Towards the Use of Models for Autonomic Network Management

N. Van Wambeke, F. Armando, C. Chassot, K. Guennoun, K. Drira and E. Exposito

LAAS-CNRS - Université de Toulouse

Toulouse, France
{van.wambeke, armando, chassot, kguennou, khalil, exposito }@laas.fr

Abstract This paper presents a model-based framework to support the automated and adaptive deployment of communication services for QoS. The application domain targets cooperative group activities applied to military emergency operation management systems. Various models are introduced to represent the different levels of cooperation (applicative/middleware/transport). The adaptation decision process relies on structural model transformations while its enforcement is based on the dynamic composition of micro-protocols and software components. Automated deployment is performed both at the transport (i.e. UDP-TCP level) and middleware level. The architecture to support automated network management based on these models is introduced and illustrated.

1 Introduction

Cooperative group activities using wireless mobile communicating systems constitute an increasingly evolving application domain. It is likely to be one of the most important directions that may enable reliable and efficient human and machine-to-machine cooperation under the current networking systems and software, and may deeply shape their future deployment. Such activity-support systems have to deal with dynamically evolving activity-level requirements under constantly changing network-level unpredictable constraints. Maintaining reliable connectivity and QoS in such a communication context is difficult. Adaptive service provisioning should help the different provisioning actors to achieve this goal and constitutes a challenge for different research communities.

Ad hoc solutions are not likely to be applicable to solve such a complex problem. Providing a basic framework for automated services and QoS deployment may constitute an important contribution towards solving such a problem.

Please use the following format when citing this chapter:

Van Wambeke, N., Armando, F., Chassot, C., Guennoun, K., Drira, K. and Exposito, E., 2008, in IFIP International Federation for Information Processing, Volume 284; *Wireless and Mobile Networking*; Zoubir Mammeri; (Boston: Springer), pp. 459–470.

Aiming to answering this problem, we propose a model-based framework for adaptability management. Our framework has been elaborated in the context of network management systems with service provisioning at the transport and network layers of the TCP/IP stack as the final objectives.

Our approach provides, refines and exploits different models, each one representing a different point of view on the context. The models that represent other aspects of communication are automatically generated from higher level models representing the cooperation requirements and the communication constraints. Our research efforts have been developed to cover communication at the transport layer as well as the network layer.

Our paper is organized as follows. Section 2 describes related work. Section 3 describes the different models of the framework. Section 4 presents an architecture to support the use of these models for automated network adaptation management as well as an example of their use in response to a change of collaboration. This architecture is currently under study and development within the European NETQoS Project. Section 5 provides conclusions and future works.

2 Related Work

2.1 Classification of Context Adaptation Solutions

This section studies and classifies the main facets of adaptation: its objectives, techniques and properties.

Adaptation Objectives

Adaptation targets several objectives depending on the context in which it takes place. QoS aspects such as access bandwidth issues in roaming scenarios are considered in [7]. End to end QoS optimization for the Best Effort Internet makes heavy use of adaptation techniques [1]. Security in wireless networks, such as firewalls activation and deactivation, can also benefit from adaptability [10]. Resources optimization related to device power, computation or storage capability are presented in [9].

Adaptation Techniques

Application layer – Reference [14] addresses adaptation of video streaming applications for the Best-Effort Internet. The proposed techniques are based on two mechanisms: an applicative congestion control (rate control, rate-adaptive video encoding) and time aware error control with FEC.

Middleware layer – Reflexive architectures such as OpenORB or Xmiddle [2] are good supports for adaptation as they allow run-time modification of the architecture.

Transport layer - TCP's congestion control is a well-known adaptation example. In [1] various types of mobile applications in wireless Internet are studied. Adaptation consists in parameterization of congestion control mechanisms using context information. In [5, 6] the architectural adaptation of transport protocols by dynamic composition of protocol modules are presented, these approaches are detailed in section 2.2.

Network layer - [4] addresses QoS-aware routing problems within mobile networks. In [10], dynamic provision of IP services for military wired/wireless networks is considered. In a policy-based networking management context, the need for self-adaptation is considered in [12], using a learning-based approach.

MAC layer - The solutions handle connection and access QoS problems for mobile users using different terminals and roaming. [7] provides a solution for optimizing the handover latency but the other QoS requirements are not considered.

Adaptation Properties

The adaptation is *behavioral* when a service can be modified without modifying its structure. TCP and protocols in [1] provide behavioral adaptation. This easy to implement approach limits adaptability because the components have to be recompiled to be extended. Adaptation can not be performed during run-time.

The adaptation is *architectural* when the services' structure can be modified. The replacement components can be implemented following a plug and play approach where the new component has the same interfaces as the replaced one.

Finally, adapting components can be distributed or centralized. In the first case, adaptation is *vertical* as changes are local. In the second case, it is horizontal and synchronization between adapting peers has to be managed.

2.2 Dynamically Configurable Protocol Architectures and Model based Adaptation

Dynamically configurable protocol architectures are based on the *protocol module* concept [6]. A protocol is then viewed as the composition of various protocol modules in order to provide a given service. These architectures can be classified depending on their internal structure: the event based model and the hierarchical model. The Enhanced Transport Protocol (ETP) [5] follows a hybrid approach combining both models.

These protocol architectures are a good choice for *self-adaptation* as they provide *run-time* architectural reconfiguration. The modules composing them can change during communication. This *run-time* architectural adaptation raises many problems such as: (1) synchronization of peers; and (2) the choice of the best composition.

Adaptation management still remains a complex problem, particularly when it is required at several layers (Transport, Middleware …) simultaneously [8]. In

such cases, the need to ensure coherency of the adaptation choices, both within and between layers clearly appears.

Informal methods lead to suboptimal solutions, often specific to a problem. This is due, in part to the complexity of the problem. To overcome these limitations, graph based formal approaches are appropriate to coordinate architectural adaptation at different layers of the stack. In [3], we illustrate this approach by using graph based models and graph transformation rules. In the present paper, we complement this initial work by an architecture to manage these models.

3 The Proposed Model-based Framework

The framework is composed of three main models: **the Connection Model**, **the Cooperation Model** and **the Adaptive Deployment Model (ADM)**. The communication context is captured by the **Connection Model** while the **Cooperation Model** captures the activity cooperation context. The relationships between these models presented hereafter are summarized on Figure 1.

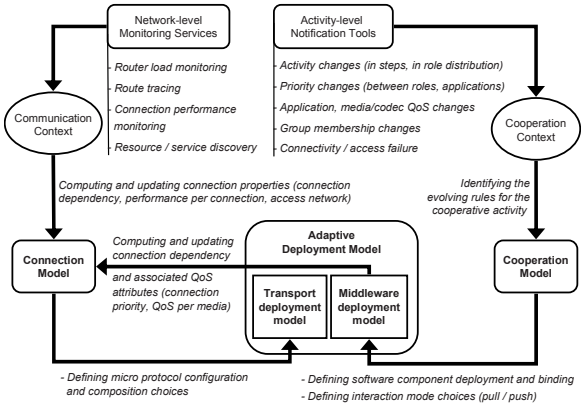

Fig. 1. Relationship between context elements and framework models

3.1 General Overview of the Framework Models

The activity requirements are derived from the cooperation context and captured by the **Cooperation Model** which captures the changes occurring at the activity level. These include modifying activity phases, role distribution, modifying priorities between roles and applications, modifying QoS parameters of applications,

media and codecs, dynamic group membership, and access and connectivity failures.

The communication context includes connection dependencies, connection performances and the characteristics of the access network which are captured by the **Connection Model**. This model expresses the connection dependencies and the associated Quality of Service (QoS) attributes including connection priority and per media QoS parameters. The communication context changes are monitored by a set of network-level monitoring services. Changes occurring at this level include router load, routing choices, connection performances, and resource and service discovering.

The **Adaptive Deployment Model (ADM)** is generated from the above two context models. It is composed of two sub-models, the **Middleware Deployment Model** and the **Transport Deployment Model (TDM)**.

The **Middleware Deployment Model (MDM)** represents the different software components supporting the information exchange between the different actors of the cooperative activity. Such components are event producers, event consumers, and channel managers interacting following the publish/subscribe paradigm or simple clients and servers interacting through direct message exchange. The different bindings of information requesters to information providers and the different interaction modes are also elements of the **MDM**. These elements can change for adaptability purposes at runtime.

The **Transport Deployment Model (TDM)** is deduced from the **Connection Model** and the **Middleware Deployment Model**. It represents the transport level decisions. In the case of dynamically configurable protocol architectures, the different protocol modules as well as their configurations are represented.

3.2 Framework Instantiation Example

In order to illustrate the use of the previously presented models, their application to Military Emergency Operation (MEO) management is presented in the next paragraphs.

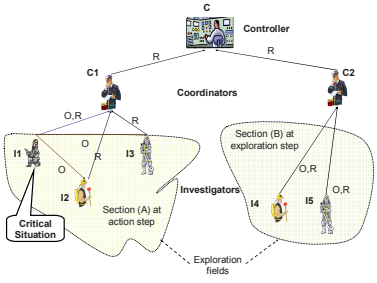

Fig. 2. Example of cooperations in MEO

3.2.1 Application context: Military emergency operation management

We consider the context of Military Emergency Operation (MEO) management systems (see Figure 2) which involve structured groups of communicating actors that cooperate to manage a given crisis. The cooperating actors have roles and use communication devices with unequal communication, processing, energy and storage resources. Devices are fixed or mobile, and communicate through wired and/or wireless networks.

Cooperation is based on data exchange between members: Observation data (O) and Report data (R) are produced periodically or immediately after a particular event. An activity controller supervises the teams, receives the coordinator reports summarizing the current situation and mission progress. According to actions and objectives assigned by the controller, a coordinator manages a team of investigators by giving orders and assigning tasks to be performed.

Investigators explore the field; they observe, analyze and submit reports of the situation to coordinators. For each team, two phases are considered. During the exploration phase (section B), investigators communications have the same priority. The action phase (section A) corresponds to the discovery of a critical situation. The investigator who discovers it is given high priority for communications.

3.2.2 Description of the elaborated models

In the context presented above, the various models introduced in our approach are detailed in the following paragraphs. As a summary, Figure 3 provides a global view of the elaborated models, their relationship as well as the different techniques used for automating their implementation.

The **Cooperation Model** (CoopM) is deduced from the cooperation context which is subject to changes, e.g. from one phase of the activity to another. When such changes happen, *reconfiguration rules* are used to automate the model's adaptation. These rules are written as *graph transformation* rules introduced in [3].

The CoopM involves (1) characterizing all valid configurations as a graph grammar, (2) describing valid configurations, (3) defining all possible structural changes at the cooperation level as graph transformation rules. This formally provides the valid reconfiguration rules and actions to be performed in reaction to changes in the cooperation context.

For cooperation level graphs, node labels represent actor identifiers, cooperation roles, hosting devices, and mission phases. Edge labels represent exchanged data, priority, and the required level of QoS.

The **Middleware Deployment Model** (MDM) is deduced from the CoopM using model *refinement rules* expressed as *graph-grammar* productions. It supports two architectural paradigms: the Client/Server and the Publish/Subscribe.

For graphs handled at the middleware deployment level, nodes represent the deployment elements. They are labeled by parameters such as type (e.g. P/C/G on Figure 4), and hosting devices. Edges are labeled by related communication characteristics such as QoS and priorities.

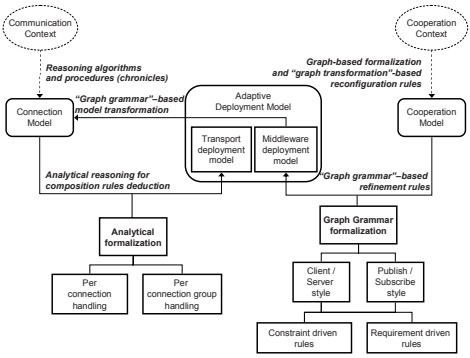

Fig. 3. Techniques for automated model adaptation in the framework

Refinement from the CoopM into the MDM is implemented using extended *edNCE graph grammars* [11]**.** The system allows the generation of all the deployment configurations with respect to a given cooperation model. On the other hand, it also allows automatic conformance verification of dynamically evolving cooperation and deployment instance models.

The **Connection Model (ConnM)** is deduced from the MDM following a set of model transformation rules expressed as *graph-grammar productions*.

The connections are represented by graph nodes as first level elements of the **ConnM**. Dependant connections are immediate neighbors in the graph.

In practice, a dependency relationship means that the two connections share at least one common resource, such as access networks or routers. The higher the dependency degree is, the more it will be suitable to coordinate the connections to improve their performance. Dependency also results from topology properties such as sharing of the sending and/or receiving hosts, or *n* common routers. Such information may be useful to *estimate* the probability of a common bottleneck when its presence cannot be determined by the monitoring services.

Node labels, such as (c1, R, QoS_R, high, AN_{MC1}, AN_{MC}, $Perf_{c1}$) on Figure 4 represent, respectively, the the connection id, the transported data type, the QoS required, the priority of the connection, the access network of the sender and the receiver, and the observed performances (i.e. delay, loss rate).

Edge labels refine the dependency degree between each pair of dependant connections, using values deduced from the MDM and monitoring information.

The **Transport Deployment Model (TDM)** is built in two stages. First, a *per-connection* decision is taken. This decision is based on reasoning procedures that take the **ConnM** as input and output the protocol modules composition to be used in a dynamically configurable transport protocol in order to optimize the QoS. This approach of our work detailed in [13]. Then, a *per-group of connections* decision is taken in order to consider dependency and priority properties. This decision refines the compositions and adds modules for managing priorities.

The reasoning process is based on two models, the *composition model* and the *decision model*. The composition model is used to define the *conditions* of the va-

lidity of the assembled protocol modules. By such, it reduces the size of the set of potential composition candidates for the reasoning process. The decision model is used to guide the process of choosing a composition among all the valid ones in order to maximize the overall user perceived efficiency (i.e. the required QoS). In previous works [13], we have shown that this problem is equivalent to a multi-criteria optimization problem given a proper formal description of the candidate protocol modules composition.

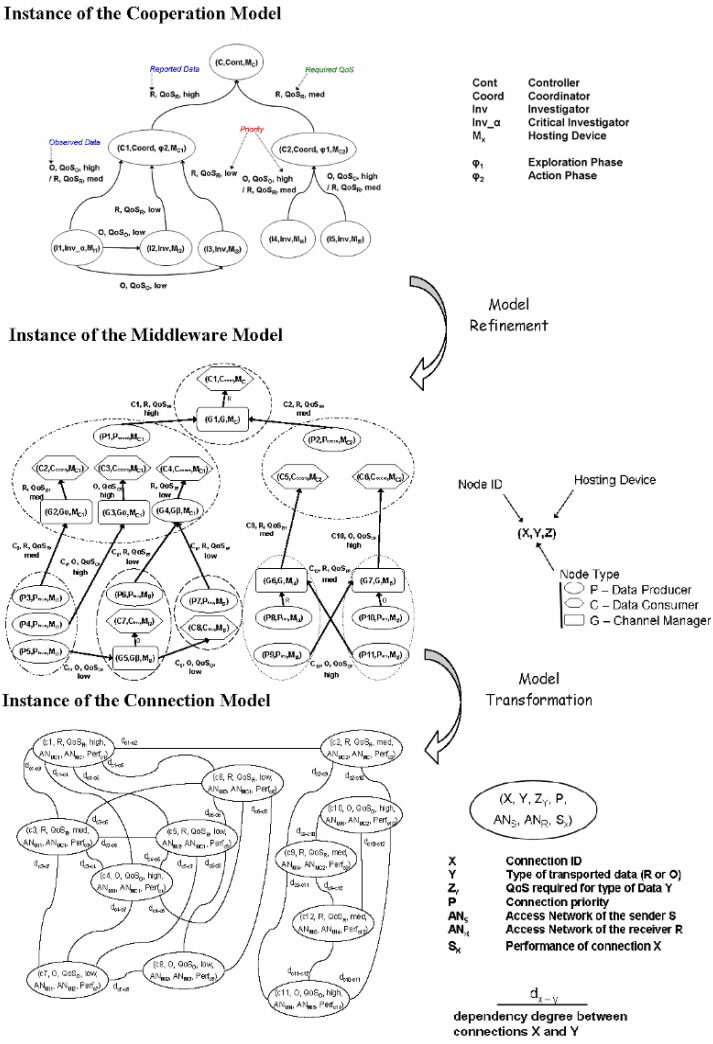

Fig. 4. Example of refinement and transformations from the Cooperation Model to the Connection Model

4 Implementing an Architecture to Support Model based Adaptation

In this section, the details and benefits of using the previously introduced models in the provisioning process is described in the context of the NETQoS IST project which addresses the problem of QoS management using a policy-based approach.

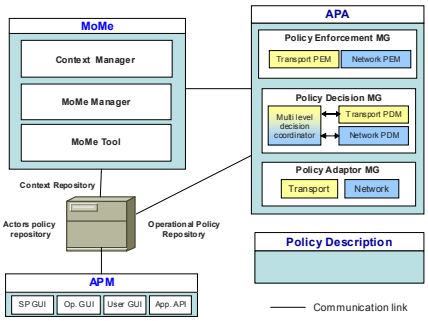

Fig. 5. General NETQoS architecture

The general architecture of the NETQoS system distinguishes four main entities (see Figure 5):

The *Policy Description* is used to specify the actor-level policies, the operational policies, etc. The *Automated Policy Adaptor* (APA) does not provide QoS by itself, but decides upon and dispatches operational policies. It is responsible for the provisioning process in which the models are used. The *Actor Preference Manager* (APM) provides NETQoS GUI/API allowing users to define policies. These policies (e.g. requirements, preferences, profile, quality reporting…) may be expressed before or during the communications. This information is used by the APA as input to the **CoopM**. The *Monitoring and Measurement* (MoMe) captures context evolution, (e.g. evolving actor's policies, end systems/network resource change). This information is used by the APA as input to the **ConnM**.

4.1 Models for the Automated Policy Adaptor (APA)

The APA decides, dispatches, and adapts the operational policies that take into account the actors dynamic requirements as well as the evolving context. It is composed of three main components. The *Policy Decision Manager* (PDM) is in charge of the decision process. This process is based on the use of the models presented in section 3. The *Policy Enforcement Manager* (PEM) is in charge of the deployment of the policies decided by the PDM on the policy enforcement points.

The *Policy Adaptation Manager* (PAM) is in charge of the adaptation, individually or by groups, when the communication or the cooperation context changes. The following paragraphs detail the three APA components implementation.

Policy decision manager (PDM)
The PDM decides an optimal set of policies to be settled at the Network and/or at the Transport level to satisfy the set of actor-level policies. This *provisioning* is performed using information contained on the **MDM** for network provisioning as well as the **TDM** for transport provisioning. The models are derived from the **CoopM and ConnM** which are constructed using context monitoring information. Each time the PDM takes a decision, it transmits it to the PEM presented below.

Policy enforcement manager (PEM)
The PEM is in charge of dispatching the PDM decisions to the actual policy enforcement point (PEP). For instance, for a transport level adaptation, the PEM dispatches the transport protocol configuration rules to be applied on the end nodes.

The PEM is independent of the network and transport technologies that are used to enforce the policies, (i.e. the PEM provides decisions in a generic language). Consequently, adaptors have to be provided on the PEP themselves to translate the generic PEM rules into specific technology-dependant rules.

Policy adaptation manager (PAM)
The PAM is in charge of the adaptation when the context changes. It may decide to adapt the policy by re-creating a completely new one or simply amending the one in place.

The PAM mainly acts when it receives alarms from the MoMe component informing it that the communication or cooperation contexts have changed. It performs adaptation by applying the *graph-grammar* transformations to the actual **Connection and Cooperation Models** and re-generating derived (**MDM, TDM …)** models.

4.2 Example: Model Based Adaptation

In this section, the Military Emergency Operations context presented in section *3.2.1* is used to illustrate the adaptation steps that take place on the models constructed by the APA when the cooperation goes from the exploration step to the action step. This adaptation corresponds to the discovery of a critical situation by one of the investigators taking place in the activity.

For instance on the **MDM** presented on Figure 6a, the investigators M_1 and M_2 are in the exploration step. Two channels are implemented: each one is in charge of a specific (data, priority) couple. Assuming a mobile participant is allowed to host only one event service component, the two channel managers (CMs) are deployed on participants M_1 and M_2.

When an investigator discovers a critical situation, the policies/preferences change and MoMe component informs the APA of this event. The APA then modifies the **Cooperation Model** according to the user's preferences. In this case, the user has specified that in exploration step, each investigator reports directly to its coordinator while in the action step, the investigator who discovered a critical situation reports to both his direct coordinator as well as his fellow investigators.

To support the changes of the **Cooperation Model** when the mission goes from the exploration step (step 1) to the action step (step 2), several adaptation actions are performed on the **MDM**. These actions lead to different possible architectures, one of them is illustrated on Figure 6b.

Architecture transformation is guided by rules that consider not only changes of the **Cooperation Model**, but also changes of resource-oriented parameters such as machines' energy and storage/computation capacity. For instance, in the action step, four CM are implemented: one per (data, priority) couple. In order to save energy, two CM are deployed on the controller's machine M_3, and only one CM is deployed on each of the investigators' machine M_1 and M_2. Such transformation may also be caused by resources parameters only. In such cases, a "good" transformation should have no impact on the upper **Cooperation Model**.

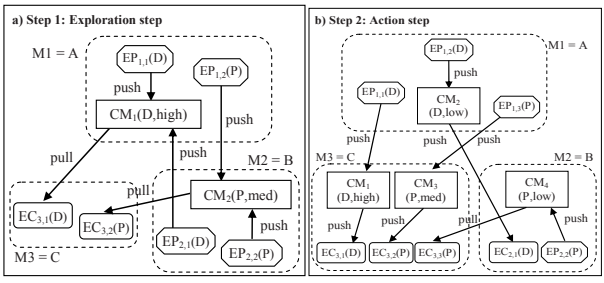

Fig.6 Graph transformations from exploration step to action step – MDM

This new **MDM** is used to automatically update the **Connection Model**, both models will then be used to adapt the policy in place and deploy the updated decision on the different network and transport PEP resulting in optimized QoS to all users in this new step of their mission.

5 Conclusion and Perspectives

In this paper, different models have been elaborated and implemented to help automating adaptive deployment for QoS management. Different points of view have been considered to capture the influence of the cooperation as well as the communication contexts. The interest of our approach resides in its capacity to support the full automation of the network management tasks in evolving contexts.

An architecture that uses these models in the context of automated policy based network management has been presented. This architecture allows activity-level requirement to be expressed by the user and have the system behave accordingly. The use of the models in the decision process has been illustrated in the context of simple Military Emergency Operations.

Extending the work presented here, future work includes the implementation and benchmarking of the NETQoS architecture. Scalability issues as well as convergence speed of the decision algorithms have to be addressed.

Acknowledgments: Part of this work is done in the context of the EU IST FP6 NETQoS (STREP) project and by a French DGA (Direction Générale de l'Armement) grant.

References

1. Akan O B, Akyildiz I F (2004) ATL: An adaptive transport layer suite for next-generation wireless Internet. IEEE Journal on Selected Areas in Communications 5:802–817.
2. Capra L, Emmerich W, Mascolo C (2003) Carisma: Context-aware reflective middleware system for mobile applications. IEEE Transactions on Software Engineering 10:929–945.
3. Chassot C, Guennoun K, Drira K, Armando F, Exposito E, Lozes A (2006) Towards autonomous management of QoS through model driven adaptability in communication-centric systems. Int Transactions on Systems Science and Applications 3:255–264.
4. DaSilva L A, Midkiff S F, Park J S, Hadjichristofi G C, Davis N J, Phanse K S, Lin T (2004) Network mobility and protocol interoperability in ad hoc networks. IEEE Communications Magazine 11:88–96.
5. Exposito E J (2003) Design and implementation of quality of service oriented transport protocol for multimedia applications. PhD thesis, National Polytechnic Institute of Toulouse.
6. Hutchinson N C, Peterson L L (1991) The x-kernel: An architecture for implementing network protocols. IEEE Transactions on Software Engineering 1:64–76.
7. Kaloxylos A, Lampropoulos G, Passas N, Merakos L (2006) A flexible handover mechanism for seamless service continuity in heterogeneous environments. Computer Communications 6:717–729.
8. Landry R, Grace K, Saidi A (2004) On the design and management of heterogeneous networks: A predictability-based perspective. IEEE Communications Magazine, 11:80 – 87.
9. Marshall I, Roadknight C (2001) Provision of quality of service for active services. Computer Networks 1:75–85.
10. Perez G, Skarmeta A G (2004) Policy-based dynamic provision of IP services in a secure vpn coalition scenario. IEEE Communications Magazine 11:118 – 124.
11. Rozenberg G (1997) Handbook of Graph Grammars and Computing by Graph Transformation, World Scientific Publishing, ISBN 981-02-2884-8
12. Samaan N, Karmouch A (2005) An automated policy-based management framework for differentiated communication systems. IEEE Journal on Selected Areas in Communications 12:2236-2248.
13. Van Wambeke N, Armando F, Chassot C, Exposito E (2008) A model-based approach for self-adaptive Transport protocols. Comput Commun. doi: 10.1016/j.comcom.2008.02.026.
14. Wu D, Hou Y T, Zhu W (2001) Streaming video over the internet: Approaches and directions. IEEE Transactions on Circuits and Systems for Video Technology 11:282-301