Design for a generic knowledge base for autonomic QoE optimization in multimedia access networks

Steven Latré, Pieter Simoens, Bart De Vleeschauwer, Wim Van de Meerssche,
Filip De Turck, Bart Dhoedt and Piet Demeester
Ghent University - IBBT - IBCN - Department of Information Technology
Gaston Crommenlaan 8/201, B-9050 Gent, Belgium
Tel: +3293314981 , Fax: +3293314899
e-mail: steven.latre@intec.ugent.be

Steven Van Den Berghe, Edith Gilon - de Lumley
Alcatel-Lucent Bell Labs
Copernicuslaan 50, B-2018 Antwerpen, Belgium

***Abstract*—The management of access networks to guarantee the Quality of Experience for multiplay services, is complicated by the heterogeneity in service specifics and home network configurations. The appropriate action to restore the QoE is highly dependent on the type of service, the affected user, etc. This requires an almost per-user and per-service management. Through autonomous reasoning, the required QoE restorative actions for underperforming services can be determined in a scalable way. This reasoning process needs a solid knowledge base, containing information about the network, the services and the QoE restorative actions that can be undertaken. This article gives an overview of these requirements and presents an OWL/SWRL based knowledge base that can be used in an autonomous QoE management architecture for the access network. Besides storing all information, the proposed knowledge base will autonomously deduce new information. This new information can be used to raise alarms to higher layers in the architecture or to autonomously propose QoE restorative actions for underperforming services.**

I. Introduction

In today's broadband access networks, operators are focusing on the deployment of new added value services such as IPTV over DSL networks, Video on Demand (VOD) and on-line gaming. Of prime importance in defining the quality of these services is the Quality of Experience (QoE): the quality as experienced by the end-user. Each of these services has large service demands in terms of bandwidth, tolerable packet loss, delay and jitter. In order to meet these demands, current access networks are evolving from a best-effort service delivery to a QoS-aware triple-play service delivery. The management of these networks to guarantee the Quality of Experience has been complicated by several factors. First, the degradation of service QoE due to network anomalies such as packet loss and jitter, is highly dependent on the type of service and the current context of the network. For example, an interactive service such as VoIP is typically more vulnerable to delay than a broadcast service. Second, user home networks are complex networks of their own, consisting of different technologies (e.g. wired and wireless) and different user devices (e.g. set top boxes, traditional PCs, etc.), with users concurrently accessing different services.

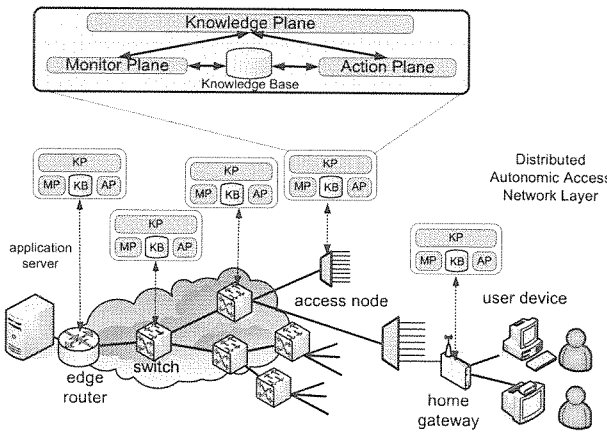


Fig. 1. Three-layer autonomous framework for QoE optimization in the access network. It consists of both distributed and central components.

Due to this heterogeneity in service specifics and home network configurations, every drop in service QoE must be tackled in a different way. Autonomic management of the access network offers a solution that is able to perform a per-user and per-service management in a scalable way. Through autonomic reasoning, the best suitable QoE restorative action (e.g. performing retransmissions) is chosen in each situation. We defined an autonomic architecture to maximize the QoE of all running services in the access network. This architecture spans the complete network from service originator up to the end user, as shown in Figure 1. It consists of 3 layers: the Monitor Plane (MPlane), the Knowledge Plane (KPlane) and the Action Plane (APlane). Their functionality is detailed in section III. A solid knowledge base is required to build the management framework upon. In this paper, we report about the design of an ontology for autonomous QoE reasoning in the access network.

The remainder of this paper is structured as follows. In section II, we explore related work in the field of ontolo-

978-1-4244-2067-4/08/$25.00 (c)2008 IEEE

gies and autonomous QoE reasoning. Our generic reasoning architecture and its requirements for the knowledge base are discussed in respectively section III and IV. Section V presents the designed ontology and section VI details how this ontology can be used to incorporate autonomic behavior where the right configuration of QoE optimizing actions is autonomously chosen when a QoE drop occurs. The performance of an ontology approach is discussed in section VII, while section VIII concludes this paper.

II. Related work

The concept of a Knowledge Plane was originally presented in [1] as a layer that enables the autonomic detection and recovery of network anomalies. We applied the Knowledge Plane concept in an access network and have first set out the architecture of an autonomous access network Knowledge Plane in [2]. We focus on the complexity of the Quality of Experience management in networks, incurred by the wide variety of new services that has emerged in recent years, such as Broadcast TV, Video on Demand, IP telephony, etc. In previous research, we have concentrated on monitoring algorithms for the Monitor Plane, to provide the Knowledge Plane with a set of information of the network status and the QoE of the running services [3], [4], [5]. These algorithms were successfully integrated on the access node of a network equipment vendor [6]. In [7], we reported on the architecture of an autonomic Knowledge Plane component on the access node and an analytical component for the reasoning. The concept of applying neural networks to implement the Knowledge Plane to enhance the QoE was introduced in [8], together with some first results and design guidelines.

All previous papers focused on the different steps of the autonomic loop (monitoring, reasoning and taking QoE optimizing actions). In each step, more knowledge becomes available, and the need of gathering this plethora of information in a scalable and highly accessible manner increases. In this paper, we are targeting a knowledge base that stores all relevant information to the problem domain and provides the higher layers (MPlane, KPlane and APlane) with a solid basis of knowledge. We propose to use ontologies, modeled through the Ontology Web Language (OWL) [9] in combination with a rule based extension called Semantic Web Rule Language (SWRL) [10]. Ontologies are mainly used in the context of the Semantic Web but have also proven their use in the design of context-aware applications [11], [12]. Furthermore, ontologies have been successfully applied to computer networks related topics such as sensor networks [13] and P2P routing [14]. In the domain of network management, ontologies have also been proposed. The authors of [15] argue that ontologies alone do not suffice to perform a complete network management and use a combination of an ontology model and policies. In [16], an ontology is presented to model service operator policies to represent both the data and relationships between the data for all stakeholder views. The ontology design presented in this paper complements the designs presented in [15] and [16] but focuses more on a way to optimize the QoE of running services. Furthermore, we use the SWRL rule language to deduce additional information and introduce autonomic behavior.

III. Architecture

In Figure 1, a functional view on the architecture is presented. It spans the complete network from service originator to the end-user and its functionality is defined through three separate layers: the Monitor Plane, The Knowledge Plane and the Action Plane. These 3 layers are constructed around a central knowledge base and map to the Monitor-Analyze-Plan-Execute (MAPE) autonomous control loop that was originally presented by IBM [17]. The architecture is presented in Figure 2.

The main objective of the MPlane is to provide the autonomic loop with a complete and detailed view of the network. Probes at every network element (e.g. access nodes, video servers) monitor parameters such as packet loss, router queue sizes etc. More service specific information can also be monitored such as the frame rate of video services. The data generated by the MPlane algorithms can be summarized both temporarily (e.g. calculating an average over the last hour) and spatially (e.g. comparing retransmission requests for the same service from different users). The knowledge base thus not only comprises raw monitoring data, but should also contain processed monitor info, to facilitate the reasoning process of the Knowledge Plane.

The Knowledge Plane determines in an autonomic way the right QoE optimizations to take, based on the input from the Monitor Plane. We defined a three-step process where each step analyzes the knowledge currently available and forwards its decision to the next component in the chain. First, suboptimal situations that lead to QoE drops in the controlled network are detected. The relevant knowledge is forwarded to the problem tracking step. Here, the location and cause of the QoE drop are pinpointed. In some cases, the reasoning component might decide that the currently available knowledge is not sufficient to determine the appropriate actions to take. In such a case, it can decide to perform additional tests through active monitoring. The final step in the Knowledge Plane is the problem solving step, where the appropriate QoE restorative action is determined. Examples of restorative actions include applying Forward Error Correction (FEC) to a bit stream to make it more resilient against packet loss, switching a video to a lower bit rate, etc. The KPlane operates in tight interaction with the knowledge base. It extracts information about the network configuration, the available restorative actions and the specifics of the running services. The knowledge base also contains the history of previously undertaken actions and their effect. This allows to improve its decision process by evaluating the effect of previous actions in similar circumstances, a function of the learning controller. Additionally, the knowledge base will have an autonomic component of its own that can evaluate the knowledge available and trigger some actions on the higher layers. For example, the knowledge base can analyze the measured monitor values and warn the KPlane when a monitor value exceeds a certain threshold.

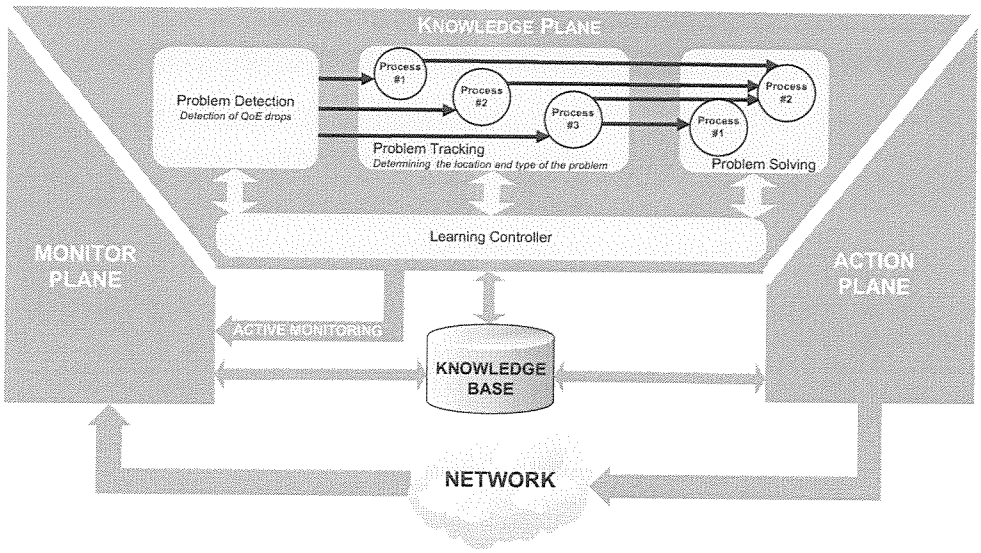


Fig. 2. A generic and autonomic architecture to enable the QoE maximization of multimedia services.

The Action Plane is responsible for the execution of the QoE optimization actions determined by the KPlane. The Action Plane receives a complete configuration of which actions to take from the KPlane. Based on this configuration, the APlane instructs specific nodes to alter their network configuration. The knowledge base must thus contain an overview of the possible actions and the capabilities of each network node.

The Knowledge Base provides a way of storing all knowledge derived from the three planes and can be seen as the basis for enabling autonomic behavior. This autonomic behavior is primarily achieved through the KPlane functionality, which uses the knowledge base information to steer both the MPlane and APlane. As the solution proposed in this paper is able to propose new QoE restorative actions for underperforming services, our knowledge base will also incorporate autonomic behavior. However, this autonomic behavior should be seen as a fast way of autonomously solving local problems. In the past, we discussed the design of specific KPlane components based on neural networks [8]. The autonomic behavior introduced by the knowledge base complements these components but does not try to replace them.

IV. Requirements analysis

From the previous section, it is clear that every plane in the architecture operates in tight cooperation with the knowledge base. Therefore, it must be properly structured so that it can be easily accessed and queried by each plane. Furthermore, the myriad of knowledge must be stored in a generic representation that can express the complete network, the services and the possible QoE restorative actions. On top of that, it should be easily extensible to allow adding new services, actions, etc. In this section, we provide a detailed overview of all requirements for the knowledge base.

A first set of requirements is related to the network topology. Our architecture targets an access network model, spanning all network elements along the end-to-end path from service originator up to the end user. The knowledge base must contain a network model with the topology, link and node characteristics. An important challenge is how to model the home network topology and its devices in a generic way. As stated in the introduction, this is a complicating factor in the QoE management.

Once the network model is contained in the knowledge base, knowledge must be added about the services running over the network and the policy for each of these services. The network route of each service is valuable information, since present access networks transport a lot of multicast services such as Broadcast TV. Other important service parameters are the location and number of the requesting users. For these type of services, it is important to correlate the information of all users requesting the service. Furthermore, the operator must be able to define a policy for each service. This includes QoS transport classes in the access network, tolerable delay, jitter, etc.

In order to determine the appropriate action, the reasoner must know which actions can be performed at which location. For example, it is important to know which nodes are capable of adding FEC packets to a bit stream. This also achieves the distributed behavior of the Knowledge Plane, present in all nodes of the access network. Combined with the policy defined by the operator, the knowledge base should allow easy detection of underperforming services.

The information about the network topology, the provided services and the QoE restorative actions, is rather static. The knowledge base must also contain more dynamic parameters, reflecting the actual network status. The Monitor Plane continuously fills up the knowledge base with information about the status of all network components. It is not scalable to keep all this information in the knowledge base. Therefore, structures

must be present to represent higher level monitor information, aggregated over time and/or space.

V. An OWL/SWRL based knowledge base

In this section we propose an ontology based approach as knowledge base implementation. The proposed ontology uses OWL to model the different concepts and SWRL as a basic rule language for deriving new information. This section is organized as follows. First, the basic concepts of ontologies and their advantages with respect to other representations are explained. Second, we detail how the different concepts discussed in section III and IV can be modeled and how SWRL can be used to detect problematic services.

A. Introduction to ontologies, OWL and SWRL

A brief definition of an ontology can be found in [18]: "An ontology is a specification of a conceptualisation in the context of knowledge sharing". An ontology describes in a formal manner the concepts and relationships, existing in a particular system and using a machine-processable common vocabulary within a computerised system. An ontology is constructed by formalising the concepts in the problem domain and specifying the relationships and logic that exist behind the concepts. For example, monitoring the packet loss of a service can be defined by constructing two concepts: $PacketLoss$ and $Service$. The two concepts can be connected by defining a relationship between $PacketLoss$ and $Service$ called $isMonitorValueFor$.

As ontologies incorporate first order logic, they go beyond class definitions in object oriented programming languages or regular databases. Once the concepts and relationships are chosen, logic expressions and inference rules can be declared to further deduce information about the concepts and properties within the ontology. OWL is a modeling language for ontologies based on XML and incorporates the ability to form logic expressions to classify existing individuals into concepts. For example, using OWL, a $BadUser$ can be defined as subconcept of the $User$ concept that has at least one relationship with the concept $BadService$, where a $BadService$ concept denotes a problematic service. Using an off-the-shelf reasoner and these logic expressions, an ontology can autonomously deduce this new information. SWRL augments the OWL language with the ability to add rules to an OWL knowledge base. These rules provide the freedom to insert new individuals into the knowledge base and use mathematical equations or any other defined relationship. For example, based on two concepts: a $PacketLoss$ and a $PacketLossPolicy$ (stating how high the packet loss may be) a $PacketLoss$ instance can be classified as an instance of the $BadPacketLoss$ concept if its value is greater or equal than the one of the $PacketLossPolicy$. Such behavior is not possible by only using OWL. More generic examples are described in the next paragraph.

B. Ontology design

We constructed an ontology using OWL that incorporates the requirements of the knowledge base described in section IV. For each function, we describe a high level view of the ontology submodel that represents this information. Furthermore, we illustrate how we used the concepts of OWL restrictions and SWRL rules to come to an autonomic component inside the knowledge base itself that is capable of raising alarms if needed.

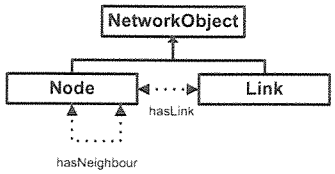


Fig. 3. Ontology submodel for describing the network topology. The $hasNeighbour$ relationship will be autonomously deduced using SWRL

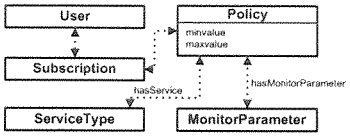


Fig. 4. Ontology submodel for describing the services available in an access network. Using the policy concept, different policies for each service can be enforced. SWRL provides capabilities for autonomously deducing problematic concepts (services, users, etc.).

1) Network topology information: Figure 3 shows how the network topology can be modeled in an ontology. We define a network as being a set of network objects. Each network object can either be a link or a node. The relationship between these two concepts states how nodes are connected to links and vice versa. This is a rather trivial solution but one that immediately highlights an advantage of employing ontologies: based on existing topology information new information can be deduced. Here, SWRL can be used to define the symmetric $hasNeighbour$ relationship; two different nodes will be neighbours if they share the same link. When using SWRL, a rule inference engine will add the $hasNeighbour$ relationship between any two instances of the $Node$ concept if rule (1) applies for those instances. As the $hasNeighbour$ relationship is defined as symmetric, the $hasNeighbour(?b, ?a)$ relationship will automatically be added too.

$$Node(?a) \wedge Node(?b) \wedge differentFrom(?a, ?b) \\ \wedge hasLink(?a, ?l) \wedge hasLink(?b, ?l) \wedge Link(?l) \\ \Rightarrow hasNeighbour(?a, ?b) \quad (1)$$

2) Service and policy information: A more complex case is modeling administrative information such as service and policy information. This part of the ontology enables an operator to model which users have access to the network, what kind of subscription they have and what services they may access. As illustrated in Figure 4, an instance of a $User$ concept can have a $Subscription$ (e.g. a high speed internet subscription) and each subscription can have access to one or more service types (e.g. a BroadcastTV service), denoted by the $ServiceType$ concept. Furthermore, an operator can

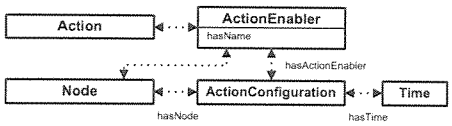


Fig. 5. Ontology submodel for describing the QoE optimizing capabilities and the past and current configuration of each node.

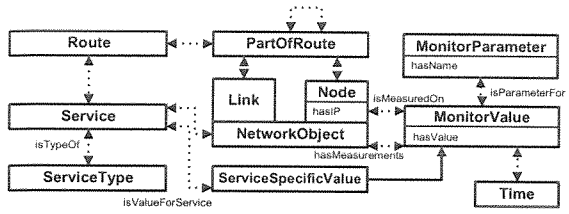


Fig. 6. Ontology submodel for describing the monitoring of services inside an access network. The "Route" and "PartOfRoute" concepts can be deduced using SWRL. A combination of SWRL and OWL can also be used for deducing network anomalies (bad monitor values, bad nodes, etc.).

enforce a policy for each service and subscription type. We defined a policy as being a minimum and maximum value for any combination of the $MonitorParameter$, $ServiceType$ and $Subscription$ concepts. These values describe the minimum and maximum value an instance of the $MonitorParameter$ concept (e.g. packet loss ratio) must have to be classified as a normal $MonitorParameter$. A policy will be defined for each combination of $ServiceType$, $Subscription$ and $MonitorParameter$. As will be discussed later, if a monitor value does not fit the range specified in a policy, the ontology will autonomously detect this and a flag will be raised activating the problem detection step in the architecture described in section III.

3) Information about QoE optimizing actions: This submodel will describe the capabilities present in the Action Plane and the configuration of each node at a given time. We want to describe static information such as the different high-level actions, how these actions can be implemented and the QoE optimizing capabilities of each node in the network. We also require more dynamic information such as the current and past configuration of each node in terms of QoE optimizing actions. Figure 5 denotes the concepts and relationships available in this ontology. In this submodel, instances of the $Action$ concept (e.g. applying FEC) have different $ActionEnabler$ instances (e.g. generate FEC parity packets, decode the parity packets and reconstruct the stream) which represents the static Action Plane knowledge. The dynamic ActionPlane knowledge is represented through the $ActionConfiguration$ concept which defines which instances of the $ActionEnabler$ concept exist for a given time and node. This enables us to incorporate past information, which can be necessary for the reasoning step in the Knowledge Plane.

4) Monitor information: As illustrated in Figure 6, the information available in this submodel is the most advanced of the four submodels and represents the knowledge available in the Monitor Plane. Here, different aspects are described: the complete set of monitor values measured, the different services currently running over the network, the different routes for each service, etc. Therefore, several new concepts are defined. A concept $MonitorValue$ and its child concept $ServiceSpecificValue$ is introduced. These concepts describe the actual measured values and have a relationship with a certain $MonitorParameter$ value, that describe a certain metric that can be monitored (e.g. packet loss, jitter, delay).

Instances of the $ServiceSpecificValue$ concept represent monitor values directly related to a specific service (e.g. the measured packet loss ratio for a VoD running from node A to B) and therefore have a relationship with a new concept: $Service$. The $Service$ concept defines an actual service running in the network between two or more endpoints. As different type of services (e.g. BroadcastTV, VoD, on-line gaming) can occur in the network, a service will have some service type. Furthermore, the $MonitorValue$ concept has a relationship with a pre-defined $Time$ concept to mark the time at which a measurement was performed and relationships with the $Node$ and $NetworkObject$ concept to mark where a measurement was performed and what part of the network was measured, respectively. Finally, each service can have one or more routes, defined as a "linked list" of $PartOfRoute$ concepts. This information can also be deduced using OWL and SWRL.

$$\begin{aligned} &ServiceSpecificValue(?mon) \wedge hasValue(?mon, ?val) \\ &\wedge isValueForService(?mon, ?ser) \wedge Service(?ser) \\ &\wedge Policy(?pol) \wedge hasService(?pol, ?sertype) \\ &\wedge ServiceType(?sertype) \wedge isTypeOf(?sertype, ?ser) \\ &\wedge hasMonitorParameter(?pol, ?monpar) \\ &\wedge isParameterFor(?mon, ?monpar) \\ &\wedge MonitorParameter(?monpar) \\ &\wedge minvalue(?pol, ?minval) \wedge maxvalue(?pol, ?maxval) \\ &\wedge [swrlb : lessThan(?val, ?minval) \\ &\vee swrlb : greatherThan(?val, ?maxval)] \\ &\Rightarrow BadServiceSpecificValue(?mon) \end{aligned} \tag{2}$$

This submodel provides the knowledge base with a huge amount of information that changes rapidly over time. Therefore, the monitor information can be seen as the most crucial and largest piece of information available in the ontology. Consequently, it is important to further classify this large set of data to have a better look at the different pieces of information available without needing to look at each single monitor data value. Here, we explain how SWRL can be used to aid by means of alarm raising and give some examples. The general idea is to use SWRL to determine whether a particular monitor value can lead to network anomalies by comparing it with some reference values (e.g. described in the $Policy$ concept). If such a value exceeds the allowed values, it will be classified

as being part of a specific concept: $BadMonitorValue$ or $BadServiceSpecificValue$ for general and service specific monitor values, respectively.

By adding different SWRL rules that will classify the monitor values into $BadMonitorValue$ and $BadServiceSpecificValue$ concepts, the operator can enforce a policy that dictates when a certain network configuration can be seen as a problematic situation. This forms the basis for the higher layer Knowledge Plane functionality of detecting problems based on the information available in the knowledge base. A example of such a policy can be that any type of monitor parameter for a specific service may not exceed a certain threshold. Rule (2) illustrates how this can be translated to SWRL. At first sight, this seems rather complex but the majority of terms in the rule are constructs to support some of the specifics of ontologies (e.g. open world reasoning). The terms $swrlb : lessThan$ and $swrlb : greaterThan$ are built-in SWRL functions that perform numerical comparison.

Based on the new concepts of $BadMonitorvalue$ and $BadServiceSpecificValue$ additional concepts can be defined that classify bad users, nodes, links, services, routes, etc. Again, the operator has the freedom of defining what the specifics of these concepts are. For example, one operator can define a $BadNetworkObject$ concept as a network object that has at least one $BadMonitorValue$ or $BadServiceSpecificValue$, while another operator will require that all monitor values are instances of the $BadServiceSpecificValue$ or $BadMonitorValue$ concepts. This can be expressed fairly straight forward using OWL restrictions (3) and (4) on the $BadNetworkObject$ concept, respectively.

$$\text{BadMonitorValue} \cup \text{Bad ServiceSpecificValue} \quad \exists hasMeasurements \quad (3)$$

$$\text{BadMonitorValue} \cup \text{Bad ServiceSpecificValue} \quad \forall hasMeasurements \quad (4)$$

VI. Detailed scenario description

As described in section V the proposed ontology can classify existing monitor information to raise alarms if a monitor value exceeds the allowed values stated in a policy. Through this alarm raising technique, drops in the QoE of monitored services will be detected. However, real autonomic behavior also means that the right QoE optimizing actions should be executed in response to a QoE drop. In this section, we present how the ontology can be used to realize such autonomic behavior.

To illustrate the QoE optimizing behavior of both techniques, we employed a scenario where a number of Broadcast TV (BCTV) channels are being streamed in the access network. However, in contrast to a typical BCTV service, the streaming server transmits the same video stream in different bit rates. For each bit rate version, the server also transmits a stream of Forward Error Correction (FEC) packets. These FEC packets enable the client to restore the data packet stream in case a number of data packets are lost at the cost of additional traffic. We use optimal erasure codes, where the reception of N packets out of N data packets and K FEC packets allows the receiver to reconstruct the original N data packets. For this scenario, we position the architecture consisting of the knowledge base and the three planes on the access node. The MPlane in the access node gathers information on the packet loss between the access node and the end device. Also information about the used bandwidth on the access line is collected. The obtained monitor values are stored in the ontology and monitor values that indicate a QoE drop will be autonomously detected through the definition of a policy. In this scenario, a policy can state that the packet loss ratio of a BroadcastTV service type must be between 0% and 0.5% and the used bandwidth must be between 0% and 0.90% of the total link capacity. When a monitor value exceeds the value stated in its policy it will be classified as a problematic value (for example, as a result of rule 2). The KPlane will then need to propose a QoE optimizing action to the APlane to restore the original QoE . In the proposed scenario, the APlane can take two QoE optimizing actions. First, it can lower or increase the video bit rate to solve congestion or increase the video quality, respectively. Second, it can add more or less FEC packets to tackle packet loss due to bit errors.

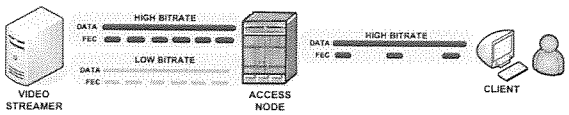


Fig. 7. An example of an action in the described scenario. In the access node, it is decided that the client should receive the high bandwidth channel. The appropriate number of FEC packets is also determined.

Figure 7 shows an example of the actions that can be taken in the access node. The KPlane component decides that the client should receive a high bandwidth video and a number of FEC packets. This can correspond to a scenario where there is packet loss on a link in the home network or on the xDSL line and where enough bandwidth is available on the access line to support both the high bandwidth version of the BCTV channel and the FEC packets. Only some of the generated FEC packets are streamed to the client, as no more are needed to solve the packet loss.

While the proposed ontology is a way of constructing a knowledge base, its SWRL based reasoning technique can also be used to perform the KPlane functionality. This augments the ontology from a knowledge base capable of performing alarm raising to an autonomic reasoner that can use its knowledge to optimize the QoE of services. For example, SWRL can be used to express the amount of FEC packets that are necessary to cope with a certain level of packet loss. This is illustrated in rule (5) which sets the number of FEC packets to $\left\lceil \frac{N*Ploss}{1-Ploss} \right\rceil$, where $Ploss$ is the measured packet loss and N is the number of data packets in the FEC stream. This

value is the theoretic value K of parity packets needed to cope with $Ploss$ packet loss. In this rule, $BadMonitorValue$ and $RecentMonitorValue$ are two derived concepts that will autonomously be deduced using an ontology reasoner. The $BadMonitorValue$ was introduced in section V, while $RecentMonitorValue$ denotes the last added monitor value. The IP addresses of the access node and client, together with the FEC parameter N can be seen as a constant. This is just an illustration of a possible rule. More complex rules are also possible that combine different actions together. When this ontology approach is used, a measurement value that exceeds a policy will autonomously be classified as problematic. This classification will in turn fire some SWRL rules that add a particular instance of the $ActionConfiguration$ concept to the knowledge base. Finally, this will trigger the APlane that will execute the new configuration into the network.

$$\begin{aligned} & BadMonitorValue(?m) \wedge hasValue(?value) \\ \wedge & MonitorParameter(?p) \wedge hasName(?p,' \text{packetloss}') \\ & \wedge isParameterFor(?p,?m) \\ & \wedge RecentMonitorValue(?m) \\ & \wedge Node(?client) \wedge Node(?access\_node) \\ & \wedge isMeasuredOn(?m,?access\_node) \\ & \wedge hasIP(?access\_node, < ip\_of\_access\_node >) \\ & \wedge hasMeasurements(?client,?m) \\ & \wedge hasIP(?client, < ip\_of\_client >) \\ & \wedge swrlb : multiply(?tmp1,?value,N) \\ & \wedge swrlb : subtract(?tmp2,1,?value) \\ & \wedge swrlb : integerdivide(?fecvalue,?tmp1,?tmp2) \\ & \wedge ActionEnabler(?filterFEC) \\ & \wedge Time(?now) \\ & \wedge hasName(?filterFEC,' \text{Filter FEC}') \\ & \Rightarrow ActionConfiguration(?fec) \\ & \wedge hasValue(?fec,?fecvalue) \\ & \wedge hasActionEnabler(?fec,?filterFEC) \\ & \wedge hasNode(?fec,?access\_node) \\ & \wedge hasTime(?fec,?now) \end{aligned} \tag{5}$$

The ontology can provide the architecture with the KPlane reasoning functionality as long as the reasoning process can be expressed in rules and the rules are not too complex. However, in some cases it is difficult to find a clear mapping between the measured values (e.g. packet loss ratio) and the desired QoE optimizing action (e.g. how much FEC to apply). In this case, more advanced techniques are necessary. In the past, we proposed two reasoning components for the KPlane that both are able to determine the best QoE optimizing actions for the described scenario when random packet loss or congestion occurs. The first is a analytical reasoner, based on a set of equations, the second is neural network approach. We refer to [8] for more information. Both approaches use the knowledge base functionality presented by the ontology. Using the additional capabilities of the ontology to also propose QoE optimizing actions can be seen as an extension to the knowledge base and a third approach in implementing the KPlane functionality. All approaches are viable solutions to find the best QoE optimizing actions in response to a QoE drop. The ontology approach has the advantage of simplicity as it uses the already employed knowledge base; the analytical and neural network based approach will be more powerful.

TABLE I
PERFORMANCE OF THE ONTOLOGY IN TERMS OF TIMING CONSTRAINTS OF READ AND WRITE OPERATIONS FOR TWO COMMON ONTOLOGY FRAMEWORKS. IN THIS CASE, JENA CLEARLY UNDERPERFORMS WHEN COMPARED TO THE PROTÉGÉ OWL API.

	Monitor value operation time (ms)			
Framework	Read	Write	STDEV(Read)	STDEV(Write)
JENA	6.7767	22.3111	0.068967	0.134956
Protégé OWL	2.2912	5.7113	0.00426	0.08381

VII. PERFORMANCE EVALUATION

In this section, we evaluate the performance of our ontology approach in terms of timing constraints. The framework presented in section 2 is targeted to operate in real-time on the scale of an access network. The ontology will be frequently updated with monitor values and/or action configurations to improve the QoE. Therefore, the time for the reasoner to check the ontology consistency and/or to detect failing services is an important parameter. We implemented the proposed ontology on two common ontology frameworks: the JENA framework complemented with the SDB library [19] and the Protégé-OWL API [20]. Both frameworks used a PostgreSQL 8.2.6 database as a storage backend. We measured the time necessary to store and retrieve a monitor value. All tests were performed on an AMD Athlon 2200Mhz processor with 512MB RAM and repeated 10.000 times. Table I illustrates the results of these tests. It is clear that the Protégé OWL API outperforms JENA. As storing a monitor value takes 5.7ms in the Protégé OWL API, approximately 175 monitor values can be stored per second.

These tests highlight the importance of summarizing the available monitor information. As different services are passing through each node in the network, a huge amount of monitoring information will be available. The network nodes will need to aggregate this information and signal it to the centralized autonomic architecture. The amount of aggregation is a key factor in the effectiveness of this approach. Additionally, local autonomic architectures that only have a limited knowledge about the network can be placed in the network nodes themselves. In this case, the network nodes tackle local problems and signal a more aggregated version to the centralized architecture, which solves crucial QoE drops. Investigating the level of aggregation needed and improving the performance of the ontology (e.g. by employing a hybrid in-memory and database approach) is part of future work.

VIII. Conclusions and future work

We designed an ontology that models the knowledge present in an autonomic architecture that optimizes the QoE in multimedia access networks. The ontology forms a knowledge base for three higher layers: a Monitor Plane that will monitor the network and publish information into the ontology, a Knowledge Plane that will reason about the available information to detect network anomalies and propose QoE optimizing actions and an Action Plane that is responsible for executing the QoE optimizing actions into the network.

The proposed ontology models both static information (e.g. the network topology and QoE optimizing capabilities of each node) and dynamic information (e.g. monitor values measured by the Monitor Plane). We described how the SWRL rule language can be used to reason on the available knowledge and insert new information. We illustrated, amongst others, some SWRL rules to autonomously define the hasNeighbour relationship between two network nodes and to autonomously achieve some kind of alarm raising where monitor values that can lead to QoE drops are detected.

Besides basic alarm raising, the ontology can also be used to enable real autonomic behavior where the right configuration of QoE optimizing actions is computed using SWRL rules. We presented a scenario and illustrated how SWRL rules can be used to find the best QoE optimizing actions for this scenario. We also illustrated how a more complex technique based on neural networks can be used if ontologies don't suffice.

The designed ontology focuses on the information present in the Monitor Plane and how this information can be used to detect network anomalies and propose new QoE optimizing actions. In related work, other ontologies have been proposed that concentrate more on a higher level view of network management where policies, users and other stakeholders are the main concepts. Our ontology focuses on the autonomic character of the network management. Using SWRL rules, the ontology can perform autonomic behavior of its own to improve the QoE of all services in the network. As these ontologies are complentary with ours, we foresee to combine them with our solution to provide a clearer view of today's broadband networks. Furthermore, we are targeting a closer integration with the architecture described in section III that is also able to formalise the decision making process.

Acknowledgment

This work is partly funded by the European Commission via the FP6 IP project Multi Service Access Everywhere (MUSE), phase II [21]. S. Latré would like to thank the Special Research Fund of Ghent University (BOF) for financial support through his Ph.D. grant. P. Simoens is funded by Ph.D grant of the Fund for Scientific Research, Flanders (FWO-V). F. De Turck is funded by a postdoctoral grant of the Fund for Scientific Research, Flanders (FWO-V).

References

[1] D. D. Clark, C. Partridge, C. J. Ramming, and J. T. Wroclawski, "A knowledge plane for the internet," in *SIGCOMM '03: Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*. New York, NY, USA: ACM Press, 2003, pp. 3–10.

[2] P. Simoens, B. De Vleeschauwer, W. Van de Meerssche, F. De Turck, B. Dhoedt, E. Gilon, K. Struyve, and T. Van Caenegem, "Towards Autonomic Access Networks for Service QoE Optimization," in *First IEEE International Workshop on Modelling Autonomic Communications Environments (MACE2006)*, 2006, pp. 223–233.

[3] B. De Vleeschauwer, W. Van de Meerssche, P. Simoens, F. De Turck, B. Dhoedt, P. Demeester, T. Van Caenegem, H. Dequeker, K. Struyve, E. Gilon, and E. Six, "Enabling Autonomic Access Network QoE Management through TCP Monitoring," in *1st IEEE Workshop on Autonomic Communications and Management (ACNM)*, May 2007.

[4] P. Simoens, B. De Vleeschauwer, W. Van de Meerssche, F. De Turck, B. Dhoedt, P. Demeester, T. Van Caeneghem, K. Struyve, H. Dequeker, and E. Gilon, "RTP Connection Monitoring for Enabling Autonomic Access network QoS Management," in *12th European Conference on Networks & Optical Communications (NOC)*, June 2007.

[5] B. De Vleeschauwer, W. Van de Meerssche, P. Simoens, F. De Turck, B. Dhoedt, P. Demeester, E. Gilon, K. Struyve, and T. Van Caenegem, "On the Enhancement of QoE for IPTV Services through Knowledge Plane Deployment," in *Broadband Europe*, 2006.

[6] E. Gilon, J. Hoet, H. Dequeker, R. Huysegems, E. Six, W. Van de Meerssche, P. Simoens, B. De Vleeschauwer, C. Pena, B. Nagel, and P. Vetter, "Service Rich Access Networks: The Service Plane Solution," in *BroadBand Europe*, 2007.

[7] B. De Vleeschauwer, P. Simoens, W. Van de Meerssche, S. Latré, F. De Turck, B. Dhoedt, P. Demeester, S. Van Den Berghe, and E. Gilon, "Autonomic QoE Optimization on the Access Node," in *Broadband Europe*, 2007.

[8] P. Simoens, S. Latré, B. De Vleeschauwer, W. Van de Meerssche, F. De Turck, B. Dhoedt, P. Demeester, S. Van Den Berghe, and E. Gilon, "Design of an autonomic QoE reasoner for improving access network performance," in *The Fourth International Conference on Autonomic and Autonomous Systems (ICAS)*, 2008.

[9] "OWL," OWL Web Ontology Language, W3C Recommendation, February 2004.

[10] "SWRL," A Semantic Web Rule Language Combining OWL and RuleML, W3C Member Submission 21 May 2004.

[11] T. Gu, H. Pung, and D. Zhang, "Toward an osgi-based infrastructure for context-aware applications," *IEEE Pervasive Computing*, vol. 3, no. 4, pp. 66–74, October 2004.

[12] D. Preuveneers, J. Van den Bergh, D. Wagelaar, A. Georges, P. Rigole, T. Clerckx, Y. Berbers, K. Coninx, V. Jonckers, and K. De Bosschere, "Towards an extensible context ontology for ambient intelligence," *Ambient Intelligence*, pp. 148–159, 2004.

[13] M. Eid, R. Liscano, and A. E. Saddik, "A novel ontology for sensor networks data," in *Proceedings of 2006 IEEE International Conference on Computational Intelligence for Measurement Systems and Applications,*, 2006, pp. 75–79.

[14] J. Li and S. Vuong, "Ontsum: A semantic query routing scheme in p2p networks based on concise ontology indexing," in *21st International Conference on Advanced Information Networking and Applications, 2007. AINA '07*, 2007, pp. 94–101.

[15] D. Xiao and H. Xu, "An integration of ontology-based and policy-based network management for automation," in *Computational Intelligence for Modelling, Control and Automation, 2006 and International Conference on Intelligent Agents, Web Technologies and Internet Commerce, International Conference on*, 2006, pp. 27–27.

[16] S. Davy, K. Barrett, M. Serrano, J. Strassner, B. Jennings, and S. van der Meer, "Policy interactions and management of traffic engineering services based on ontologies," in *Network Operations and Management Symposium, 2007. LANOMS 2007. Latin American*, 2007, pp. 95–105.

[17] IBM, "An Architectural Blueprint for Autonomic Computing," 2003.

[18] T. Gruber, "A translation approach to portable ontology specifications," *Knowledge Acquisition*, vol. 5, pp. 199 – 220, 1993.

[19] "Jena, a semantic web framework for java," [online] http://jena.sourceforge.net/.

[20] "Protégé-owl api," [online], http://protege.stanford.edu/plugins/owl/api/.

[21] "Multi Service Access Everywhere," [online] http://www.ist-muse.org.

IEEE ACNM 2008

2nd IEEE Workshop on Autonomic Communications and Network Management

11th April 2008 in Salvador, Brazil

In conjunction with:

NOMS 2008

IEEE/IFIP Network Operations and Management Symposium

Sponsors:

IEEE

IEEE COMMUNICATIONS SOCIETY

EU FP6 IST Project Autonomic Network Architecture (ANA)

Workshop Program

12:40 - 14:00 *Lunch Break*

14:00 - 15:40 Session 2:

A Functional Composition Framework for Autonomic Network Architectures
M. Sifalakis, A. Louca, A. Mauthe, M. Sifalakis, A. Louca, A. Mauthe

A Model for Designing Autonomic Components Guided by Condition-Action Policies
Gustavo A. L. de Campos, Ana Luiza B. de P. Barros, Jerffeson T. de Souza, Joaquim Celestino Junior

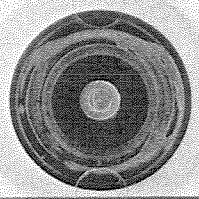
Design for a generic knowledge base for autonomic QoE optimization in multimedia access networks
Steven Latr´e, Pieter Simoens, Bart De Vleeschauwer, Wim Van de Meerssche, Filip De Turck, Bart Dhoedt and Piet Demeester, Steven Van Den Berghe, Edith Gilon - de Lumley

Next >>

2nd IEEE Workshop on Autonomic Communications and Network Management

IEEE

IEEE COMMUNICATIONS SOCIETY



IEEE Catalog Number: CFP0803E
ISBN: 978-1-4244-2067-4
Library of Congress: 2008900181

COMPACT disc DIGITAL DATA

ACNM 2008
Salvador, Brazil
April 11th, 2008