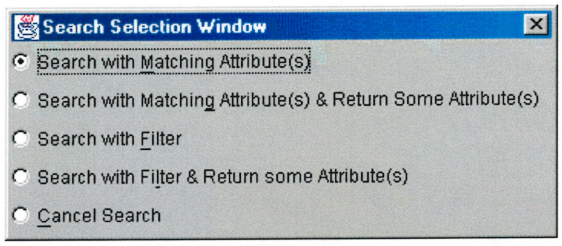


Figure 4.8 Boîte de dialogue pour les recherches

Les figure 4.9(a) et 4.9(b) sont les boîtes de dialogue pour entrer le filtre de recherche et les attributs de recherche.

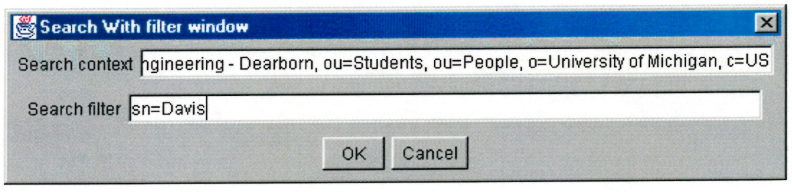


Figure 4.9(a) Boîte de dialogue pour entrer le filtre de recherche

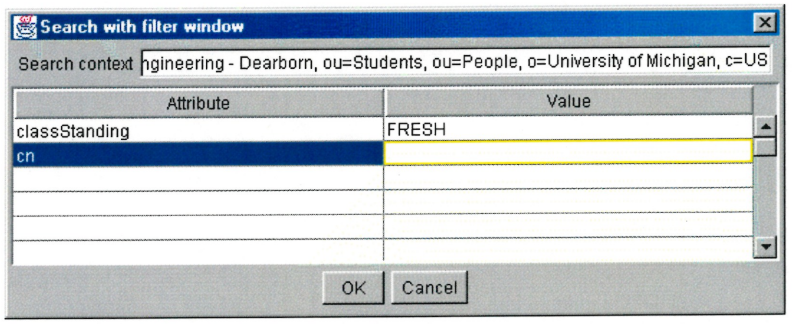


Figure 4.9(b) Boîte de dialogue pour entrer les attributs de recherche

Les figures 4.10(a) et 4.10(b) sont des résultats de recherche dans le serveur.

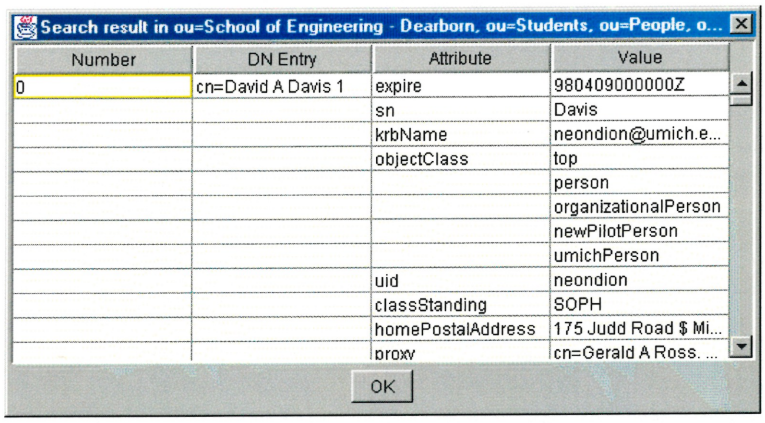


Figure 4.10(a) **Résultat de recherche pour le filtre sn=Davis**

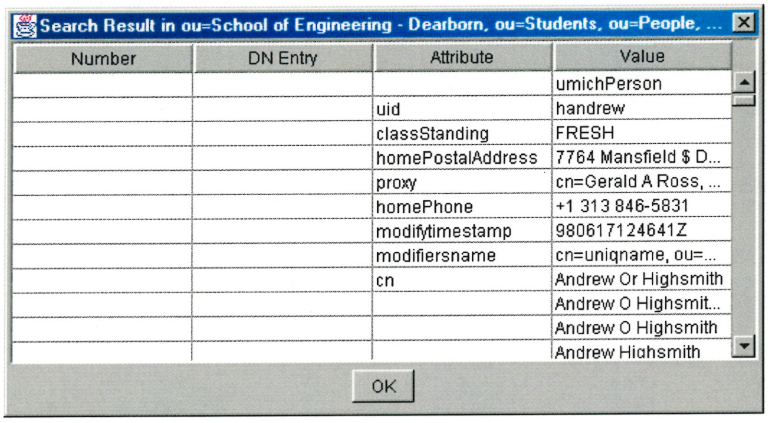


Figure 4.10(b) **Résultat de recherche pour l'attribut classStanding=FRESH**

La figure 4.11 est la fenêtre d'interaction entre le manager et les clients SNMP. Dans cette figure il y a :

- À gauche en haut, les informations permettant de construire les objets locaux et distants ; les boutons pour effectuer ces initialisations et exécuter les opérations de gestion.
- À gauche au milieu, une table pour afficher les résultats des requêtes.
- À gauche en bas, une zone de texte pour afficher les événements d'exécution.
- À droite en haut, un ensemble d'informations pour surveiller un élément géré à temps réel.
- À droite en bas, un graphe pour afficher la courbe de variation d'un objet géré.

Le menu "Monitor" permet d'afficher le résultat des requêtes et la courbe de variation.

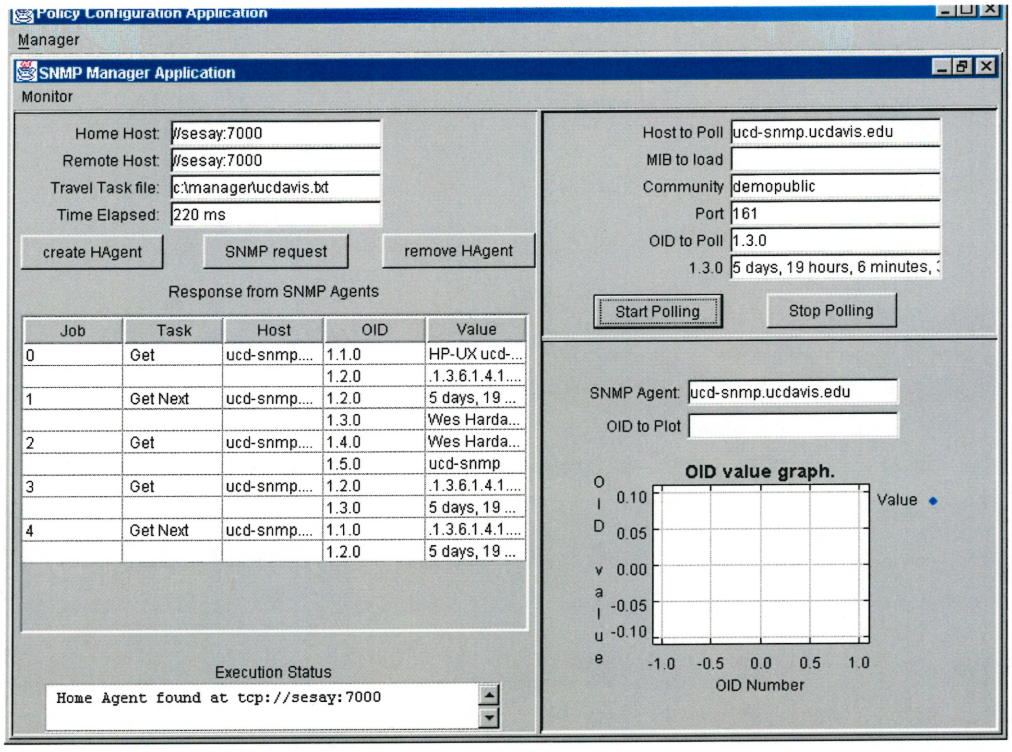


Figure 4.11 Résultat du monitoring sur l'agent SNMP de Univ. Davis en Californie

En conclusion partielle de ce chapitre, l'architecture développée dans cette recherche est composée de plusieurs éléments intéressants. Toutefois, pour évaluer concrètement son utilisation et sa performance, il conviendrait d'implémenter les mécanismes de contrôle, ou importer une implémentation existante (iphighway.com a développé un serveur publique COPS).

Pendant la réalisation de cette architecture, plusieurs éléments du système de gestion de politiques n'étaient pas disponibles et notre temps de recherche est limité. Ce qui a

fait que nous n'avons pas pu expérimenter tout le système de notre architecture. Nous n'avons testé qu'un composant de l'architecture, les applications de gestion.

L'application que nous avons développée dans cette recherche est basée sur la gestion sur L'Internet, i.e., l'utilisateur peut l'exécuter n'importe où sur l'Internet pour gérer ou rechercher les données de gestion. Nos tests de l'application de configuration sur l'agent SNMP de l'université Davis de Californie et les serveurs LDAP de l'université de Michigan et des Pays-Bas, nous ont donné des résultats très satisfaisants.

Chapitre 5

Conclusion

Dans ce mémoire, nous avons une architecture de système de gestion de politiques. De cette architecture, nous avons créé un système de configuration de politiques de gestion pour laquelle nous avons développé une application de configuration de politiques. Pour un rappel, ce système de configuration de politiques avait pour but de :

- Automatiser la reconfiguration des composants. Les services différenciés et prévisibles requièrent cette reconfiguration automatique pour assurer la qualité de service dont les objets gérés ont besoin. Pour ce faire, nous avons choisi le système de gestion basé sur les politiques.

- Simplifier la gestion des systèmes de réseaux larges et complexes, tout en augmentant la performance et l'efficacité du système. Pour ce faire nous suggérons la distribution des tâches des serveurs de stockage de politiques et la distribution des tâches de contrôle de politiques.

- Intégrer la gestion des services à celle des composants du réseau. Nous avons alors implanté deux outils, qui devraient permettre aux administrateurs de configurer les politiques de gestion dans LDAP et de contrôler les objets gérés du réseau.

Nous avons testé l'application de configuration de politiques sur des serveurs publics, pour évaluer son efficacité. L'application fonctionne très bien en lecture sur ces serveurs. Nous l'avons testé sur un serveur (OpenLDAP) [Ldap] que nous avons

installé pour vérifier l'écriture dans le serveur. Dans l'ensemble, elle fonctionne très bien tant que l'utilisateur est connecté légalement (i.e., possède les droits d'accès requis).

Nous avons opté pour la mobilité du manager SNMP pour diminuer le trafic dans le réseau, mais toutes les plates-formes n'acceptent pas les agents mobiles encore. Cela nous a limité dans les tests. Ainsi, dans l'avenir, il serait intéressant de plus développer cette partie.

Aussi, la sécurité utilisée dans les deux outils (client LDAP plain texte et Manager SNMP) n'est pas suffisante pour un réseau d'entreprise où la sécurité est d'importance majeure. Dans l'avenir, il serait intéressant de renforcer ce point aussi.

Notre application est utilisable n'importe où sur l'Internet pour configurer les politiques et contrôler les objets gérés d'un réseau. Il peut permettre aussi de surveiller les serveurs LDAP et les serveurs et clients COPS.

Références

[Ade 98] Adel Ghlamallah and Raouf Boutaba.
Implementing a Distributed Web-based Management System in Java. International Telecommunications Symposium (ITS'98), Aug 98

[Bau 97] M. A. Bauer et al.
Services supporting management of distributed applications and systems. IBM Systems Journal, vol. 36, Number 1, 1997

[Bri 01] S. Brim, B. Carpenter, F. Le Fancheur, D. Black.
INT-DRAFT: Per Hop Behavior Identification Codes, February 2001

[Cay 99] Cay S. Horstmann, Gary Cornell.
Core Java 2 - Vol. I-Fundamentals, Sun Microsystems Press, 1999

[Cha 00] K. Chan, et al.
INT-DRAFT: The Policy Device Auxiliary MIB, July 2000

[Cha 97] Jean-Marie Chauvet.
Corba, ActiveX et Java Beans. Editions Eyrolles 1997

[Coc 97] William R. Cockayne, Michael Zyda.
Mobile Agents, Manning 1997

[CORBA] En ligne : http://www.omg.org

[Chr 01] Christine Tomlinson, et al.
INT-DRAFT: The Java LDAP Application Program Interface, February 2001

[Dav 97] David Perkins, Evan McGinnis.
Understanding SNMP MIBs. Prentice Hall, 1997

[DMTF] En ligne http://www.dmtf.org

[Fra 00] Francis Reichmeyer, et al.
INT-DRAFT: COPS usage for Policy Provisioning (COPS-PR), October 2000

[Ger 95] German Goldszmidt and Yechiam Yemini.
Distributed Management by Delegation. In Proceedings of the 15th International Conference on Distributed Computing Systems, June 1995

[Hei 99] Heinz-Gerd Heigering, Sebastian Abeck, Bernhard Neumair.
Integrated Management of Networked Systems, Morgan Kaufmann, 1999

[Her 99] Shai Herzog, et al.
A Policy Framework for Integrated and
Differentiated Services in Internet. IEEE Network
September/October 99

[Hug 00] Hugh Mahon, Yoram Bernet, Shai Herzog, John Schnizlein.
INT-DRAFT: Requirements for a Policy Management System, November 2000

[JAVA] En ligne: http://java.sun.com/docs/books/tutorial/trailmap.html

[JAVA-MAPI] En ligne: http://www.javasoft.com/products/JavaManagement/

[Jam 97] James Won-Ki Hong et al.
Web-Based Internet Services and Network Management. IEEE Communications, October 1997

[JNDI] En ligne:
http://spectral.mucs.mu.edu/javadev/jndi/tutorial/trailmap.html

[Jür 95] Jürgen Schönwälder et al.
Distributed Network Management Approaches and Problems. University of Twente, 1995

[Lau 97] Laura Lemay, Charles L. Perkins.
Le Programmeur Java 1.1, Simon & Schuster Macmillan,1997

[LDAP] En ligne: http://www.openLDAP.org

[Lei 93] Allan Leinwand, Karen Fang.
Network Management: A Practical Perspective. Unix and Open Systems series, 1993

[Luc 96] Luca Deri.
Network Management for the 90s. Proceeding of ECOOP'96 workshop, July 1996

[Mel 98] Mellquist, Peter Erik.
SNMP++ An Object-Oriented Approach to Developing NETWORK MANAGEMENT APPLICATIONS, Hewlett-

Packard Professional Books, 1998

[McC 01] K. McCloghrie, et al.
INT-DRAFT: Framework Policy Information Base, March 2001

[Moo 00] B. Moore, E. Ellesson, J. Strassner.
INT-DRAFT: Policy Core Information Model, May 2000

[Nic 97] C. Nicolas, C. Avare, F.Najman, M. Castanet.
Java client-serveur. Editions Eyrolles 1997

[Noe 97] Noemie Simoni, Simon Znaty.
Gestion de réseau et de service. InterEditions 1997

[Orf 98] R. Orfali, D. Harkey.
Client/Server Programming with JAVA and CORBA. Second Edition Wiley 1998

[Pet 97] Thèse de doctorat de Petre Dini.
Gestion automatique de la configuration dans les réseaux et systèmes ouverts distribués, DIRO, Université de Montréal 1997

[Pet 95] Petre Dini, Gregor v. Bochmann, Rachida Dssouli. Sur la voie de l'intégration dans la gestion des systèmes ouverts distribués. CFIP'95

[Ptolemy] En ligne http://ptolemy.eecs.berkeley.edu/java/ptplot.html

[RedBook] Heinz, Larry Brown, Franz-Stefa Hinner, Wolfgang Reis, Johan

Westman.
Undestanding LDAP; en ligne: http://www.redbooks.ibm.com

[RFC 1212] K. McCloghrie, M. T. Rose.
RFC 1212: Concise MIB definitions, March 1991

[RFC 1213] K. McCloghrie, M. T. Rose.
RFC 1213: Management Information Base for Network Management of TCP/IP-based Internets: MIB II, March 1991

[RFC 1157] J. Case et al.
RFC-1157: A Simple Network Management Protocol (SNMP), May 1990

[RFC 1155] M. T. Rose, K. McCloghrie.
RFC 1155: Structure and Identification of Management Information for TCP/IP-based Internets, August 1990

[RFC 1441] J. Case, K. McCloghrie, M. Rose, S. Waldbusser.
RFC 1441: Introduction to version 2 of the Internet-standard Network Management Framework (SNMP v2), April 1993

[RFC 1633] R. Braden, D. Clark, S. Shenker.
RFC 1633: Integrated Services in Internet Architecture: an overview, June 1994

[RFC 1757] S. Waldbuser.
Remote Network Monitoring Management Information Base, February 1995

[RFC 1777] W. Yeong et al.

RFC-1777: Lightweight Directory Access Protocol, March 1995

[RFC 1778] T. Howes, S. Kille, W. Yeong, C. Robbins.
RFC 1778: The String Representation of Standard Attribute Syntaxes, March 1995

[RFC 1779] S. Kille
RFC 1779: A String Representation of Distinguished Names, March 1995

[RFC 1831] R. Srinivasan.
RFC 1831: Remote Procedural Call (RPC) Specification Version 2, August 1995

[RFC 2251] M. Wahl, et al.
RFC 2251: Lightweight Directory Access Protocol (v3), December 1997

[RFC 2570] J. Case, R. Mindy, D. Patain, B. Stewart.
Introduction to version 3 of the Internet-standard Network Management Framework (SNMP v3), April 1999

[RFC 2748] Shai Herzog, et al.
RFC-2748: Common Open Policy Service (COPS), January 2000

[RFC 2749] Shai Herzog, et al.
RFC 2749: COPS Usage for RSVP (COPS-RSVP), January 2000

[RFC 2830] J. Hodges, R. Morgan, M. Wahl.
RFC 2830: Lightweight Directory Access Protocol (LDAP v3): Extension for Transport Layer Security, May 2000

[Rum 91] James Rumbaugh et al.
Object-oriented modeling and design, Prentice-Hall, Inc. 1991

[San 00] Luis Sanchez, Keith McCloghrie, Jon Saperia.
INT-DRAFT: Requirements for configuration Management of IP-based networks, July 2000

[Sap 00] J. Saperia.
INT-DRAFT: Policy Configuration with SNMP, July 2000

[Sch 01] J. Schoenwaelder.
INT-DRAFT: SNMP over TCP Transport Mapping, March 2001

[Sid 95] Dr. Sidnie M. Feit.
SNMP A Guide to Network Management. McGraw-Hill, Inc., 1995

[Slo 94] Sloman Morris.
Network and Distributed Systems Management. Addison Wesley, 1994

[Smi 01] A. Smith, et al.
INT-DRAFT: Structure of Policy Provisioning Information, February 2001

[Sni 00] Y. Snir, et al.
INT-DRAFT: Policy Framework QoS Information Model, November 2000

[Sri 99] P. Srisuresh, L.A. Sanchez.
INT-DRAFT: Policy Framework for IP Security, February 1999

[SNMP-API] En ligne: http://www.adventnet.com

[Sta 93] William Stallings.
SNMP, SNMPv2, and CMIP The Practical Guide to Network Management Standards. Addison Wesley, 1993

[Ste 96] Steve Vinoski.
CORBA: Integrated Diverse Applications Within Distributed Heterogeneous Environments, IEEE Communications, vol. 14, No12, Feb. 1997

[Str 01] J. Strassner, et al.
INT-DRAFT: Policy Core LDAP schema, March 2001

[SunMS] En ligne http://www.sun.com

[Tho 97] Thomas J. Mowbray, William A. Ruh.
Inside CORBA: distributed object standards and applications. Addison Wesley 1997

[UCDavis] En ligne : http://www.uc-snmp.ucdavis.edu

[UofMich] En ligne http://www.itd.umich.edu

[VOY] En ligne http://www.objectspace.com

[Wal 00] Walter Weiss, David Durham, Andrea Westerinen.
INT-DRAFT: Network Information Model Requirements, July 2000

[WBM] http://www.microsoft.com/management/