2009 Eighth International Conference on Networks

Effective and Resource-Efficient Multimedia Communication Using the NIProxy

Maarten Wijnants Wim Lamotte

Hasselt University and Interdisciplinary institute for BroadBand Technology (IBBT)
Expertise Centre for Digital Media and transnationale Universiteit Limburg
Wetenschapspark 2, BE-3590 Diepenbeek, Belgium
e-mail: {maarten.wijnants,wim.lamotte}@uhasselt.be

Abstract

Adequate and resource-efficient support for multimedia communication in transportation networks is rapidly gaining importance. At the same time, computer networks are being faced with increasing degrees of diversity, for instance in terms of client terminal specifications. We are therefore convinced that any next-generation network should be able to successfully address both these issues. This paper reports on our continued development of the NIProxy, a network intermediary which strives to improve the features and performance of IP-based networks with regard to multimedia communication in an attempt to optimize the experience provided to users of distributed applications. In particular, we describe how the NIProxy system was extended with support for upstream network traffic shaping as well as outbound multimedia service provision. The implications of these enhancements are discussed and we demonstrate and validate, through experimental evaluation, the new possibilities they entail and their added value in terms of user experience optimization.

1 Introduction

In recent years, user interest in networked multimedia services such as Voice over IP (VoIP) and real-time video streaming has risen substantially. Compared to traditional services like web browsing however, multimedia services are relatively complex and require high performance levels from the communication network. These unfortunately often cannot be guaranteed to be satisfied by the current version of the Internet. Furthermore, multimedia services also raise the need for network resource management, since they typically require considerable amounts of scarce network-related commodities (e.g. bandwidth). Effective resource management functionality is however still largely lacking in current-generation transportation networks. Another important trend in computer networking is the growing level of heterogeneity that is invading many of its different aspects. This heterogeneity is for instance exemplified by the continuous proliferation of new networking technologies and protocols and the high amount of diversity currently prevailing in the client device space.

As a result, it is our belief that any next-generation computer network should provide adequate, resource-efficient support for multimedia communication and at the same time should be able to successfully cope with heterogeneity. Rather than designing and deploying such next-generation networks from scratch, we are supporters of the approach in which the functionality of the current Internet is extended with next-gen features. Although the former solution might yield more optimal results, it also suffers from cost-efficiency issues since it discards not only the tremendous financial investments that have been made over the last decades in terms of Internet infrastructure, but also the practical experience and empirical know-how built up through years of extensive usage [4]. In the latter solution on the other hand, the existing infrastructure and know-how is largely reused, resulting in a minimization of the time and cost required to deploy the next-generation network.

This paper revolves around the NIProxy, a network intermediary which can be integrated in the Internet (and other IP-based networks) to enhance the network's functionality and performance with regard to multimedia services and applications [12]. The NIProxy is under continuous development and is consequently constantly evolving in terms of provided features and functionality. The contribution of this paper is a presentation of the most recent modifications and extensions made to the NIProxy system, which unlock two new important capabilities. First of all, they allow the NIProxy to consider not only the downstream but also the upstream direction when performing network traffic management and shaping. Secondly, whereas the application of computation on network flows was previously limited to flows destined for a NIProxy-connected client, processing and possibly even adaptation of flows emitted by clients is now also made possible. These two novel features are in addition implemented in an interoperable manner and, combined, they enable a myriad of new possibilities to improve the efficiency and effectiveness of distributed multimedia services and applications, as the presentation of representative experimental results will corroborate.

The outline for the remainder of this paper is as follows. Section 2 introduces the NIProxy network intermediary and highlights its objectives and adhered methodology. This section intertwines a summary of the NIProxy's feature list already reported on in previous work with a discussion of its novel capabilities and functionality focused on in this paper. The modifications to the NIProxy's software architecture which enabled these new features are reported on next in section 3, while section 4 presents a practical use case

978-0-7695-3552-4/09 $25.00 © 2009 IEEE
DOI 10.1109/ICN.2009.24

which exemplifies how the novel functionality could be exploited. The use case is subsequently employed in section 5 to investigate and evaluate the implications of the proposed NIProxy extensions and the novel possibilities they entail. Section 6 harbours a brief overview of related work. Finally, section 7 concludes the paper.

2 Network Intelligence Proxy (NIProxy)

2.1 Overview and Objectives

The Network Intelligence Proxy (NIProxy) [12] is a software entity that can be integrated at different locations in the topology of existing transportation networks and which strives to enhance the network's performance with regard to Quality of Experience (QoE) provision for users of multimedia services and applications[1]. As its name implies, it attempts to satisfy this objective by incorporating different types of *awareness* or *context* in the transportation network so that effective and efficient delivery of content to clients becomes possible. In particular, the NIProxy exploits its contextual knowledge to perform *network traffic shaping* as well as *multimedia service provision*. As will become apparent later on, these two mechanisms are complementary in terms of their QoE optimization possibilities. Furthermore, a key feature of the NIProxy is that it integrates its both QoE-increasing mechanisms in such a manner that they are able to collaborate, which in turn enables QoE optimization to an extent that could not be attained by applying the two mechanisms independently (see section 2.2.4).

2.2 Methodology and Approach

2.2.1 Enabling Context-Sensitive Networking

The NIProxy's contextual knowledge is twofold and comprises both network- and application-related information (see figure 1). The NIProxy's *network awareness* encompasses information regarding the state of the transportation network (i.e. the currently prevailing channel conditions), which is acquired through active network probing. Supplementary, the NIProxy monitors the bandwidth consumption of all network flows that pass through it, this way extending its network awareness with knowledge of the bandwidth requirements of individual network streams. The NIProxy's *application awareness* on the other hand is composed of knowledge of the networked application(s) clients are currently running and can hence vary significantly depending on the kind of application(s) under consideration. One example is the relative importance or significance assigned by the user to individual network flows (or flow aggregates, e.g. audio or video). Since its application awareness consists of

[1] Throughout this paper, we will use the term QoE to formally denote the user's experience and satisfaction; in contrast to Quality of Service (QoS), it is a rather subjective metric.

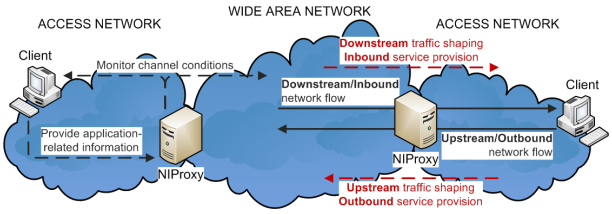


Figure 1. NIProxy operation: (left) Context compilation; (right) Downstream/inbound versus upstream/outbound.

application-specific information, the NIProxy is forced to rely on the application software to amass this type of context. In particular, the NIProxy expects the application to relay application-related information to it. To facilitate this process and to reduce the amount of modification required to the application software, a generic, reusable support library called the Network Intelligence Layer (NILayer) is provided [12]. The NILayer exports an API that allows developers to provide the NIProxy with context pertaining to their application with minimal effort.

2.2.2 Network Traffic Shaping

The NIProxy supports network traffic shaping, meaning it is capable of orchestrating the bandwidth consumption of networked applications. From a high-level point of view, the network traffic shaping mechanism exploits the network awareness amassed by the NIProxy to prevent the transportation network from being over-encumbered with data. At the same time, an intelligent allocation of the bandwidth capacity that is actually available is drawn up, grounded on the NIProxy's application awareness. In other words, the NIProxy's network traffic shaping mechanism strives to achieve resource-efficient multimedia networking by ensuring the available network resources are exploited as effectively as possible, i.e. in such a manner that the user's QoE is maximized. The actual results that the NIProxy will hereby be able to attain largely depend on the quantity as well as quality of the application-related information it has at its disposal.

Implementation-wise, the traffic shaping mechanism operates by organizing network flows in a *stream hierarchy*, a tree-like structure. Internal stream hierarchy nodes implement a certain bandwidth distribution strategy, whereas the leaf nodes correspond to actual network flows. Various types of internal nodes are available, each having distinct characteristics and capabilities. By adequately structuring the stream hierarchy, it is possible to express relationships between network flows or, conversely, to differentiate between them (or between ensembles of flows, e.g. audio versus video). As a result, constructing and maintaining the

stream hierarchy can be considered as an act of providing the NIProxy with application awareness and hence falls under the responsibility of the application software.

Once the stream hierarchy has been composed and presuming it is updated correctly over time, performing network traffic shaping simply amounts to allocating the correct amount of bandwidth to the hierarchy root node. The internal nodes, starting with the root node, will subsequently commence apportioning this bandwidth according to the particular bandwidth distribution technique each implements. Eventually, portions of the available bandwidth will reach one or more hierarchy leaf nodes, which will employ the bandwidth they are allocated to forward their associated network flow to its destination. An extensive description of the NIProxy's implementational approach to network traffic shaping, including a discussion of the operation of the different types of internal nodes, can be found in our previous work [13].

The NIProxy's network traffic shaping functionality was previously confined to the *downstream* flow direction, by which we refer to network traffic that is destined for a NIProxy-connected client (see figure 1). Thanks to the NIProxy extensions and architectural innovations reported on in this paper, the NIProxy is now however also capable of coordinating the *upstream* bandwidth consumption of networked applications. In other words, orchestration of network streams transmitted by the NIProxy client itself is now also supported. As will be discussed in more detail in section 3, this is achieved by maintaining, for each connected client, separate stream hierarchy instances that are respectively responsible for managing the client's downstream and upstream network traffic.

2.2.3 Multimedia Service Provision

Complementary to network traffic shaping, the NIProxy is also equipped with multimedia service provision functionality [12]. In other words, the NIProxy is able to exert services on network flows containing multimedia content on behalf of its connected clients. The NIProxy supports the notion of service chaining, meaning it is possible to consecutively apply different services on the same network flow. As a result, collaborative processing of a single network stream by multiple, independent services is enabled. Like the network traffic shaping mechanism, services can query and exploit the NIProxy's dual awareness. Thanks to its service provision mechanism, the NIProxy is able to address the heterogeneity issue identified in the introduction and the growing adaptability requirements of emerging networked applications that stem from it.

The NIProxy's multimedia service provision functionality is implemented using a *plug-in* approach [12]. Each service corresponds to a NIProxy plug-in that is (un)loaded dynamically as it becomes needed (obsolete). NIProxy services are consequently neatly separated not only from each other but also the remainder of the NIProxy's software architecture. This allows services to be either generic or application-specific. Generic services have a higher reusability factor and can be combined to rapidly compose complex and compound services, whereas application-aware services increase the applicability of the NIProxy by enabling it to cater to the specific requirements of individual applications. Another important advantage of the adhered plug-in approach is that it allows simple and rapid extension of the NIProxy's functionality, i.e. through the installation of additional plug-ins.

Whereas initially only *inbound* service provision was supported, it is now also possible to provide *outbound* services using the NIProxy. The distinction between inbound and outbound services is clarified in figure 1. With inbound services, we refer to the application of processing on network flows near the end of their passage through the transportation network (i.e. at a time the network flow has almost reached its final destination). Outbound services on the other hand are applied on network flows soon after they originated from the transmitter. The required architectural modifications that made the provision of outbound services possible will again be treated in detail in section 3; a discussion of the implementation of an example outbound NIProxy service is provided in section 4.

2.2.4 Improved User QoE Optimization through Interoperability

An important feature of the NIProxy is that interoperation between its both QoE-increasing mechanisms is supported. Instead of implementing the bandwidth management and service provision mechanisms as isolated entities, the NIProxy allows them to interact and collaborate with each other. This is true for both the downstream/inbound and upstream/outbound flow direction. It is for instance possible for NIProxy services to consult and even influence the bandwidth distribution strategies which the NIProxy draws up for clients (an example hereof will be presented in section 4). This in turn allows for the development of services having a very high level of performance as well as efficiency. In addition, as will be validated in section 5, supporting interoperation enables the NIProxy to elevate its user QoE optimization capabilities to a performance level that surpasses the results that can be achieved by applying the two mechanisms independently.

3 Software Architecture

As stated in the introduction, the NIProxy system is under constant development; the objective of this paper is to present recent updates made to its software architecture and

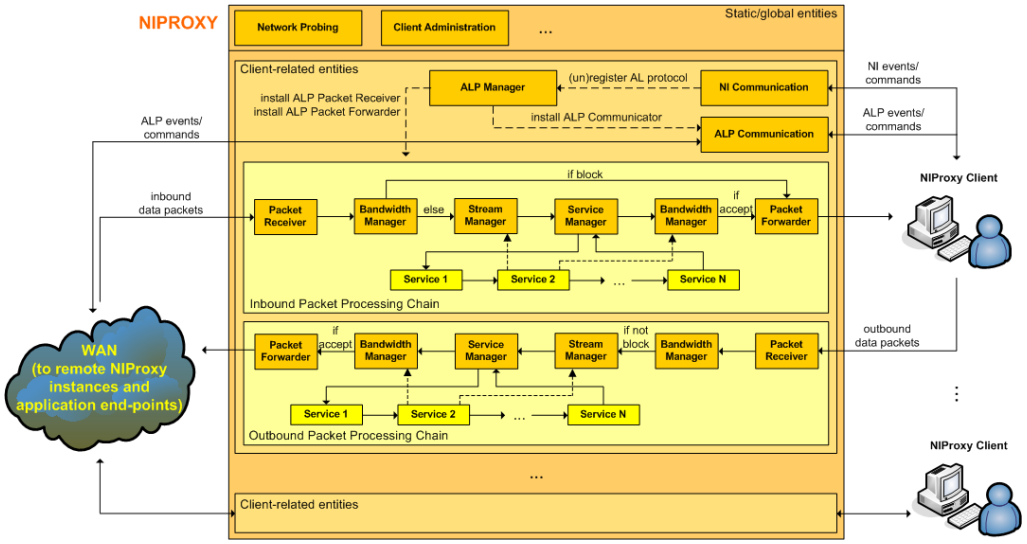


Figure 2. NIProxy software architecture.

to evaluate the novel capabilities and possibilities they enable. A schematic overview of the current software architecture is shown in figure 2. Compared to its original design, which was reported on in [12], the NIProxy's software architecture has undergone two major enhancements. First of all, to extend the NIProxy's practical applicability and usability, improved support for application-layer protocols (ALPs) was added to its software architecture. This addition is hence expected to have a positive effect on NIProxy adoption. Secondly, the NIProxy software architecture was extended with an outbound *packet processing chain* to enable upstream network traffic shaping as well as outbound multimedia service provision. Both enhancements are largely unrelated and space limitations unfortunately compel us to completely focus our attention on the latter topic.

3.1 Original Design

In its initial design, the NIProxy maintained a so-called packet processing chain (PPC) for each of its connected clients. Data packets intercepted from the Wide Area Network (WAN) needed to pass through this chain before they were possibly forwarded to their final destination (i.e. the corresponding NIProxy client) [12]. In other words, it performed downstream network traffic shaping and implemented what we have denoted in this paper as inbound service provision. This packet processing chain was retained, in its original capacity, in the NIProxy's current software architecture[2]. We therefore would like to refer the reader to our previous work [12] for a comprehensive discussion of the chain's exact mode of operation and refrain ourselves in this paper to a succinct review of the responsibilities of the main software components it links together. A first important constituting component is the *Bandwidth Manager*, which is responsible for managing the client's stream hierarchy (in this case the downstream variant) and for ensuring the bandwidth distributions inferred from it are enforced by the NIProxy. The *Stream Manager* on the other hand acts as the NIProxy's network awareness repository by recording the bandwidth consumption of network flows and storing all kinds of network-related information. Finally, the *Service Manager* is capable of loading and unloading NIProxy services on behalf of its associated client and applies the currently loaded services on intercepted data packets. In particular, whenever the Service Manager is handed a packet, it consecutively passes it to the loaded services that have registered interest for the network flow to which the packet belongs. Services are allowed to alter the contents of the packets they receive; if they do so, subsequent services will receive the altered version of the packet instead of the original. Note that, as indicated in figure 2 by the dashed arrows, NIProxy services have an interface to both the Bandwidth and Stream Manager components. It is this interface that enables the two QoE improving mechanisms currently provided by the NIProxy (i.e. network traffic shaping and multimedia service provision) to interoperate and collaborate.

3.2 Supporting Upstream/Outbound Operations

To translate the NIProxy's QoE optimization capabilities to the upstream/outbound direction, its software architecture was extended with an outbound packet processing chain which ought to be traversed by all data traffic generated by the NIProxy-connected client it is associated with.

[2] In figure 2, we have simply prefixed its name with the term "inbound" to differentiate it from the novel outbound packet processing chain which will be described in section 3.2.

As can be deduced from figure 2, the outbound packet processing chain mimics its inbound counterpart in terms of composing components as well as operating procedure. From an implementational perspective, a single entity is even used to represent the inbound and outbound variants of the Bandwidth Manager, Stream Manager and Service Manager components. In case of the Bandwidth Manager for instance, this entity simultaneously maintains the client's down- and upstream stream hierarchy and ensures the correct hierarchy is consulted depending on the direction of the data packets it is handed over. Similarly, the type of services applied by the Service Manager (i.e. inbound or outbound) is simply determined by the direction of the network flow to which the intercepted network packet belongs.

4 Use Case: Outbound Video Transcoding

In this section we present an example outbound service which will serve as practical use case to demonstrate and evaluate the NIProxy's upstream network traffic shaping and outbound multimedia service provision mechanisms. The service introduces outbound video transcoding functionality in the NIProxy and hence enables it to reduce the bitrate of outbound video streams[3]. This is achieved by decreasing the stream's temporal resolution and at the same time increasing its compression ratio (by using a larger step size for quantizing the transform coefficients), while leaving the spatial resolution of the video stream intact. The outcome is a video stream whose spatial resolution is identical to the original, but which is less fluid and has a lower visual quality. Implementation-wise, the service largely resembles the (inbound) video transcoding service presented in our previous work [12]. In this paper we will therefore not repeat all of the service's technical details but instead concentrate on its general mode of operation.

As input, the service expects outbound video streams at their original quality (i.e. as they are emitted by the video source). The output of the service is either this original version (OV) of the video stream or its transcoded variant (TV). To determine which video version to output, the service consults the upstream bandwidth distribution strategy calculated for the video source by the NIProxy. More specifically, the service exploits its interface with the Bandwidth Manager to access the video source's upstream stream hierarchy and to determine whether the hierarchy leaf node representing the transcoded version of the video stream is currently assigned upstream bandwidth. The decision whether to perform transcoding is hence dictated entirely by the NIProxy's network traffic shaping algorithm. As a result, unnecessary transcoding operations are eliminated, which is important since video transcoding is a computationally complex task consuming considerable amounts of scarce processing power.

As mentioned in section 2.2.2, the responsibility for composing and maintaining the stream hierarchy lies with the application software. This means the video source is responsible for ensuring the video streams it transmits are adequately incorporated in its upstream stream hierarchy. The transcoded version of these streams however do not originate from the video source itself but instead are generated on-the-fly by the outbound video transcoding service. The service consequently introduces a new type of outbound network flow, which should also be represented in the video source's upstream stream hierarchy. Therefore, for each distinct outbound video flow it is handed over, the service creates a new hierarchy leaf node, associates it with the transcoded variant of this flow and subsequently includes it in the upstream stream hierarchy as direct sibling of the leaf node corresponding to the original video version. Once the transcoded version of the outbound video flow has been incorporated in the stream hierarchy, the NIProxy's network traffic shaping algorithm will start considering it when managing the video source's upstream bandwidth capacity. Finally, besides updating the upstream stream hierarchy, the service also extends the NIProxy's network awareness by supplying the Stream Manager component with information regarding the bandwidth requirements of the transcoded version of outbound video flows.

5 Evaluation

This section harbours some representative experimental results which comprehensively demonstrate the added value of the NIProxy extensions proposed in this paper. In particular, the impact of the proposed extensions on the NIProxy's user QoE optimization capabilities will be evaluated by analyzing the results produced during two distinct experiments. We start this section however with a concise description of the setup in which the experiments were conducted.

5.1 Experimental Setup

To evaluate the conversion of the NIProxy's network traffic shaping and multimedia service provision functionality to the upstream/outbound flow direction, we simulated a simple multimedia streaming scenario involving a single server and possibly multiple clients. Based on incoming client requests, the server streamed multimedia data to clients using unicast. Besides the multimedia server and client(s), the experimental setup also contained a NIProxy instance which shaped the multimedia traffic emitted by the server. Finally, we also assumed the existence of an entity

[3]Due to our transcoder lacking the flexibility to enable transformation to a continuous range of values, video streams are in the current implementation always transcoded to a fixed percentage of their original bitrate. It is expected that supporting transcoding to arbitrary bitrates would enable the NIProxy to generate more dynamic and effective bandwidth distributions and hence to further improve its QoE optimization results.

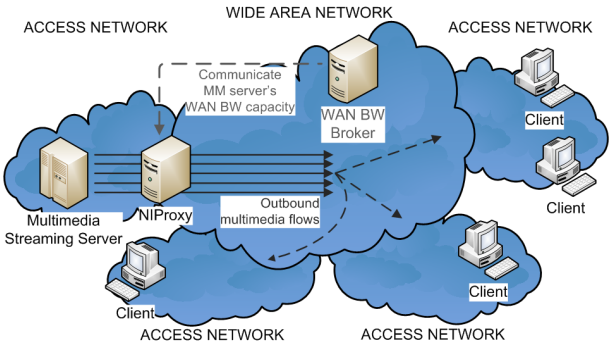


Figure 3. Experimental setup.

responsible for managing and allocating the bandwidth capacity of the network backbone, which we will refer to as the *WAN bandwidth broker*. In the experiments, the bandwidth broker hence determined the amount of backbone bandwidth the multimedia streaming server could maximally consume, for instance grounded on the Service Level Agreement (SLA) negotiated between the server and the operator of the WAN.

An overall picture of the experimental setup is provided in figure 3. By deploying the multimedia streaming server and the client(s) in separate access networks, we were able to guarantee that all of the server's outbound network traffic needed to pass through the WAN and hence consumed backbone bandwidth. Also notice that the NIProxy instance was located at the end of the server's access connection, this way enabling it to adapt and shape the server's outbound network traffic before it reached the WAN. As a final remark, the experimental setup could have easily been extended with NIProxy instances providing downstream bandwidth management and inbound service provision for the clients involved in the experiments. We however opted not to do so to ensure the reader's attention is focused on the NIProxy's novel functionality proposed in this paper and to allow intelligible and direct distillation of its impact from the generated experimental results. It should be apparent however that the extended setup would enable a number of additional possibilities in terms of user QoE optimization and would hence most likely be able to further improve the presented results.

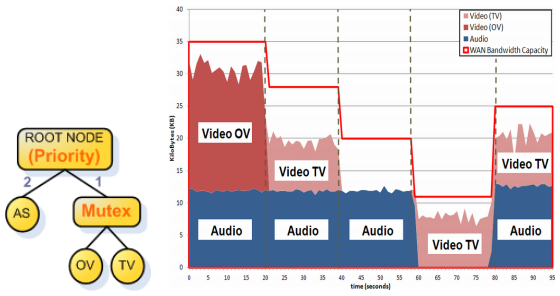
5.2 Experiment 1

In the first experiment, the multimedia server simultaneously streamed a video fragment and its accompanying audio stream to a single client. The NIProxy instance included in the experimental setup was hence responsible for ensuring the WAN bandwidth reserved for the server by the broker was apportioned intelligently among these two network flows. The server's WAN bandwidth capacity was in addition made subject to considerable fluctuations, which conceptually partitioned the experiment into a number of discrete intervals. By doing so, we were able to simulate a dynamic environment and investigate the NIProxy's performance in such a context. In practice, these fluctuations could for instance be caused by the arrival of new network flows that need to be accommodated by the WAN.

The (upstream) stream hierarchy which steered the NIProxy's decision making process during the experiment is illustrated in figure 4(a). Due to the constrained nature of the experiment, the stream hierarchy had a straightforward layout and contained only a limited number of nodes. In particular, the root of the hierarchy consisted of an internal node of type `Priority`[4] and directly distinguished the audio and video flow emitted by the multimedia server from each other. Since there was no notion of multiple audio versions in the experiment, the audio stream (AS) was made a direct child of the root as a hierarchy leaf node. The video flow on the other hand was available in two distinct qualities (i.e. the original version OV emitted by the multimedia server and a transcoded variant TV generated by the NIProxy's outbound video transcoding service). Since in this experiment it would be wasteful in terms of WAN bandwidth consumption if the client received both video flows simultaneously (the two video versions transport identical content and differ only in their quality parameters), their corresponding hierarchy leaf nodes were grouped together using a `Mutex` internal node [13] before they were added to the hierarchy root as a child. Finally, during the experiment's setup phase, the client indicated to the multimedia server it preferred audio over video. The server made the NIProxy aware of this application-related information by ensuring it was adequately captured in its upstream stream hierarchy. In particular, this was achieved by assigning a higher priority value to the leaf node representing the audio flow. Also note that the stream hierarchy did not change over time; only the bandwidth amount assigned to its root node was altered at the beginning of each new experiment interval (based on the bandwidth capacity information provided by the WAN bandwidth broker).

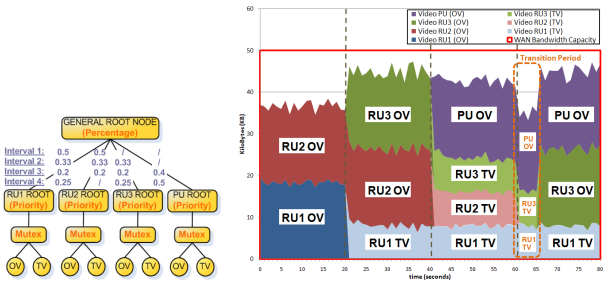
Based on the just described stream hierarchy, the NIProxy shaped the multimedia server's upstream bandwidth consumption as illustrated in figure 4(b). In this network trace, the dashed vertical lines separate the different experiment intervals. As can be seen, interval transitions always coincided with an alteration of the upstream bandwidth capacity available to the multimedia server. Also note from the trace that the original and transcoded versions of the video flow respectively required more and less bandwidth than the audio stream.

[4] A `Priority` node statically distributes the bandwidth it has at its disposal by first considering its child with the highest priority value; any excess bandwidth is subsequently allocated to the child with the next highest priority, etcetera [13].



(a) (b)

Figure 4. Experiment 1: (a) Upstream stream hierarchy; (b) Network trace (stacked graph) of the server's WAN bandwidth consumption.



(a) (b)

Figure 5. Experiment 2: (a) Upstream stream hierarchy; (b) WAN network trace.

Analysis of the network trace reveals that the WAN transmission of the audio flow was prioritized throughout the entire experiment. Only during the fourth experiment interval the audio stream was not forwarded over the WAN. This was however caused solely by a lack of adequate upstream bandwidth, not because precedence was given to the video flow. Any bandwidth left unused by the audio flow was subsequently employed to implement the video streaming. Only when insufficient WAN bandwidth was available to stream the original video version, the switch to the lower-quality transcoded version was made. As a result, the client always received the video flow at the highest quality possible, given the server's current upstream bandwidth capacity.

5.3 Experiment 2

In contrast to the first experiment, the second experiment involved multiple clients which each requested a particular video fragment from the multimedia streaming server. We decided not to include any other form of multimedia content in this experiment because doing so would merely have complicated the produced results without providing any additional insight in the NIProxy's capabilities. Also contrary to the first experiment, the multimedia server now disposed of a steady WAN bandwidth capacity (i.e. 50 KiloBytes per second). The experiment nonetheless included a dynamic aspect, this time caused by the arrival and departure of clients during experiment execution. This resulted not only in the initiation and suspension of outbound server flows at runtime, but also in shifts in individual stream importance. This latter effect is explained by the assumption that contracts existed between the multimedia streaming server and its clients, causing clients to be categorized as either *regular* or *premium* users. Compared to regular users, premium users should receive an improved service and a preferential treatment from the server, whenever possible.

The dynamic joining and leaving of clients again conceptually divided the experiment into a number of consecutive intervals. In particular, in the first interval the server's client list consisted of two regular users (RU1 and RU2). The second interval was initiated by the arrival of a third regular user (RU3), while the third interval commenced when a premium user (PU) requested a video fragment from the multimedia server. Finally, the transition to the fourth interval was triggered by the departure of regular user RU2. The multimedia server consequently needed to update its upstream stream hierarchy several times during the execution of the experiment. This stream hierarchy is depicted in figure 5(a) and can be considered as a generalization of the hierarchy used in the first experiment to a multi-client scenario. In particular, this time a root node of type `Percentage`[5] was used to discriminate not between the different network streams requested by a certain client but between the server's currently connected clients. Whenever a client joined the experiment, it was represented in the stream hierarchy according to the approach adhered to in the first experiment. In particular, on each new client connection, a subtree with a root node of type `Priority` was constructed and added to the general root node; all ensuing requests for network streams issued by the client were subsequently incorporated in its subtree as children of this `Priority` node[6]. Finally, figure 5(a) also illustrates the percentage values that were assigned to the involved clients during the different intervals of the experiment. These values were calculated by the multimedia server based on the contracts concluded with clients. More specifically, to favor them over regular users, premium users received a twice as high percentage value.

The server's WAN bandwidth consumption during the second experiment is depicted in figure 5(b). The bound-

[5] Associates a percentage value $p_i \in [0, 1]$ with each child; children are apportioned their corresponding percentage $p_i * BW$ of the bandwidth amount BW available to the `Percentage` node [13].

[6] Notice that since in this experiment all clients requested exactly one video fragment from the server, each intermediate `Priority` node had only a single child. As a result, the presence of these nodes in the stream hierarchy had in this case no influence on the shaping of the multimedia server's upstream network traffic.

aries of the experiment's different intervals are again indicated using dashed vertical lines. In the first interval, sufficient WAN bandwidth was available to stream to all currently connected clients (i.e. regular users RU1 and RU2) the video fragment they had requested at its original quality. This changed at the beginning of the second interval, when regular user RU3 joined the experiment and also issued a streaming request. The NIProxy therefore invoked the outbound video transcoding service for client RU1 and commenced forwarding the transcoded version of the video stream destined for RU1 over the WAN. Notice that since all clients involved in this phase of the experiment had an equal importance (i.e. it were all regular users), the NIProxy had to arbitrarily select a client to "penalize". In this case, this turned out to be RU1. The third interval was initiated by the arrival of premium user PU. This resulted in the downscaling of the video streams destined for the three regular clients to transcoded quality, this way freeing up sufficient WAN bandwidth to accommodate the original version of the video fragment requested by PU. Finally, the departure of regular user RU2 at the start of the fourth interval allowed the NIProxy to upgrade the video stream requested by RU3 back to original quality. The decision to benefit RU3 instead of RU1 was again made randomly. Also note the short period of time, indicated in the network trace using the dashed rectangle, during which the server's upstream bandwidth distribution was non-optimal (i.e. the available WAN bandwidth capacity was not exploited as completely and effectively as possible). This small transition period is explained by the fact that the NIProxy needed some time to ascertain that RU2 had actually left the experiment.

5.4 Discussion

A first important finding that can be deduced from the produced results is that the NIProxy-managed client (i.e. the multimedia streaming server) at all times respected its allocated WAN bandwidth capacity. In case this bandwidth capacity would have been determined by a SLA, possible financial repercussions associated with contract violation would hence have been avoided. In addition, by accurately enforcing the WAN bandwidth allocation devised by the bandwidth broker, the NIProxy contributed to WAN congestion prevention. As a result, the stability of the WAN was increased and its performance became more predictable, which in turn is likely to have had a positive influence on the experience provided to the users for which the upstream network traffic was destined. Secondly, besides ensuring the upstream bandwidth capacity was not exceeded, the NIProxy at the same time attempted to maximize its utility. In particular, guided by its application awareness, the NIProxy ensured the WAN bandwidth available to the multimedia streaming server was partitioned as intelligently and effectively as possible over its set of outbound network flows. In the first experiment for instance, the receiving user's preference for audio was reflected successfully in the server's upstream bandwidth distribution, whereas in the second experiment the NIProxy ensured the server's premium users received a preferential treatment. It should be apparent that this coordination and optimization of the upstream bandwidth consumption of the multimedia streaming server positively affected the QoE witnessed by the server's users. Finally, the presented experimental results also exemplify the ability for outbound multimedia services and the NIProxy's network traffic shaping mechanism to collaborate and the interesting capabilities this unlocks. In particular, the availability of the outbound video transcoding service improved the NIProxy's upstream network traffic shaping efficiency by enabling it to stream lower-quality video versions over the network backbone in the event of upstream bandwidth shortage.

6 Related Work

The NIProxy is a network intermediary which aims to extend current IP-based transportation networks with next-generation features and therefore provides network traffic shaping as well as multimedia service provision functionality. Both these techniques are topics of active research. Two of the initial explorers of the issue of network resource management, and client downstream bandwidth in particular, were Floyd and Jacobson [5]. More recently, Massoulié and Roberts studied the topic from a more mathematical point of view [7]. Work in the network traffic shaping context is however not limited to theoretical research. Interesting examples of concrete systems and frameworks include the Exact Bandwidth Distribution Scheme (X-BDS) [6] and the bandwidth sharing algorithm presented in [2]. The NIProxy's multimedia service provision mechanism on the other hand is largely related to the Service Oriented Architecture (SOA) paradigm and hence shares many of its underlying principles and resulting advantages [10]. Examples of platforms and frameworks adhering to this paradigm abound in the literature; see, for instance, the research by Klara Nahrstedt (e.g. [8]), the AWON architecture [3] and the Odyssey platform [9]. Finally, the NIProxy also shows substantial interfaces with architectures that are concerned with Quality of Service (QoS) provision like, for instance, the Congestion Manager architecture [1] and the OverQoS framework [11].

As a result, the NIProxy does not innovate in terms of the objectives it sets forth nor the techniques it employs to achieve them. What does distinguish the NIProxy from related research is that it integrates these techniques in a single system and in addition does so in an interoperable and collaboration-enabled manner. Also, the NIProxy's awareness includes application-related context, a type of knowledge that is often left unconsidered in other approaches.

7 Conclusions and Future Research

The majority of present-day distributed applications and services already involve the exchange of one or more types of multimedia content. Computer networks should in our opinion therefore be equipped with mechanisms and techniques that enable effective and resource-efficient multimedia communication. To this end, we have in this paper presented our continued work on the NIProxy, a network intermediary which enhances IP-based networks with network traffic shaping as well as multimedia service provision functionality. In particular, we have described how these two features were generalized so that they no longer only cover the downstream/inbound flow direction but instead can now also be applied on upstream/outbound network traffic. Besides discussing the architectural modifications that were required to achieve this generalization, we have also presented an example outbound NIProxy service which enables it to transcode video streams to a lower bitrate at an early stage of their traversal through the network (i.e. soon after they were emitted by the source). Finally, using this outbound video transcoding service as practical use case, we have experimentally evaluated the added value of the proposed NIProxy extensions. The produced experimental results clearly corroborate that the NIProxy delivers on its goal to improve the multimedia capabilities of IP-based networks and, by doing so, positively impacts the experience witnessed by users of networked multimedia applications.

As part of future work, we intend to pursue our efforts to improve and further extend the functionality provided by the NIProxy system. In the short term, we plan to more thoroughly investigate the possibilities yielded by outbound service provisioning through the implementation of a number of additional outbound NIProxy services. In the more distant future, we would like to experiment with a network setup encompassing multiple NIProxy instances to simultaneously control on the one hand the upstream bandwidth consumption of multimedia sources and on the other hand the last-mile downstream delivery of multimedia content to sinks. This would result in a near end-to-end solution in which only the (upstream) network connection between the multimedia source and its managing NIProxy instance is not covered. As previously mentioned in the evaluation section, it is expected that such an extended setup will be able to push the results in terms of user experience optimization to an even higher level.

Acknowledgments

This research is part of the IBBT E2E QoE project, funded by the Flemish government. Part of this research is also funded by the EFRD.

References

[1] D. Andersen, D. Bansal, D. Curtis, S. Seshan, and H. Balakrishnan. System Support for Bandwidth Management and Content Adaptation in Internet Applications. In *Proceedings of the 4th Symposium on Operating Systems Design and Implementation (OSDI 2000)*, pages 213–226, San Diego, California, USA, Oct 2000.

[2] F. M. Anjum and L. Tassiulas. Fair Bandwidth Sharing among Adaptive and Non-Adaptive Flows in the Internet. In *Proceedings of the IEEE Conference on Computer Communications (INFOCOM 1999)*, pages 1412–1420, New York, USA, March 1999.

[3] T. Banka, P. Lee, A. P. Jayasumana, and J. Kurose. An Architecture and a Programming Interface for Application-Aware Data Dissemination Using Overlay Networks. In *Proceedings of the 2nd IEEE International Conference on COMmunication System softWAre and MiddlewaRE (COMSWARE 2007)*, Bangalore, India, Jan 2007.

[4] C. Dovrolis. What Would Darwin Think about Clean-Slate Architectures? *ACM SIGCOMM Computer Communication Review*, 38(1):29–34, Jan 2008.

[5] S. Floyd and V. Jacobson. Link-sharing and Resource Management Models for Packet Networks. *IEEE/ACM Transactions on Networking*, 3(4):365–386, Aug 1995.

[6] V. Hnatyshin and A. S. Sethi. Architecture for Dynamic and Fair Distribution of Bandwidth. *International Journal of Network Management*, 16(5):317–336, Sept-Oct 2006.

[7] L. Massoulié and J. Roberts. Bandwidth Sharing: Objectives and Algorithms. *IEEE/ACM Transactions on Networking*, 10(3):320–328, June 2002.

[8] K. Nahrstedt, B. Yu, J. Liang, and Y. Cui. Hourglass Multimedia Content and Service Composition Framework for Smart Room Environments. *Elsevier Journal on Pervasive and Mobile Computing*, 1(1):43–75, March 2005.

[9] B. D. Noble, M. Satyanarayanan, D. Narayanan, J. E. Tilton, J. Flinn, and K. R. Walker. Agile Application-Aware Adaptation for Mobility. In *Proceedings of the 16th ACM Symposium on Operating Systems Principles (SOSP 1997)*, pages 276–287, Saint-Malo, France, Oct 1997.

[10] Service Oriented Architecture. `http://www.serviceoriented.org/service_oriented_architecture.html`, Oct 2008.

[11] L. Subramanian, I. Stoica, H. Balakrishnan, and R. Katz. OverQoS: An Overlay based Architecture for Enhancing Internet QoS. In *Proceedings of the 1st Symposium on Networked Systems Design and Implementation (NSDI 2004)*, pages 71–84, San Francisco, California, USA, March 2004.

[12] M. Wijnants and W. Lamotte. The NIProxy: a Flexible Proxy Server Supporting Client Bandwidth Management and Multimedia Service Provision. In *Proceedings of the IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM 2007)*, Helsinki, Finland, June 2007.

[13] M. Wijnants and W. Lamotte. Managing Client Bandwidth in the Presence of Both Real-Time and non Real-Time Network Traffic. In *Proceedings of the 3rd IEEE International Conference on COMmunication System softWAre and MiddlewaRE (COMSWARE 2008)*, Bangalore, India, Jan 2008.