INTERNATIONAL JOURNAL OF NETWORK MANAGEMENT
Int. J. Network Mgmt 2004; **14**: 351–364 (DOI: 10.1002/nem.532)

Self-aware management of IP networks with QoS guarantees

By Francine Krief[*,†]

Self-aware management allows the network to react and to adapt to changes of situation. In this paper an architecture is described for self-aware management of IP networks offering QoS guarantees. This architecture uses policy-based management and multi-agent systems. The originality of the present approach lies in the intention to give a real autonomy to the components intervening in the chain of services in terms of internal decisions and configuration. Our solution allows the self-configuration, self-provisioning and self-monitoring of service as well as the proactive service level agreement management.

1. Introduction

The need for automating the supervision activity is today a crucial element of the telecommunication networks. It is due to the increasing need for quality of service (QoS) by users as well as the growing complexity of management functions which allow its assurance and its provisioning, the difficulty of finding people qualified enough to control this complexity, and the need for cost control related to this activity. Indeed, in the current market situation, the operators seek means of reducing the investments and the operational expenditure. In this context, the reduction in human intervention in provisioning and network management asks for an ever-increasing level of automation of the network processes (the control plan) and of the management systems (the management plan). Particularly, the introduction of such a level of automation reduces greatly the tasks of the operator in the services provisioning and assurance. This tendency to create a network forever more autonomous and service-oriented is called, in the marketing language, a 'Self-Aware Network'. This concept implies the development of a data network infrastructure by the functional extension of the control plan. The level of autonomy required is reached by introducing the operational objectives and the parameters to be followed in the infrastructure, as well as by providing respective monitoring and adaptation means. The need for a management

Francine Krief is an Associate Professor in the LIPN laboratory at the University of Paris XIII. Her areas of research are network management and the ambient internet.

**Correspondence to: Francine Krief, LIPN Laboratory, University of Paris XIII, Avenue Jean-Baptiste Clément 99, 93430 Villetaneuse, France.*
[†]E-mail: krief@lipn.univ-paris13.fr

 Published online 9 July 2004

system decreases, and the operator does not need to apply corrections and adaptations so much anymore. Thus, the management system is simplified and even more oriented towards the definition of policies and operational parameters.

In this paper an architecture is described for the self-aware management of IP networks offering QoS guarantees. This architecture uses policy-based management and multi-agent systems. Section 2 gives a definition of the self-aware management. Section 3 briefly describes the policy-based management of IP networks offering quality of service guarantees. Section 4 presents multi-agent systems and their use in policy-based management. Section 5 describes the architecture we propose. Section 6 presents different functions of self-aware management. Section 7 concludes and presents future works.

2. Self-aware Management

Self-aware management can be described as the capabilities of the management processes and the subjacent infrastructure to organize themselves and operate without external assistance. The role of the administrator is limited to the guidance of these processes in their laying down operational objectives and parameters. To offer this self-aware management, it is crucial to take into account the dynamicity of the components to be managed. The following four functions are often quoted.[1]

- Self-configuration. The automatic configuration of the components and the systems pursues high-level policies
- Self-optimization. The components and the systems permanently seek to improve their performance and their effectiveness
- Self-healing. The system is able to detect, diagnose, and to repair the hardware and software problems
- Self-protection. The system is protected from the attacks or the cascades of failure. The system failures are anticipated and alarms are generated

According to this vision, an autonomic system (a 'self-managed' system) is composed of the autonomic elements (the 'self-managed' elements) having resources and providing services to the human actors or to other automatic elements. Each one of these elements is responsible for the management of its state/behavior and its interactions with the environment. The self-managed elements can be seen as agents.[2] The concepts of self-aware management apply to a broad variety of network and service technologies as well as in the field of telecommunications in the computer networks. In this paper, we are interested more particularly in IP networks offering the differentiated services.

According to this vision, an autonomic system (a 'self-managed' system) is composed of the autonomic elements (the 'self-managed' elements) having resources and providing services to the human actors or to other automatic elements.

3. Policy-based QoS Management

In self-aware management it is necessary to separate information concerning control from the resources and information on their state. Policy-Based Management (PBM) suggests such a separation and permits the operator to establish service objectives and policies that are interpreted thereafter by the network resources.[3] Thus, the decision on the resources allocation and configuration can be taken locally in an autonomous way.

The Policy-based management, defined by the Internet Engineering Task Force (IETF), proposes an infrastructure to manage IP networks offering service guarantees. This infrastructure allows a flexible behavior of the network according to the various events which can occur during its management by using the concept of policy. The policies can be defined like sets of rules which are applied to the management and the control of the access to the network resources. They also allow network administrators or service providers to influence the network element behavior according to certain criteria such as the user's identity or the type of the application, the traffic required, etc. A policy can be defined at different levels. The highest level corresponds to the business level. This policy must then be translated into a network

level policy then into a low-level policy which is understandable by the network element.

The IETF, jointly with the Distributed Management Task Force (DMTF), works on the information models related to the various policy levels. The Policy Core Information Model (PCIM),[4] is an extension of the Core Information Model (CIM) of the DMTF. The network is seen as a states machine where the policies are used to control the state changes. It must be able to identify and model the states in progress and to define the possible transitions starting from the policy rules. This model defines the roles, the priorities and the execution orders, but stays in a quite abstract form concerning objects. Concerning QoS, two extension levels were defined: the QoS Policy Information Model (QPIM)[5] which defines precise actions to be realized on the packages and the QoS Device Datapath Information Model (QDDIM)[6] which defines the actions to be undertaken on the network element. Within the policy-based management framework, the significant issue to allow the automation of the management process is automatic translation, on the one hand, of the service contract (SLS—Service Level Specification) into policy rules and, on the other hand, of the high-level policy rules obtained into policies which are understandable by the network element. This process is more difficult to implement within a multi-domain framework because operators (intra-domain service providers) have to dynamically establish inter-domain service contracts for the end-to-end service delivery.

4. Agent Approach

The introduction of multi-agent systems (MAS) into the network constitutes a promising approach.[7] Agent approach allows flexible systems to be built following complex and sophisticated behaviors because it is a combination of extremely modular components. The intelligent components (the agents) and their interaction capacities form a multi-agent system. In general, an agent can be seen as a software element which is responsible for the execution of a process part. It contains a certain level of cognition, extending from simple predefined rules to the learning machines of the artificial intelligence. It acts typically on behalf of a user or of a process allowing automation of the task. In general, the term 'intelligent agent' extends from the adaptive user interfaces called interface agents, to the intelligent processes co-operating between them to carry out a common task.[2] These agents are able to bring their expertise to solve the faults and to control the network infrastructures. Their properties, in particular of autonomy, adaptation and distribution, answer well the problems of management of strongly distributed subnetworks. They allow automatic control and management process and offer services better adapted to the user's needs. The network becomes 'smart ', i.e. it is able to adapt to a new situation, to control the delicate states and to manage the services under conditions not considered *a priori*.[8]

The agent approach in policy-based management environments mainly concerned the introduction of mobile agents, particularly for the dynamic service contract negotiation and policy rules provisioning. A mobile agent is defined, usually, as being an independent program which acts on behalf of a user or another entity and is able to move from a network node to another.[9–11] In Reference 12 mobile agents are used to automate the negotiation between customers and service providers and between service providers in order to provide the service required by the customer. The interest of such an approach lies in the negotiation process delegation at intelligent agents and the reduction of the load of the communications going through the network compared to the results of a traditional protocol (i.e. client/server type). The agent transports business policies so that the negotiation and the decision can be carried out locally.[13] The properties of the agent which are important, here, are mainly the mobility and the capacity to negotiate. In Reference 14 mobile agents are used for the policy rules provisioning in order to discharge the PDP from this task.

Intelligent agents were also introduced into policy-based management environments, particularly, to manage the security. Reference 15 proposes a model of policy-based management of security which uses a multi-agent system. The security management on the network is divided into three plans: a user plan, an intelligent plan and a network plan. In the user plan, the administrator defines and specifies the security policies to be applied in the network. The security policies specify the attacks which the system must detect and guide the agents behavior. The intelligent plan symbolizes the intelligent part of the system. This

plan detects the diagrams of security attack. It is represented by a multi-agent system and an information model used by the MAS which is based on BDI concepts (Believe, Desire, Intention).[16] The network plan represents the network. This plan collects the events related to security which occur in the network and which will be analyzed by the MAS to identify new attacks. The properties of the agent which are required here are mainly the adaptability, the co-operation and the reasoning capacity.

Their properties, in particular of autonomy, adaptation and distribution, answer well the problems of management of strongly distributed sub-networks.

5. Architecture for Self-aware Management

We propose an architecture for the self-aware management of IP networks by using policy-based management and multi-agent systems. This architecture allows the dynamic QoS management within the framework of the DiffServ model. It is also in conformity with the architecture of the IST CADENUS (Creation And Deployment of ENd-User Services in premium IP networks) project. This European project proposed a Service Level Agreement (SLA)-based framework for the configuration and the provisioning of end-user services in Premium IP.

The SLA allows a customer who subscribes to a service from a provider, to have a service level guaranteed by it. The customer can be a user, another provider offering the same level of services (a horizontal SLA) or a service provider whose offer is at a different level (a vertical SLA).[17] The contractual elements of the SLA specify the service that must be delivered. The SLS (Service Level Specification) corresponds to the technical part of the SLA. It contains the services description in technical terms.

The CADENUS project recommends the use of three levels of mediation for telecommunication services: the Access Mediator, the Service Mediator and the Resources Mediator.[18]

Our architecture includes these three mediation components and monitoring functions are introduced to allow each level to adapt its behavior to the environment which it is controlling. Each level can be seen as a self-managed entity, so the provider's tasks in provisioning and services assurance can be reduced. The level of autonomy required is reached by introducing the operational objectives and the parameters to be followed in the infrastructure, as well as by providing respective monitoring and adaptation means. The need to use the management system decreases, and the operator does not need to apply corrections and adaptations so much any more. Thus, the management system is simplified and even more oriented towards the definition of policies and operational parameters.

Each level implements their own tools of monitoring and have a meta-control level which allows it to adapt its behavior to the dynamicity of the environment it is managing. The meta-control level contains two categories of agents:

- The monitoring agent. It controls the coherence of network/network element behavior with the policies which were applied. It makes the decision to inform the others agents or the higher level before a SLA violation
- The adaptation agent. It modifies the Mediator/Network Element behavior in order to improve its operation and to optimize the service configuration.

Each level is detailed in the following sub-sections.

—5.1. The Access Mediator—

The Access Mediator (AM) is responsible for the mediating between the end-user and several service providers. It has knowledge of the end-users, the access link and the terminal type and gives them access to a particular service provider. The Access Mediator presents to the user a wider selection of services, ensuring the lowest cost and it simplifies the process of service selection. It can immediately notify the user that a new service has become available.

In Reference 19, we proposed to use mobile agents for the dynamic negotiation of SLA between a customer and several AMs. An agent

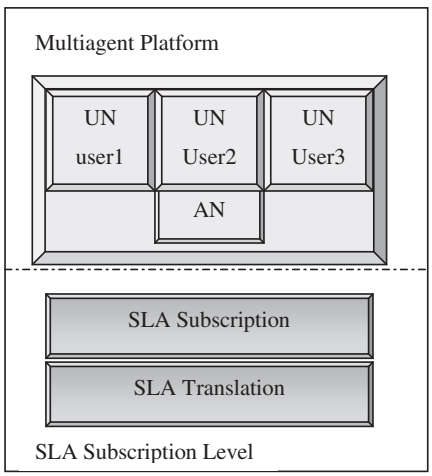

Figure 1. Access Mediator

called UO (User Overseer), located on the user's terminal, sends mobile agents, called UN (User Negotiator) to the AMs to negotiate a service according to the user's needs. They send to the UO the result of their negotiation as well as the new offers that may interest the user. The UO selects on behalf of the user the best offer among the offers of service it received.

The AM has a multiagent platform on its site that welcomes UN agents from customers. An agent called AN (Access Negotiator) negotiates services and rates on the behalf of the AMs. The interactions between agents are described in Reference 19. Figure 1 represents the Access Mediator. The AM contains a multiagent platform and two modules: SLA Subscription and SLA Translation.

SLA Subscription binds the customer to a particular service provider through a SLA which allows the customer to specify the QoS required. It also allows the service provider to identify precisely the needs of the customer so that it can manage the service as well as possible.

SLA Translation translates the new service request into an XML format and sends it to the Service Mediator concerned.

We proposed to use mobile agents for the dynamic negotiation of SLA between a customer and several Access Mediators.

—5.2. The Service Mediator—

The Service Mediator (SM) has to inform the AMs of all new service offers, so that they can present these new offers to their users. It supervises the management of the physical access to the services via the appropriate underlying network(s), using the Resource Mediator(s) concerned. The SM maintains no direct contact with the end-users for SLAs. It possibly deals with other service providers to compose its services and with network providers to support its services.

In the following we assume a Hub Business Model,[18] in which a service provider may contact several network providers to fulfill an end-to-end service. In this case, it has knowledge of every service offer. The SM controls and coordinates all the Resource Mediator(s) under its responsibility. It is responsible for the translation and the control of the SLA, and maintains a service report for the Access Mediator. It is responsible for the contractual aspect of the services which must be provided to the customers. Figure 2 represents the Service Mediator. It contains two parts: the Service Monitoring and Management Level, and the Service Meta-Control Level.

The Service Monitoring and Management Level consists of a certain number of modules ensuring each one a particular functionality.

The ***SLA subscription*** module converts the SLAs into corresponding technical parameters. These technical parameters are:[18,20]

- Scope (e.g. network ingress and egress points for the traffic flow)
- Flow identification (e.g. source/destination network addresses)
- Load descriptor (e.g. mean rate, burst size, peak rate)
- Excess treatment (e.g. marking, shaping, dropping)
- Performance guarantees (e.g. delay, packet loss, jitter, throughput)
- Service schedule
- Monitoring parameters (e.g. monitoring frequency)
- Reliability
- Cost parameters

Then, this technical description is used by the Service Configuration module and the Service Monitoring module.

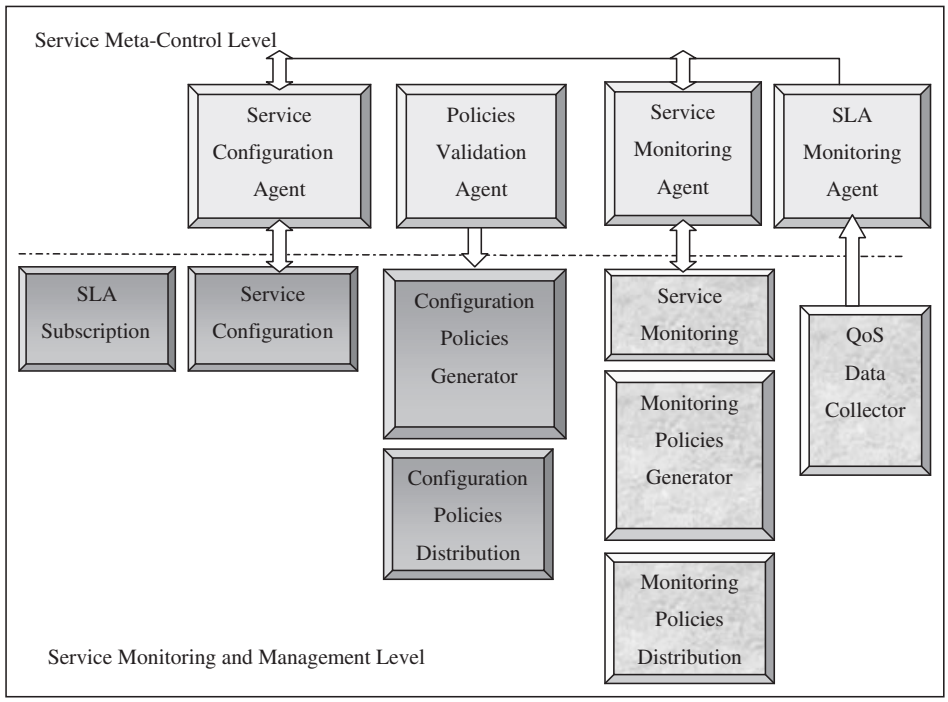

Figure 2. Service Mediator

The ***Service Configuration*** module has to identify every Resource Mediator involved in providing the end-to-end service and transmit an SLS to each one of them (hub model). To compose the different SLSs, it has different data bases (e.g. Service Description) and circumstantial information which is periodically updated (e.g. effective QoS parameter values for each service provided by a Resource Mediator, and satisfaction level associated with each service).

The ***Configuration Policies Generator*** deals with the translation of the SLS in policy rules which are stored after validation in a policy repository. These policies follow the QPIM model and so are independent of the technology used. We use the formalism defined in Reference 21 to represent these policies. What follows is the classic example of policy rule used at this level:

Policy	*ServiceConfiguration*
For	*ResourceMediator1*
On	*SourceIPaddress*
On Event	*{PolicyTimePeriodCondition}*
Do	*{GoldService}*
Constraint	*Boolean Expression*

The ***Configuration Policies Distribution*** sends the policies to the different Resource Mediators concerned.

The ***Service Monitoring*** has to define the objectives of service monitoring for each Resource Mediator involved in the provisioning of the end-to-end service. To do that, it uses monitoring parameter referring to each SLA and circumstantial information which is periodically updated (e.g. recommended QoS monitoring parameter values for each service provided by a RM). It also knows the RM impacted in the provisioning of the end-to-end service.

The ***Monitoring Policies Generator*** deals with the translation of service monitoring objectives into service monitoring policy rules. The ***Monitoring Policies Distribution*** sends these policies to the RMs concerned. These rules are used to configure the network, so that data collection and reporting depend on the SLA. This is an example of policy rule:

Policy	*ServiceMonitoring*
For	*ResourceMediator1*
On	*SourceIPaddress*

On Event *{PolicyTimePeriodCondition}*
Do *{PacketLossMonitoring}*
Constraint *Boolean Expression*

The ***QoS Data Collector*** collects service information from the RMs and makes a first data processing.

The Service Meta-Control Level is used to adapt the SM behavior to a new situation. There are two agents for the service control and the optimization of the service configuration (Service Configuration Agent and Policies Validation Agent) and two agents for the QoS assurance (Service Monitoring Agent and SLA Monitoring Agent).

The ***SLA Monitoring Agent*** analyzes the data obtained. It makes the decision to contact the Service Configuration Agent and the Service Monitoring Agent or to alert the Access Mediator of the impossibility of ensuring the service required in order, for example, to renegotiate the SLA with the customer. It also informs the Control Admission module which is used for the negotiation with the AM. The negotiation between AM and SM is not considered in this paper. The behavior of this agent is guided by policy rules such as:

Policy *SLAMonitoring*
For *SLAMonitoringAgent*
On *SourceIPaddress*
On Event *{QoSMeasure}*
Do *{alert}*
Constraint ThresholdValue

Only the SLA Monitoring Agent is concerned by this policy rule whose formalism is similar to the ones used until now.

The threshold values are significant here. The SLA Monitoring Agent creates a base of examples from the results obtained following the actions it has done. From this base, a learning method would be used to improve the threshold values, and so, its own behavior.

The ***Service Monitoring Agent*** has to adapt the task of the Service Monitoring module in order to avoid any violation of SLA. It modifies its behavior by proposing a modification of SLS monitoring. The behavior of this agent is also guided by policies such as:

Policy *NewMonitoringSLS*
For *ServiceMonitoringAgent*
On *SourceIPaddress*
On Event *{alertSLA}*
Do *{NewMonitoringSLSProposition}*
Constraint *Boolean Expression*

The ***Service Configuration Agent*** has to adapt or optimize the task of the Service Configuration module by changing its behavior (by modifying the value of a parameter such as the satisfaction level for a service or by proposing a SLA modification which will have as a result the modification of a policy rule or a set of policy rules).

Its behavior is guided by policies of the following type:

Policy *NewSLS*
For *ServiceConfiguration*
On *SourceIPaddress*
On Event *{alertSLA}*
Do *{NewConfigurationSLSProposition}*
Constraint *Boolean Expression*

The ***Policies Validation Agent*** has to validate the policy rules translated by the Translation module before their deployment over the network through the Resource Mediator(s).

Then, the policy rules are stored in the policy repository. The agent's behavior is guided by meta-policies, i.e., policies about management policies. In Reference 22 examples of meta-policies are given.

The Service Meta-Control Level is used to adapt the SM behavior to a new situation.

—5.3. The Resource Mediator—

The Resource Mediator (RM) is associated with the underlying network. It is responsible for the network performance demanded by the service provider(s). In a policy-based management environment, it acts like a PDP (Policy Decision Point). It has the responsibility for determining which policy rules need to be applied to the network elements to satisfy the Service Mediator(s). Specificity of PDPs in our architecture is to send network-

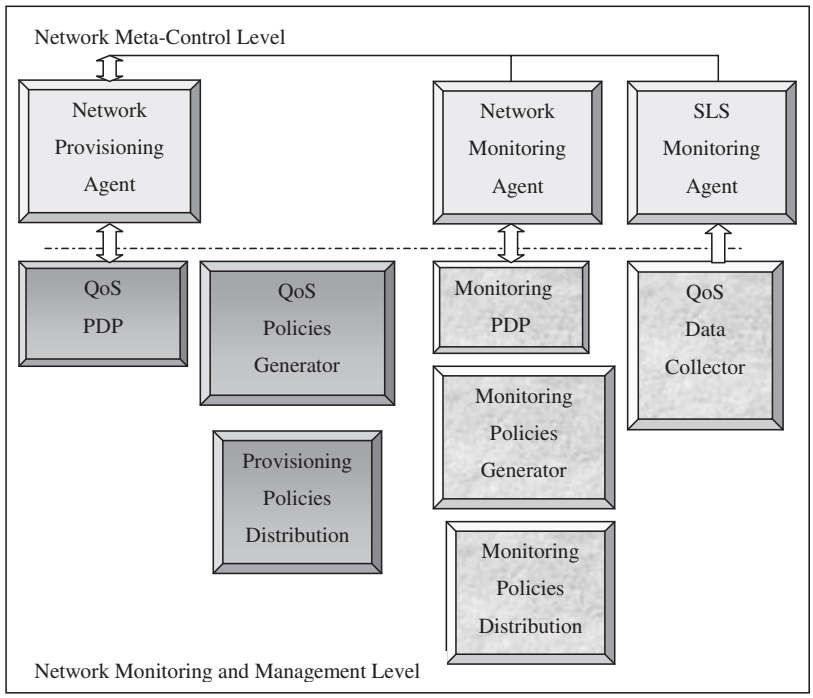

Figure 3. Resource Mediator

level policies which are not directly executable by the network elements in order to give them more autonomy. Policy rules are of the following type:

Policy *ServiceConfiguration*
For Edge*Router1*
On *SourceIPaddress*
Do *{PHBtype}*

Reference 23 gives examples of security network-level policy rules for dissemination to the network elements.

Figure 3 represents the Resource Mediator. It is divided into two parts like the SM.

The Network Monitoring and Management Level includes a ***QoS PDP*** for the provisioning of QoS policy rules and a ***Monitoring PDP*** for the provisioning of QoS monitoring policy rules.

The ***Policies Generator*** modules generate the policy rules in XML format for dissemination to the networks elements. Then, these rules are sent by the distribution modules to the network elements concerned. The functionality of the Network Meta-Control Level is to adapt the RM behavior to the dynamics. It is based on an agent for the control and the optimization of the resource management (Network Provisioning Agent) and two agents for the monitoring (Network Monitoring Agent and SLS Monitoring Agent).

The ***SLS Monitoring Agent*** analyzes the data collected by the **QoS Data Collector**. It makes the decision to contact the Network Provisioning Agent, the Network Monitoring Agent or the Negotiation module. The Negotiation module has to renegotiate the SLS parameters with the Service Mediator(s) if the RM is not able to maintain the QoS required. It is not represented on Figure 3 because RM–SM negotiation is beyond the focus of this paper. The behavior of the SLS Monitoring Agent is guided by policy rules which are similar to the ones used by the SLA Monitoring (SM level).

The ***Network Monitoring Agent*** has to adapt the task of the Monitoring PDP by modifying its behavior, so that the service level required is maintained. The agent's behavior is guided by policies

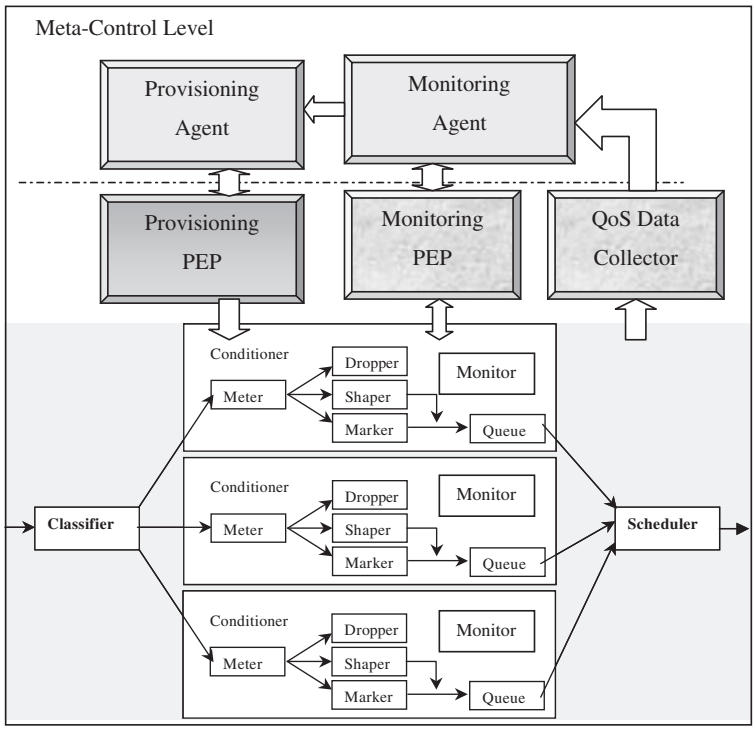

Figure 4. Example of network element (edge router)

which are similar to those used by the Service Monitoring (SM level).

The ***Network Provisioning Agent*** has to improve the Provisioning PDP work in order to respect the service level required by the SM(s). Its behavior is guided by policies which are similar to those used by the Service configuration (SM level). This level is similar to the Service Meta-Control Level: instead of controlling the end-to-end service according to the SLA negotiated with the AM, this level controls the network service according to the SLS negotiated with the RM. The main difference lies in the fact that there is no Policy Validation Agent because the policy rules follow on from policy rules which have been validated by the SM.

—5.4. The Network Elements—

Each network element has a local PDP (LPDP) and a PEP (Policy Enforcement Point). The PEP constitutes the application point of the policies. Within the framework of the DiffServ model, it configures the tools it has to help enforce the decisions: packets filtering, bandwidth reservation, traffic priority, etc.

The local PDP receives the decisions and the policy rules from the RM and translates these policy rules into policy rules/commands understandable by the PEP. To do that, it has an information base that contains the different policy rules (actions) to be executed according to the decisions received from the RM and its perception of its environment. The LPDP can be seen as a rational agent which allows the network element to manage itself. Figure 4 represents a network element. It contains two modules for the enforcement of the policy rules:

- The ***Provisioning PEP*** for the enforcement of provisioning policy rules
- The ***Monitoring PEP*** for the configuration of the monitoring tools.

Each network element includes a Meta-Control Level composed of two agents: the Provisioning Agent and the Monitoring Agent.

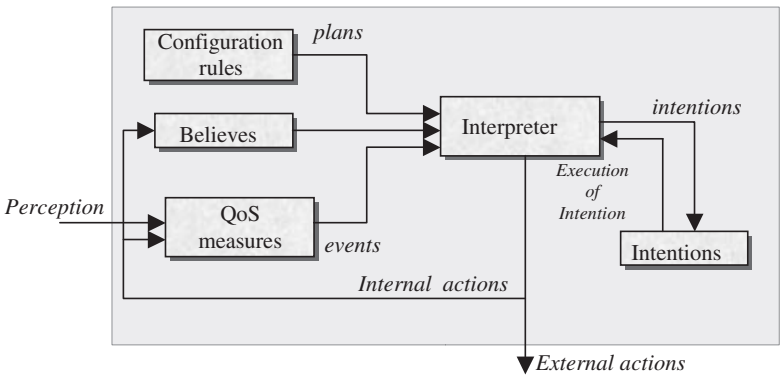

Figure 5. Provisioning agent

The ***Provisioning Agent*** can be seen as a local PDP. Depending on the network state and the policy rules sent by the RM, it pushes new configuration rules to the PEP (it executes a new plan). This allows a dynamic reallocation and management of network resources based on current network state and applications QoS requirements. It could be seen as a rational BDI agent (Figure 5). Its rationality is turned towards the execution of a set of plans (a set of configuration rules) to maintain a certain QoS.

The ***Monitoring Agent*** makes the decision to inform the LPDP, the RM or the neighbor Monitoring Agents about a risk of service degradation.

6. Examples of Self-management

Our solution allows the self-configuration, self-provisioning and self-monitoring of service as well as the proactive SLA management. In the following, different functions of self-management are described.

—6.1. Self-negotiation and Self-configuration—

The introduction of a multiagent platform in the AM allows the automatic SLA negotiation between a client and a service provider. The use of mobile agents has the advantage of limiting the cost of communication and of permitting asynchronous communications while offering a certain confidentiality level. Moreover the introduction of negotiation protocols is the only acceptable method for reaching an agreement between an AM and a user.[19] This negotiation method would be extended to the negotiation of inter-domain SLS (e.g. between RMs in a cascade model[18]).

The capability of SLA negotiation and the automation of the translation of the SLA into policy rules at different levels allow the self-configuration of end-to-end services in IP networks offering QoS guarantees.

The CADENUS architecture associated with the policy-based management was implemented in the context of French national research project ARCADE (Architecture de Contrôle Adaptative des Environnements IP) (http://www.telecom.gouv.fr/rnrt/). The purpose of this project was to set a general model for the control of IP networks. The implementation of this architecture allowed the network elements to be configured dynamically by taking into account the QoS and security needs of the customers. Figure 6 represents the ARCADE middleware.

In this context, we implemented an intelligent interface which identified the QoS needs of the application and the user and dynamically negotiated the SLS with the service provider via the SLS PDP.[24] This intelligent interface was implemented on the multi-agent platform called oRis (http://www.enib.fr/~harrouet/oris.html) and successfully tested by activating different applications with real time constraints such as the VIC (Video Conferencing Tool)

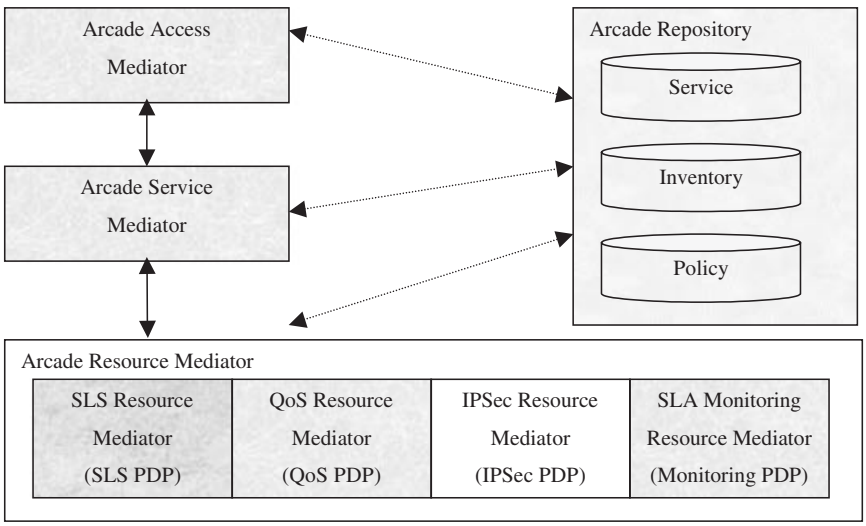

Figure 6. Arcade Middleware

(http://www-nrg.ee.lbl.gov/vic/) and the RAT (Robust Audio Tool) (http://www-mice.cs.ucl.ac.uk/multimedia/software/rat/) applications. The intelligent interface negotiated successfully the SLS parameters of these applications according to the users' needs. The network adapted dynamically the configuration of the network elements, so that the user's demand is satisfied.

The capability of SLA negotiation and the automation of the translation of the SLA into policy rules at different levels allow the self-configuration of end-to-end services in IP networks offering QoS guarantees.

—6.2. Self-monitoring and Self-adaptation—

We have extended the CADENUS architecture to the self-configuration of SLA monitoring tools. This new functionality allows the monitoring to adapt to a change of SLA (e.g. new QoS needs), of topology (e.g. new components involved), of network behavior (e.g. the data collected become insufficient to analyze the change of behavior of the network). Moreover, the introduction of monitoring functions associated with a meta-control layer at every level of the architecture allows the proactive management of the SLA.

The meta-control level is used to modify the entries of the monitoring and management modules in order to respect the SLA/SLS. Figure 7 represents the interactions between agents in order to adapt the system to the environment that it is controlling and to maintain the SLA/SLS required.

It is preferred to introduce agents only into the meta-control level because it is supposed that monitoring and management modules are already present on the market.[18] However, reactive agents could replace these modules. We have also privileged the introduction of MAS only in the meta-control level to facilitate the introduction of agent technology in the telecommunication architectures.

Concerning policy-based management, a local PDP has been introduced in the network elements to permit them to become relatively autonomous. Policies are distributed to all the routers that are concerned and every one of them according to its context will seek the good parameters to configure its part of the service.

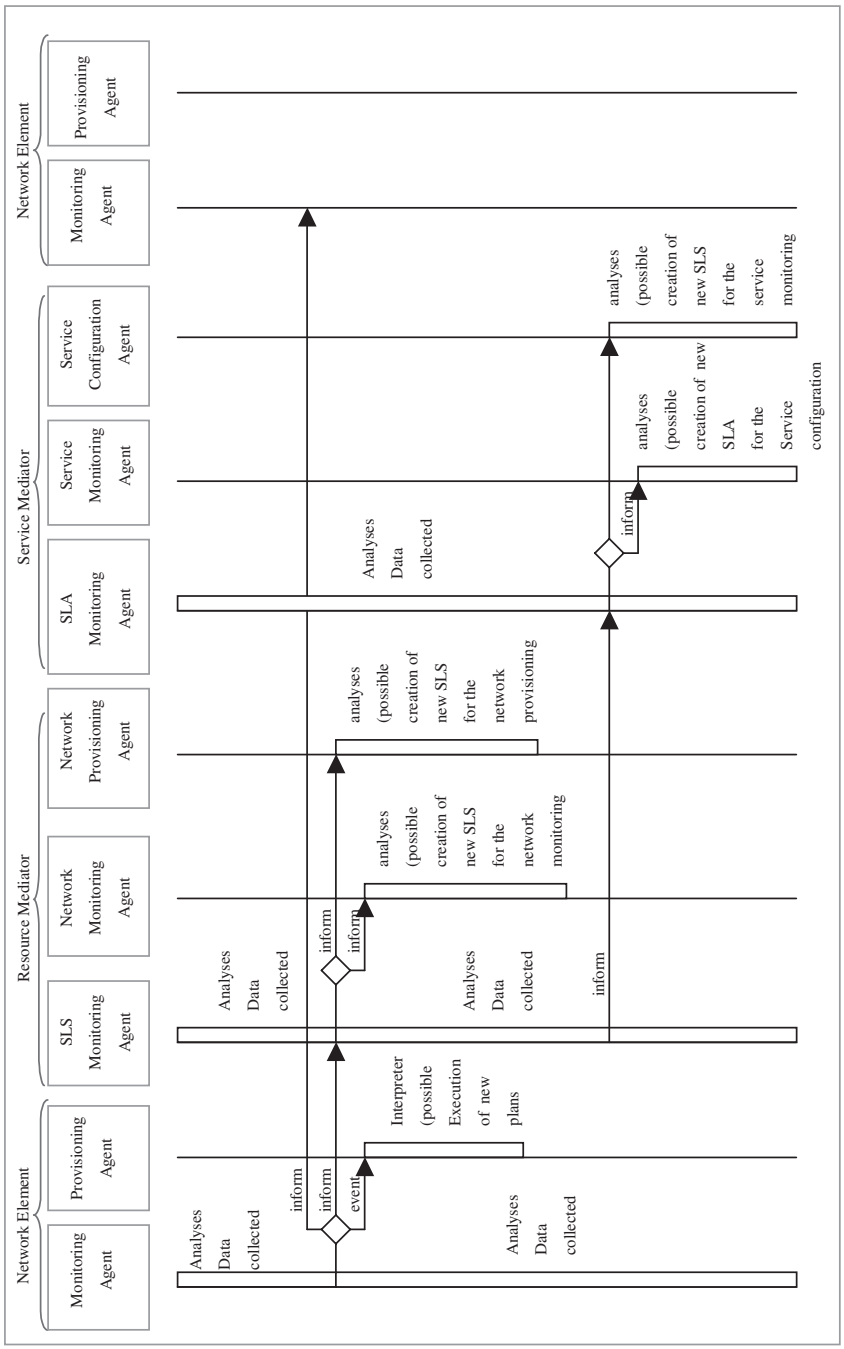

Figure 7. Example of interactions between agents

7. Conclusion and Future Works

An architecture for the self-aware management of IP networks with QoS guarantees has been described. The originality of the present approach lies in the intention to give a real autonomy to the components intervening in the chain of services in terms of internal decision and configuration. Our solution allows the self-configuration, self-provisioning and self-monitoring of service as well as the proactive SLA management.

Our future works concern the description of learning methods which would be used in our architecture. Some works already exist concerning the association of MAS with learning methods.[25–27]

A dynamic negotiation of SLA between a user and an AM has been introduced and we would like to extend this approach to the negotiation between RMs in a cascade model.[18]

Finally, in the context of the French national research project SWAN (Self-aWAre maNagement) (http://www.telecom.gouv.fr/rnrt/), some functionalities of this architecture will be implemented.

References

1. Kephart JO, Chess DM. *The Vision of Autonomic Computing*. IEEE Computer Society, January 2003.
2. Jennings NR. On agent-based software engineering. *Artificial Intelligence*. 2000;**177**(2):277–296.
3. Yavatkar R, Pendarakis D, Guerin R. *A Framework for Policy Based Admission Control*. RFC 2753, 2000.
4. Moore B, Ellesson E, Strassner J, Westerinen A. *Policy Core Information Model*. RFC 3060, 2001.
5. Snir Y, Ramberg Y, Strassner J, Cohen R, Moore B. *Policy QoS Information Model*. Internet Draft, November 2001.
6. Moore B, Durham D, Strassner J, Westerinen A, Weiss W, Halpern J. *Information Model for Describing Network Device QoS Datapath Mechanisms*. Internet draft, May 2002.
7. Gerhard W. *Multi-agent Systems: A Modern Approach to Distributed Intelligence*. MIT Press; Massachusetts, 1999.
8. Boukhatem N, Campedel B, Chaouchi H, Guyot V, Krief F, Nguyen TMT, Pujolle G. *I3—A New Intelligent Generation for Internet Networks*. Conference on Intelligence in Networks. Smartnet'2002, IFIP WG 6.7, Kluwer, Saariselkä, Finland, April 2002.
9. Feridun M, Kasteleijn W, Krause J. Distributed management with mobile components. *IEEE IM'99*, October 1999.
10. Krause S, Magedanz T. Mobile service agents enabling intelligence on demand in telecommunications. *GLOBECOM'96*, 1996.
11. Breugst M, Hagen L, Magdanz T. Impacts of mobile agent technology on mobile communications system evolution. *IEEE Personal Communication Magazine*, August 1998, pp 56–69.
12. Agoulmine N, Fonseca M, Marshall A. Multi domain policy based management using mobile agent. *Lecture Notes in Computer Science*, 2000, pp 235–244.
13. Fonseca M. Policy and SLA based architectures for the management and control of emerging multi-domains networks and services. Ph.D. (University of Paris 6, France), September 2003.
14. Boukhatem N, Bakour H. Distributed policy decision enforcement based on active networks. *Opensig'2001*, London, UK, 2001.
15. Boudaoud K, Guessoum Z, McCathieNevile C, Dubois P. Policy-based security management using a multi-agent system. *HPOVUA'2001*, Berlin, Germany, June 2001.
16. Rao AS, Georgeff MP. *Modelling Agents within a BDI—Architecture*. Fikes R, Sandewall E (eds), Morgan Kaufmann 1992.
17. Marilly E, Martinot O, Bergé-Brezetz S. Service level agreement management. *GRES'01*. Marrakech, Morocco, 2001.
18. Brajeul A, Dugeon O, Potts M. Project Recommendations. IST-1999-11017, Deliverable D6.3, June 2003.
19. Klein G, Krief F. Mobile agents for Dynamic SLA Negotiation. *International Workshop on Mobile Agents for Telecommunication Applications*. MATA'2003. Lecture Notes on Computer Science, Springer, Marrakech, Morocco, October 2003.
20. Goderisetal D. Service Level Specification Semantics and Parameters. Draft-tequila-sls-00-txt. November 2000.
21. Radisic I. *Using Policy-based Concepts to Provide Service Oriented Accounting Management*. NOMS. IEEE Publishing, Florence, Italy, April 2002.
22. Lupu EC, Sloman M. Conflicts in policy-based distributed systems management. *IEEE Transactions on Software Engineering* 1999;**25**(6):852–869.
23. Gay V, Duflos S, Kervella B, Diaz G, Horlait E. Policy-based quality of service and security management for multimedia services on IP networks in RTIPA project. *MMNS 2002*, Chicago, 2002.
24. Jrad Z, Krief F, Benmammar B. An intelligent user interface for the dynamic negotiation of QoS. *ICT'2003*, Tahiti, France, 2003.
25. Agüera S, Guerra A, Martinez M. Memory based reasoning and agents adaptative behavior. *MICAI 2000*, Acapulco, Mexico, April 2000, LNCS vol. 1793. Springer Verlag.

26. Olivia C, Chang CF, Enguix CF, Ghose AK. Case-based BDI agents: an effective approach for intelligent search on the World Wide Web. In *AAAI Symposium on Intelligent Agents*, Stanford, CA, USA, 1999.
27. Guerra A, El-Fallah A, Soldano H. BDI multi-agent learning based on first-order induction of logical decision trees. *Conference on IAT*, Maebashi, Japan, November 2001. ■

