3ème Colloque Francophone sur la Gestion de Réseaux et de Services - GRES'99, Montreal, juin 1999

# Un modèle de surveillance utilisant des agents intelligents*

Karina Marcus
Institut Eurécom
BP 193, 06904 Sophia Antipolis, France
phone + 33 4 93 00 26 55
e-mail marcus@eurecom.fr

2 juin 1999

**Résumé**

Le processus de surveillance d'un réseau est essentiel pour obtenir une vision de la santé du réseau (statuts des éléments) à tout moment, ainsi que pour aider l'administrateur à prendre des décisions de ré-configurations et observer ensuite les changements de comportements des éléments.

Dans cet article nous voulons montrer comment une architecture de gestion de réseaux basée sur des agents intelligents peut implémenter un modèle de monitoring de réseaux général.

**Mots-clés:** Monitoring, agent intelligent, gestion des fautes, gestion distribuée

## 1 Introduction

Le processus de surveillance d'un réseau est essentiel pour obtenir une vision de la santé du réseau (statuts des éléments) à tout moment, ainsi que pour aider l'administrateur à prendre des décisions de ré-configurations et observer ensuite les changements de comportements des éléments. Plusieurs modèles de surveillance ont déjà été proposés [3, 8, 13], et très généralement ces modèles incluent aussi la surveillance de processus qui s'exécutent dans

*La recherche à Eurecom est partiellement financée par ses sponsors industrielles : Ascom, Cegetel, France Telecom, Hitachi, IBM France, Motorola, Swisscom, Texas Instruments, et Thomson CSF.

les systèmes distribués. Dans cet article nous traitons plus spécifiquement la surveillance d'équipements de réseau, fonction fondamentale de la gestion de réseaux.

Dans un premier temps très centralisée, maintenant la gestion de réseaux devient de plus en plus distribuée [6, 7, 17]. Dans cette diréction la communauté scientifique commence à se pencher de plus en plus sur l'utilisation des agents intelligents pour déléguer des tâches qui doivent être effectuées à distance, de façon à réduire le trafic engendré par une administration centralisée et à pouvoir programmer ces agents intelligents d'une façon simple et de haut niveau [1, 12, 15, 17].

Nous voulons montrer comment une architecture de gestion de réseaux basée sur des agents intelligents peut implémenter un modèle de monitoring de réseau général. Cette architecture a été implémentée d'une façon expérimental dans le projet DIANA [2]; ici l'objectif est de montrer qu'une extension de cette première version est possible pour un monitoring plus complèt, tout en s'appuyant sur le modèle de monitoring donné par Sloman dans [8].

Dans la section 2 le modèle de monitoring qui sera utilisé comme référence sera présenté. Ensuite nous donnerons l'architecture de base pour le système d'agents, avec la terminologie qui sera utilisée dans la suite. La section 4 montrera comment le système d'agents peut implémenter le modèle de monitoring donné. On clôtura cet article avec les conclusions obtenues et quelques perspectives pour les travaux futurs.

# 2 Monitoring

Le processus de monitoring peut être défini comme le processus de collecte dynamique, d'interprétation et présentation de l'information concernant des objets ou des processus logiciels sous surveillance [5]. Il peut être utilisé pour des activités de gestion comme la gestion des performances, la gestion des configurations, gestion des fautes, gestion de la sécurité, etc, [14].

Lors d'un monitoring, on observe le comportement du système et les informations sont collectées et ensuite utilisées pour prendre des décisions de gestion.

Le modèle adopté pour le monitoring est donné dans [8] et concerne seulement la monitoring de réseaux et systèmes distribuées basés sur le paradigme des objets. On parle donc des objets gérés, qui possèdent une interface opérationnelle et une interface de gestion (voir figure 2). Ces objets peuvent changer de status, et chaque changement est vu comme un événe-

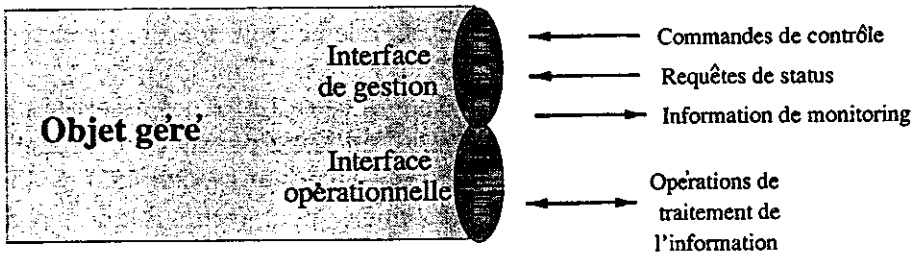


FIG. 1 – *Objet géré*

ment. Les événements peuvent être combinés pour créer d'autres événements qui possèdent une signification plus particulière, spécifique à un contexte.

Ce modèle est basé sur le modèle de gestion d'événements de [4], avec quelques modifications. Les quatre principales activités de ce modèle sont :

1. génération : événements importants sont détectés et il y a la génération de rapports ;
2. traitement : comprend des fonctionnalités comme la combinaison de traces, validation, actualisation des bases de données, corrélation, filtrage, etc ;
3. dissémination : les rapports de monitoring sont distribués aux utilisateurs qui les ont demandés ;
4. présentation : l'information est présentée aux utilisateurs dans une forme appropriée.

En s'appuyant sur ce modèle nous allons montrer comment un système de multi-agents peut être utilisé pour le monitoring. Évidemment on voit que pour que le système de monitoring soit distribué il faut utiliser plusieurs agents, avec une répartition de domaines. La délégation de tâches – principal outil dans un système multi-agents – sert à bien repartir le monitoring sans entraîner un trafic important. On détaillera ce système dans la suite.

# 3 Les agents intelligents

Beaucoup de discussion est menée autour des agents intelligents ce dernier temps. Les propriétés fondamentales des agents intelligents, ainsi que ses principaux composants sont toujours en train d'être redéfinies [10, 11, 16]. On n'essayera pas ici de mettre l'accent sur une ou autre définition, mais on

regardera un agent comme étant tout simplement un logiciel persistent qui agit sur l'environment à la demande et à l'intêret d'un utilisateur.

Un agent maintient une base de *beliefs*, c'est-à-dire des assertions sur la réalité comme l'agent la voit ; les *beliefs* d'un agent peuvent correspondre ou non à la réalité, l'agent lui-même ne pouvant pas le vérifier tout seul.

Dans cet article un agent est autonome, peut communiquer, peut coopérer et déléguer des tâches à d'autres agents. Il est aussi délibératif – possède une capacité de raisonner, et réactive – des action prédéterminées sont associées à l'occurrence d'événements.

Dans le projet DIANA [1] un agent est implémenté à l'aide des deux types de modules : le *brain* et les modules de compétence, qui sont des modules spécialisés et qui peuvent être chargés et déchargés de l'agent sans aucune rupture dans l'exécution du programme de l'agent.

Le *brain* est responsable des activités suivantes :

- maintenir la base de données de l'agent, i.e. des *beliefs* (créateurs, temps de création, les modules abonnés, ...) ;
- de la gestion des modules de compétence, des pré-requis de ces modules, du chargement et déchargement des modules, etc ;
- de l'envoi des *beliefs* aux modules intéressés ;
- de la communication inter-agents, et des informations nécessaires pour l'établissement de la communication (adresses, ports, noms), entre autres.

Un module de compétence est conçu dans l'objectif d'accomplir une fonction de gestion de réseaux spécifique. Un tel module peut être incorporé dynamiquement par l'agent de façon à ce que celui devienne plus "capable". Chaque module de compétence possède une interface qui communique au *brain* les informations d'initialisation ; pendant l'utilisation d'un module, celui-ci peut demander des services qui sont à la charge d'autres modules, et le *brain* est capable de déterminer qui fournit quel type de service. Enfin, toute communication entre le module et le *brain* après l'initialisation du module se passe par les *beliefs*. Un module a à sa disposition les opérations de création, actualisation, élimination de *beliefs*, et il est notifié par le *brain* au moyen des notifications.

Comme principal addition au modèle proposé dans [1], l'agent peut posséder une couche réactive qui sera responsable des services de base programmables, comme des tests de dépassement de seuils, des calculs utilisant des variables collectés, comme moyenne, variance, etc.

# 4 Adaptation du modèle en utilisant les agents intelligents

Dans l'architecture proposée il y a un agent central qui est chargé de l'interface avec l'utilisateur. L'utilisateur crée en dynamique un module de compétence correspondant au monitoring désiré. L'agent central délègue à chaque agent dans le système le monitoring sur son domaine, avec les paramètres demandés par l'utilisateur (comme ensemble d'éléments, intervalle, information à collecter, etc). Le *brain* de l'agent central maintient une table avec le type de *belief* qui intéresse à l'utilisateur en considérant le module de compétence qui l'utilisateur a créé.

## Génération

Chaque agent collecte localement l'information que lui est demandée par l'agent central. Cette information comprend surtout de l'information concernant les statuts des éléments dans le domaine de l'agent, accompagnée de sa source, du moment où l'occurrence de l'événement a été détectée, entre autres.

Le module de monitoring est le responsable d'obtenir l'information directement auprès des éléments du réseau et de produire les données en forme de *beliefs*. Ce module doit pouvoir ordonner les requêtes dans le temps et combiner des demandes pour optimiser les requêtes aux éléments du réseau. Pour l'instant seulement des requêtes à des agents SNMP sont utilisés, mais si un jour des sondes RMON sont installées, plus d'information peut être collectée.

En effet, la partie d'instrumentation peut être augmentée au fur et à mesure que les besoins se font ressentir. Seulement les modules liés directement à la collecte de données et à l'interface avec l'utilisateur pourront être concernés par cette évolution. En effet, l'interface changera seulement dans le cas où l'utilisateur pourrait choisir plus d'informations. Pour la partie de traitement des nouvelles informations, tout se passe comme si des nouveaux *beliefs* soient créées. Si l'utilisateur n'a pas précisé le type d'information à collecter, mais seulement la source, la conséquence sera l'établissement d'un rapport plus complet.

Le monitoring des événement générés directement par les éléments du réseau (SNMP-trap et M-event-report, par exemple) sont traités aussi comme des *beliefs* créées par le module de compétence de monitoring, et peuvent être envoyés à l'utilisateur dans le cas où le module d'interface créé pour le monitoring les demande.

## Traitement

Étant donné que les informations sont collectées à la demande, chaque utilisateur au moyen du module d'interface précise quelle est l'information qui lui intéresse. Ce choix est enregistré par le *brain*, et toute la partie de combinaison et filtrage est faite directement au niveau de *beliefs* par le *brain* de l'agent qui exécute le monitoring local. Chaque utilisateur peut ensuite faire une validation selon sa base de données spécifique et utiliser, archiver ou éliminer l'information.

Un filtrage pourra aussi être fait au niveau de la couche réactive. Cette couche permet à l'utilisateur et aux modules de compétence de faire des manipulations et comparaisons simples entre les informations collectées, comme dépassement de seuil, vérification de la source, temps écoulé, etc.

Comme on ne collecte pas toute l'information disponible, seulement celle qui a été demandée, on ne peut pas combiner toute sorte d'information sans que le module de compétence ne l'ait pas demandée avant. La combinaison doit être faite au niveau du module même, au moment où il possède les informations.

Un filtrage global peut être effectué pour valider les informations obtenues. Un type de validation consiste à vérifier que les agents auxquels des tâches ont été déléguées sont fiables, et que l'on peut donc se fier aux informations renvoyées. Une implémentation de cette vérification peut être retrouvé dans [9].

## Dissémination

Comme déjà discuté, la dissémination est effectuée directement par le *brain* de l'agent locale et central, puisque c'est le *brain* qui est en charge d'acheminer les messages demandés par chaque module de compétence et par la communication inter-agents.

## Présentation des informations

Chaque module de compétence peut créer sa propre interface avec l'utilisateur et choisir le modèle de présentation qui le convient le mieux (diagramme, animation, texte, etc). Ceci va dépendre fortement du langage utilisé pour le conception du système d'agents. Par exemple, en utilisant Java la conception et implémentation d'une interface graphique est assez facile à l'aide de la bibliothèque des fonctions graphiques (awt). La possibilité d'utiliser des applets peut permettre la facilité de consulter cette l'interface de

monitoring depuis un *browser* http à distance. Un exemple d'une tel interface peut être retrouvé dans [1].

# 5 Conclusion

Nous avons présenté dans cet article une manière de concevoir un système de monitoring en utilisant des agents intelligents. La distribution des agents dans le réseau contribue à avoir une répartition de la surveillance, et à ne pas engendrer un trafic important.

La façon dont les tâches de monitoring sont déléguées aux agents conduit aussi à une décentralisation du filtrage, et seulement des information significatives passent dans le réseau vers l'utilisateur.

La couche réactive programmable devra permettre une génération d'événements par l'agent intelligent responsable d'une surveillance local - et cette couche étant programmable par les modules de compétence augmente la puissance des opérations et la décentralisation des calculs.

Pour la suite il y a toujours le défi d'améliorer au plus possible l'interface graphique vis à vis de l'utilisateur. On devra aussi définir quels sont les paramètres importants pour la définition d'un monitoring spécifique.

D'un autre coté on peut remarque que l'établissement d'un historique du monitoring doit pour l'instant être pris en charge par le module de compétence qui représente l'utilisateur vis à vis du *brain*. Est-il nécessaire d'avoir un historique global et une base globale de données de tous les objets avec toute l'information collectée? Pourrait-on concevoir un module de compétence chargé du historique global?

Enfin, l'objectif est de donner toujours plus d'"intelligence" aux agents de façon à ce que l'administrateur utilise la sienne pour concevoir d'un manière simple de nouveaux modules de compétence.

# Références

[1] Cheikhrouhou (M.), Conti (P.), Labetoulle (J.) et Marcus (K.). – Intelligent agents for network management: a fault detection experiment. *In: Proceedings of the 6th IFIP/IEEE International Symposium of Integrated Network Management - IM'99*, pp. 555–566.

[2] Conti (P.), Labetoulle (J.), Marcus (K.) et Cheikhrouhou (M.). – Network management system with intelligent agents. a first step with SLD. *In: HPOVUA'98 Workshop.* – Rennes, FR, 1998.

[3] Dasgupta (P.). – A probe-based monitoring scheme for an object-oriented distributed operating system. *In: ACM Proceedings of the Conference on Object Oriented Programming Systems, Languages and Applications*, pp. 57–66.

[4] Feldkuhn (L.) et Erickson (J.). – Event management as a common functional area of open systems management. *In: Proceedings of IFIP Symposium on Integrated Network Management.* pp. 365–376. – Norh-Holland.

[5] Joyce (J.), Lomow (G.), Slind (K.) et Unger (B.). – Monitoring distributed systems. *ACM Transactions on Computer Systems*, vol. 5, n° 2, may 1987, pp. 121–150.

[6] Katzela (I.), Bouloutas (A.T.) et Calo (S.B.). – Centralized vs distributed fault localization. *In: Integrated Network Management IV*, éd. par Hall (Chapman &), pp. 250–261.

[7] Kooijman (R.). – Divide and conquer in network management using event-driven network area agents. *In: ASCI Conference.* – Netherlands, may 1995.

[8] Mansouri-Samani (M.) et Sloman (M.). – *Network and Distributed Systems Management*, chap. Monitoring Distributed Systems, pp. 303–347. – Addison Wesley, 1994.

[9] Marcus (K.). – *Improving Reliability of Intelligent Agents for Network Management.* – Rapport technique n° 98-045, Institut Eurécom, dec 1998.

[10] Muller (J.P.). – *The Design of Intelligent Agents - A Layered Approach.* – Berlin, Germany, Springer, 1996, *LNAI State-of-the-Art Survey.*

[11] Nwana (Hyacinth S.). – Software agents: An overview. *Knowledge Engineering Review*, vol. 11, n° 2, octobre 1996, pp. 205–244.

[12] Sahai (Akhi.), Morin (C.) et Billiart (S.). – Intelligent agents for a mobile network manager. *In: Intelligent Networks and Intelligence in Networks*, éd. par Gaiti (D.). IFIP, pp. 449–463. – Chapman & Hall.

[13] Shim (Y.C.) et Ramamoorthy (C.V.). – Monitoring and control of distributed systems. *In: Proceeding of First International Conference on Systems Integration.* pp. 672–681. – IEEE Computing Press.

[14] Sloman (M.). – *Distributed Systems Management.* – Research Report n° DOC 87/6, Imperial College, 1987.

[15] Trommer (M.) et Konopka (R.). – Distributed network management with dynamic rule-based managing agents. *In: 5th International Symposium on Integrated Network Management.* – San Diego, may 1997.

[16] Wooldridge (M.), Müller (J.P.) et Tambe (M.). – Agent theory, architectures, and languages: A bibliography. *In: Intelligent Agents II*, éd. par Wooldridge (M.), Müller (J.P.) et Tambe (M.). pp. 408–431. – Springer Verlag.

[17] Yemini (Y.), Goldszmidt (G.) et Yemini (S.). – Network management by delegation. *In: 2nd International Symposium on Integrated Network Management*, pp. 95–107. – Washington, DC, apr 1991.