An MPEG-4 Compliant Interactive Multimedia Streaming Platform Using Overlay Networks

Hyun-Cheol Kim, Charalampos Z. Patrikakis, Nikos Minogiannis,
Pantelis N. Karamolegkos, Alex Lambiris, and Kyuheon Kim

This paper presents a multimedia streaming platform for efficiently transmitting MPEG-4 content over IP networks. The platform includes an MPEG-4 compliant streaming server and client, supporting object-based representation of multimedia scenes, interactivity, and advanced encoding profiles defined by the ISO standard. For scalability purposes, we employ an application-layer multicast scheme for media transmission using overlay networks. The overlay network, governed by the central entity of the network distribution manager, is dynamically deployed according to a set of pre-defined criteria. The overlay network supports both broadcast delivery and video-on-demand content. The multimedia streaming platform is standards-compliant and utilizes widespread multimedia protocols such as MPEG-4, real-time transport protocol, real-time transport control protocol, and real-time streaming protocol. The design of the overlay network was architected with the goal of transparency to both the streaming server and the client. As a result, many commercial implementations that use industry-standard protocols can be plugged into the architecture relatively painlessly and can enjoy the benefits of the platform.

Keywords: Interactivity, streaming, MPEG-4.

Manuscript received Sept. 02, 2005; revised Mar. 19, 2006.

This work was supported by the Ministry of Science & Technology of Korea under the title of "Development of Interactive Scalable Multimedia Streaming Platform."

Hyun-Cheol Kim (phone: + 82 42 860 3984, email: kimhc@etri.re.kr) is with Radio & Broadcasting Research Division, ETRI, Daejeon, Korea.

Charalampos Z. Patrikakis (email: bpatr@telecom.ntua.gr), Nikos Minogiannis (email: minogian@telecom.ntua.gr), Pantelis N. Karamolegkos (email: karamolegos@telecom.ntua.gr), Alex Lambiris (email: biril@telecom.ntua.gr) are with Electrical Engineering and Computer Science Department, NTUA, Athens, Greece.

Kyuheon Kim (email: KyuheonKim@khu.ac.kr) is with College of Electronic & Information, Kyunghee University, Gyeonggi-do, Korea.

I. Introduction

The development of multimedia technologies over the past years has been enabling users to access multimedia content through various types of transmission media. At the same time, it has made the use of multimedia an indispensable part of everyday communication. One of the next targets for multimedia technology is interactivity in content access such as interactive TV [1]. One of the most important current international standards for multimedia delivery is Moving Picture Experts Group's MPEG-4 [2]. Authoring technology for interactive contents based on MPEG-4 has been developed to the state that interactive content can be easily created through the use of graphical user interfaces [3]. Furthermore, MPEG-4 technology is expected to be used in various fields such as digital multimedia broadcasting (DMB) [4]. Full adoption of the MPEG-4 standard has been delayed due to the advanced features it provides, such as object-based representation of multimedia scenes, user interactivity, and advanced coding techniques, that surpass conventional methods based solely on video and audio transmission.

In this paper, we present an interactive multimedia streaming platform that supports the deployment of an MPEG-4 compliant solution over IP networks. The platform described utilizes an overlay network for scalability and incorporates QoS techniques for further enhancing the user experience.

The rest of this paper is organized as follows: In section II, we provide an overview of the media-related technologies and protocols used in our platform. The design of the proposed platform is described in section III, where the components that constitute the overall platform together with the protocols deployed for media distribution over the network are presented

in detail. Section IV presents the QoS mechanism integrated in the delivery network and its use in conjunction with MPEG-4 content. Specific scenarios and evaluation tests of the platform are depicted in section V. Finally, in the last section of the paper, conclusions and recommendations for future work towards enhancement of the platform are presented.

II. MPEG-4

MPEG-4 is an ISO/IEC standard developed by the Moving Pictures Experts Group, covering the areas of production, distribution, and content access of multimedia interactive applications.

Audiovisual scenes in MPEG-4 are composed of media objects, organized in a hierarchical fashion. Examples of primitive media objects are still images, sound, video, and so on. Primitive objects can be combined to form a scene. Organizing a representation with objects allows reusability and offers content authors greater flexibility. In addition, MPEG-4 provides the means for interacting with the composed scene (for example, pressing buttons, moving objects, or triggering event sequences) enhancing the overall user experience.

The MPEG-4 standard is divided into various parts. The parts used in the presented streaming system are parts 1, 2, 3 and 6. Part 1 (systems) covers the mechanisms for scene composition and object interactivity, parts 2 and 3 dictate the encoding/decoding details of visual and aural objects, respectively, and finally part 6, the delivery multimedia integration framework (DMIF), specifies the delivery framework used for conveying MPEG-4 streams.

1. MPEG-4 Part 1 – Systems

The coded representation of an MPEG-4 object is carried over a data stream, referred to as an elementary stream. For scene composition, the elementary streams that hold the relevant object descriptions should be retrieved and combined so as to create the actual presentation on the receiver side.

The systems part of the MPEG-4 standard [2] defines the binary format for scenes (BIFS) and the object descriptors (ODs) that describe the relationships that can be applied between objects. They also provide the means to build an object hierarchy and ultimately describe an audiovisual scene.

BIFS, as the name suggests, is a binary format used to provide scene descriptions. A scene description in MPEG-4 includes the spatial and temporal placement of objects and their ability to interact with end-users.

The ODs define the relationship between an object and the elementary streams needed to represent it. An OD may provide information such as the URL needed to access the elementary streams, the required decoder characteristics, intellectual property, and others.

2. MPEG-4 Part 6 – DMIF

Part 6 of the MPEG-4 standard [5] describes the delivery multimedia integration framework (DMIF). The DMIF is a session protocol for the management of multimedia streaming over generic delivery technologies.

The DMIF architecture aims at abstracting the communication layer from the MPEG-4 application. The application is presented with an interface and does not need to be concerned with how the interface calls will be actually implemented over the delivery channel. Figure 1 represents this concept.

In Fig. 1, the MPEG-4 application accesses data through the DMIF-application interface (DAI), irrespective of whether such data comes from a broadcast source, local storage, or from a remote server. All communication is performed through the DAI. Transparently to the application, a different DMIF instance is invoked for each of the three different sources taking care of the peculiarities of the involved delivery technology.

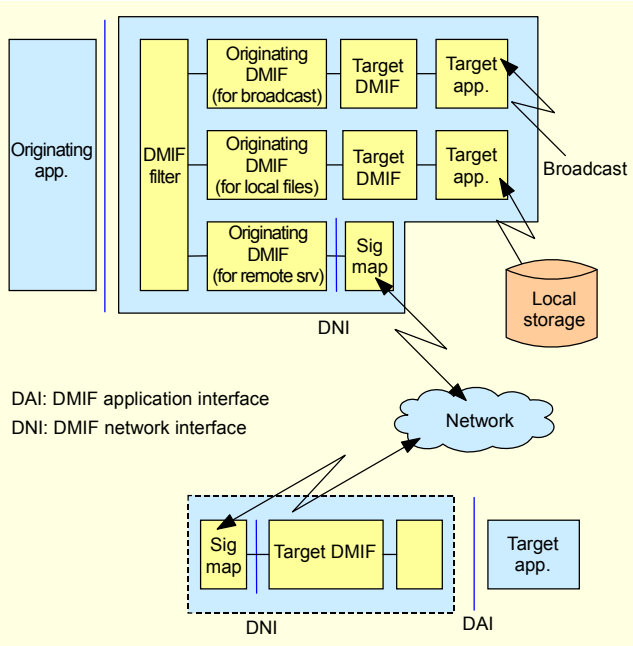


Fig. 1. DMIF communication architecture.

III. Design of the Proposed Streaming Platform

In the following sections, we will present an interactive multimedia streaming platform for MPEG-4 content that has been designed and implemented in the context of the international research project, ISMuS. The platform components include a streaming server, an MPEG-4 client, and overlay

network modules: the relay node and the network distribution manager. Administration tasks and user access are provided through a web-interface. The streaming platform was designed to support remote interactivity, taking advantage of the functionality described in the systems part of the MPEG-4 standard.

In order to reduce network congestion and avoid burdening the streaming servers with a large number of clients, the platform deploys an overlay network between the streaming server, relay nodes, and clients. The overlay network was designed to be transparent to the server and client and does not impose any specific changes in their design.

Figure 2 shows a conceptual overlay network. The proposed streaming platform targets two services: on-demand and broadcast-like. In the on-demand service, users access MPEG-4 content through a video-on-demand (VoD) approach. For the broadcast-like service, content is available either live or in a near-VoD (also known as simulated live) approach. The on-demand scenario supports full MPEG-4 interactivity.

In the following paragraphs, we briefly present existing solutions, explain some of the technical challenges faced, and provide the complete design of our platform.

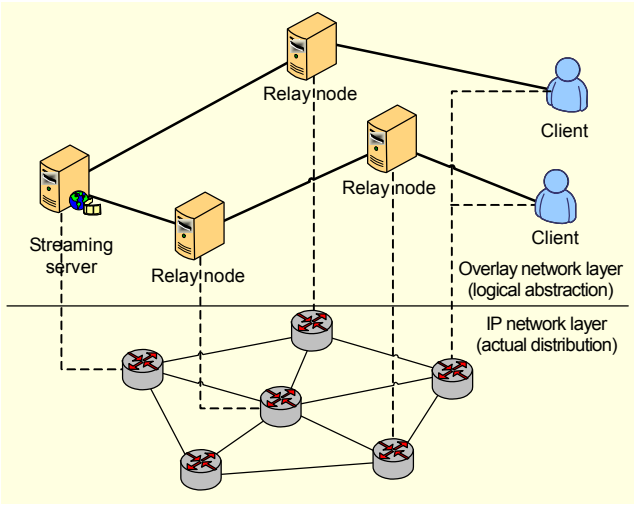


Fig. 2. Deployment of an overlay network.

1. Existing Solutions

The need for an efficient and common standard for offering media distribution services has led to the wide adoption of MPEG-4 technologies in many commercial and free/open products. However, the sheer complexity of the standard and the wide areas it tries to cover has prevented vendors from offering fully-compliant solutions. Most implementations adhere only to parts dictating audio and video encoding and largely ignore the systems and delivery parts. This, in effect, prevents us from exploiting the advanced capabilities (for example, interactivity) MPEG-4 has to offer.

The most popular MPEG-4 compliant solutions are Apple's QuickTime Streaming Server [6] and RealNetworks' Helix [7]. Apple and RealNetworks offer full streaming platforms with MPEG-4 compatibility (but with the aforementioned limitations). Our design effort was to be interoperable with these solutions and to further utilize the advanced features of the MPEG-4 standard.

2. Design Challenges

As mentioned before, one of the primary goals of our platform was to be interoperable and transparent to existing MPEG-4 solutions. For achieving this goal, we chose the industry-standard real-time transport protocol (RTP) and real-time transport control protocol (RTCP) [8] for data delivery, and real-time streaming protocol (RTSP) [9] for session control.

A. Packetization with RTP/RTCP

Packetization of MPEG-4 elementary streams is performed according to the RTP and more specifically according to RFC3640 [10], in which a generic RTP payload format for the transport of non-multiplexed MPEG-4 elementary streams is defined. The RTP provides the necessary facilities for adding flow/congestion control through the RTCP.

The RTCP is the companion of RTP and specifies the periodic transmission of control packets to session participants in order to provide feedback on the quality of the data distribution. RTCP reports include packet loss, inter-arrival jitter and round-trip time metrics. By inspecting the reports, a streaming server or a relay node can identify transmission problems and adequately adapt the delivery strategy to overcome them.

B. Mapping between RTSP and DMIF

The MPEG-4 standard includes the DMIF protocol for session control. Although the DMIF provides an extensive framework for configuring and controlling content delivery, it does not actually specify the way its functions translate over the native network layer, and remains largely application-oriented rather than network-oriented. In order to bridge the gap between the two layers, application and network, a network-oriented protocol is required. This is the role of the RTSP.

The RTSP is a session setup protocol; it negotiates and configures the parameters needed to deliver a series of streams over a network, acting as a 'network remote control' for multimedia services. The actual delivery is usually carried out-of band and in most cases is handled by the RTP. The RTSP command set includes commands such as DESCRIBE (for

obtaining media presentations and stream information), SETUP (for negotiating transport parameters), PLAY, and TEARDOWN (for playback control).

Although in most scenarios this basic set is sufficient, in order to accommodate the DMIF, more sophisticated mechanisms are required. The RTSP provides the GET_PARAMETER and SET_PARAMETER commands for passing arbitrary parameters and custom commands. These commands allow for a more fine-grained mapping from the DMIF to RTSP while remaining compliant with other RTSP implementations that simply ignore custom configuration.

3. Streaming Server

The proposed streaming server was designed to support two kinds of services: on-demand and broadcast-like. The streaming server architecture is shown in Fig. 3.

The main components of the server architecture are the server and the network manager. The two components correspond to the basic parts defined in the MPEG-4 standard, the sync layer (where time-stamping and synchronization tasks take place) and the delivery layer (where the actual delivery of the data takes place). The interface between the two modules is the MPEG-4-defined DAI.

The main function of the server manager is to parse the requested MPEG-4 content and generate sync layer packets for transmission.

The network manager is in charge of actually transmitting the content by managing client sessions, negotiating network parameters, packaging and transmitting data, and monitoring quality. The functions of the network manager are implemented with the assistance of the RTP, RTCP and RTSP protocols and through the appropriate mapping to the DMIF and MPEG-4 specifications as outlined in the previous paragraphs. The clear separation of the server and the network manager with a well-defined interface (that is, the DAI) allows the future introduction of other network protocols, for example the datagram congestion control protocol (DCCP) [23], without disrupting the main design of the server.

The aforementioned architecture is common for both on-demand and broadcast-like services. In addition, for the broadcast-like service, another component, the broadcast schedule manager, is deployed to allow programming of streams for scheduled transmission. Programming is done either locally via the server GUI or remotely through the administrative web-interface of the platform.

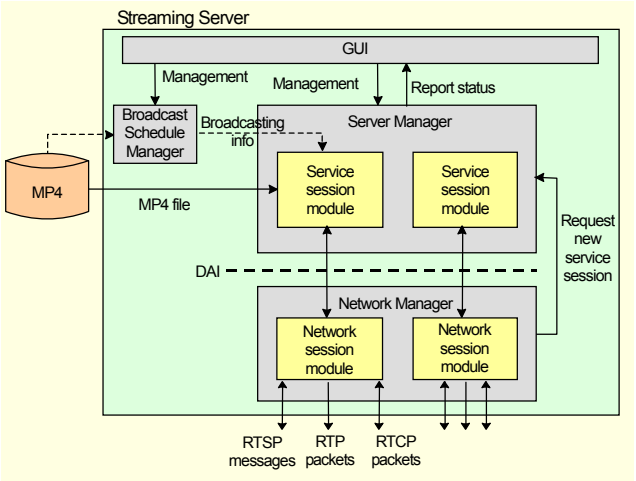


Fig. 3. The architecture of the streaming server.

4. Client

The client components are displayed in Fig. 4. Similarly with the server, there are two main modules, the contents manager and the network manager. The interface between them is defined by the DAI.

The contents manager handles de-packetization, decoding, and presentation. Its implementation is based on the MPEG-4 reference software, which incorporates flexible techniques to easily accommodate media decoders in a plug-in architecture. Currently supported media formats include MPEG-4 advanced video coding (AVC) video [11], MPEG-4 bit sliced arithmetic coding (BSAC) audio [12], and the JPEG standard for images. The plug-in architecture allows the current list to be expanded if the need for another format arises.

The network manager exhibits similar functionality to the

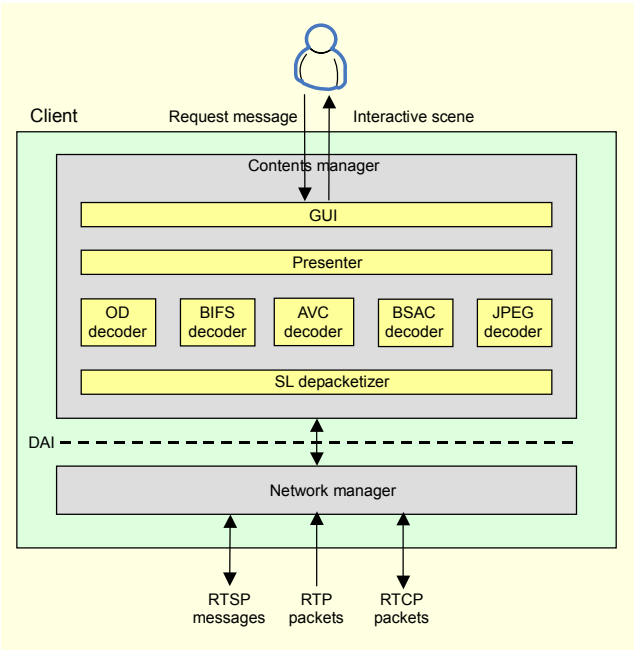


Fig. 4. The architecture of the client.

respective server component, that is, it handles all network operations through the use of the RTP, RTCP and RTSP. Still, future expansion through the use of alternative network protocols is possible thanks to the clear separation of the design components.

5. Overlay Network

The platform utilizes an overlay network to handle the task of conveying content from the media servers down to the client terminals. The overlay architecture is comprised of two different modules: the relay node and the network distribution manager (NDM). The approach we follow is similar to that of content delivery networks (CDNs) [13]: content is scattered to a series of servers, the relay nodes, and a central management unit (much like the DNS mechanism used in CDNs), in our case the NDM, is used to redirect clients to the 'closest' server holding the requested content. The meaning of the 'closest' or 'best' connection point in the architecture is determined by evaluating a series of criteria/parameters (see the following).

The overlay network can perform in two modes: broadcast-like and on-demand.

A. Broadcast-like Service

In this service, the distribution of media files is performed in real-time or in a near-VoD fashion. Content transmission is pre-scheduled and clients can connect anytime between the scheduled hours, much like watching a TV program. The following is a description of the operation of the two overlay modules.

• *Relay Node*

The relay node is the core component of the platform when it comes to media redistribution. At the lowest level, a relay node is both a proxy and a stream splitter. It forwards incoming media streams to one or more clients.

A series of relay nodes can co-operate, forming tree-like structures like those depicted in Fig. 5.

The relay nodes communicate with higher and lower nodes in the distribution hierarchy through the use of the RTSP. In terms of functionality, a relay node has a double interface: one for setting up incoming streams and one for serving outgoing streams. In the first case, the relay node behaves like an RTSP client. In the second case, it acts like an RTSP server.

By using a combination of manual and automatic configurations, a series of relay nodes can effectively distribute content to multiple users minimizing network and hardware resources. The configuration of the relay nodes is handled by the overlay network managing entity, the NDM.

• *NDM*

The NDM maintains a small database of all available content

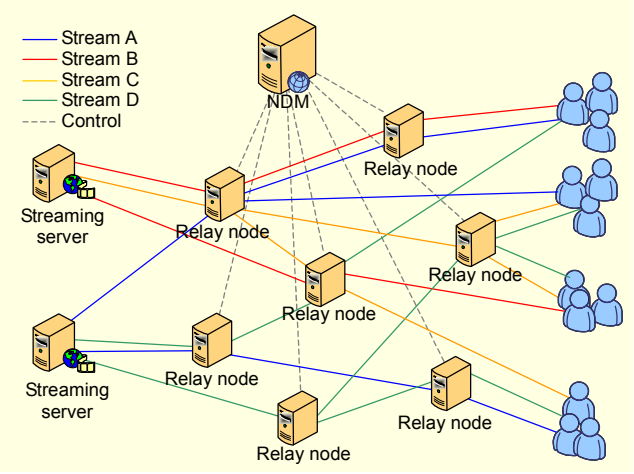


Fig. 5. Content distribution for the broadcast-like service.

and the location (that is, the streaming server) where it is stored. The NDM constantly updates its information from the streaming servers and the relay nodes. The NDM offers a central interface to the platform administrators to allow them to control relay nodes and, if needed, manually build the distribution trees from the streaming server to end-users. The information available in the NDM is not only limited to content availability but also includes network statistics (packet loss, jitter) and performance metrics (such as processing load and memory usage) obtained and periodically refreshed from the relay nodes.

Having a complete picture of the distribution network, the NDM can automatically suggest a connection point for clients through the end-user web-interface. The NDM also provides its overview knowledge of the distribution network to the relay nodes when necessary to allow them to dynamically configure themselves and automatically participate in distribution trees for offering content.

The relay nodes maintain a permanent communication channel with the NDM. Through this channel, they are able to upload QoS measurements as well as other information such as local resource availability (for example, processor load, memory usage, and network fan-out). More important, whenever a request for a locally unavailable stream arrives, they can consult the NDM in order to retrieve the requested resource and dynamically complement the distribution tree. In this case, the NDM returns a list of candidate connection points and leaves the ultimate choice to the relay node. The NDM-provided list not only contains the location (that is, a streaming server or another relay node) where the stream can be acquired, but also includes additional information such as stream quality metrics and resource availability. The relay node evaluates the provided information and chooses a connection point based on its overlay profile.

The overlay profile is a formula that can be independently configured for each relay node and dictates the weight each NDM-provided parameter (for example, server location, packet loss, jitter, processor load, fan-out, and so on) should have in making a decision. Theoretically, the formula has a form of $w_1 \times p_1 + w_2 \times p_2 + w_3 \times p_3 + \cdots$. A practical approach is to evaluate each of the n parameters on a scale of 0 to 10 (the higher, the better) and use powers of 10 as weights for ordering them. In that case, the formula would be like $10^{n-1} \times p_{n-1} + 10^{n-2} \times p_{n-2} + \cdots + 10 \times p_1 + p_0$.

By carefully configuring the overlay profile and with a mix of manual and dynamic configurations, efficient distribution can be accomplished.

In deciding on the best connection point, a relay node has the following parameters available.

- *Locality*: Each relay node reports to the NDM its geographical location (for example, a relay node in Greece could report NTUA/Athens/Greece/Europe and a relay node in Daejeon would report ETRI/Daejeon/Korea/Asia). This information can be used to quickly dismiss remote nodes and limit choices to nearby servers only.
- *Stream Availability*: This denotes whether the requested stream is actually available at the candidate connection point. Choosing a relay point that does not currently hold a stream (but satisfies another criterion such as *Locality*) will trigger the whole decision procedure at least once more. This can be useful when many clients connect from the same area and a local relay node is available, although it does not at the current time relay the requested stream. In that case, the first client will experience a longer delay until served (since the chosen relay node will actually have to receive and relay the stream from somewhere else), but the rest of them will be served immediately.
- *Stream Quality Metrics*: These are extracted by the RTCP messages. RTCP-measured quantities include packet loss, inter-arrival jitter, and round-trip propagation delay.
- *Resource Availability*: This parameter is calculated periodically at each relay node and is reported to the NDM, which provides it directly to the interested clients. It takes under consideration processor load, memory usage, and bandwidth consumption. Resource availability metrics can be used for performing load balancing over the distribution network.
- *Network Proximity and Bandwidth Estimation*: This is calculated through client performed experiments. Using a ping-like mechanism, round-trip time is measured and, in the paradigm of the packet-pair technique, an estimation of the available bandwidth can be obtained.

By ordering and/or omitting parameters, different scenarios can be deployed. For example a 'Fast Setup' scenario would rank *Stream Availability* higher than any other parameter and could then consider *Stream Quality* and *Resource Availability* as second-level criteria. An 'Optimum Quality' scenario would rank higher *Locality* and experiment-related parameters such *Network Proximity and Bandwidth Estimation*.

In section V, we will present some examples of how the overlay parameters can be put into action.

B. On-Demand Service

In this service, the user is able to access an MPEG-4 file using a VoD approach. Obviously, for the VoD service, we cannot use the same techniques used for the broadcast scenario. Since content transmission is not scheduled and each client can access any file at any time, we cannot use the same stream, replicated on-the-fly at the relay nodes for serving large numbers of clients. What we can do is mirror popular files and distribute them across the delivery network.

Streaming VoD content requires the presence of a streaming server. Next to it (that is, on the same PC), we deploy a relay node. Relay nodes functioning in VoD mode register with the NDM, as in the broadcast scenario, and periodically report their resources (processor usage, memory, fan-out, and so on) in the same way. By using an FTP-like protocol, relay nodes can download content and make it available to their adjacent streaming server.

This scenario, compared to the broadcast service scenario, is closer to the content delivery network (CDN) approach. However, we can still take advantage of the infrastructure of the overlay network and offer more than a simple DNS lookup mechanism for determining the serving point for a client. Since the relay nodes are connected to the NDM, the decision on the connection point for a client can use almost the same criteria used for the broadcast service. The NDM is equipped with an

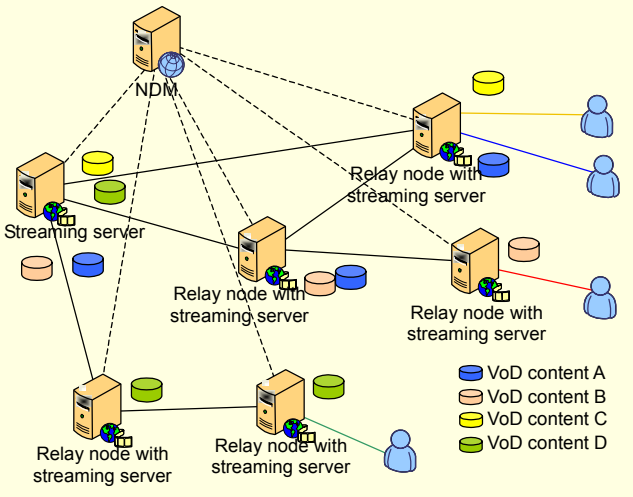


Fig. 6. Content distribution for the on-demand service.

on-demand profile that evaluates available options and redirects the client to an optimal streaming server.

IV. Using the Overlay Network for Supporting QoS

In the broadcast-like service, the overlay network has a very active role in the distribution process. The relay nodes do not limit themselves to just copying and retransmitting packets but, in addition, they actively monitor the transmission quality of the outgoing flows. For further enhancing the user experience, we supplied the relay nodes with the ability to intervene whenever packet loss is observed.

To be able to interfere with transmission, the relay nodes include all the necessary functionality that allows them to act as a miniature streaming server and an end-user application at the same time. They can analyze incoming streams, distinguish video from audio, calculate the required bandwidth, and so on. The relay node functions as a client application for incoming streams (minus the reproduction part) and a streaming server for outgoing streams (without the synchronization, multiplexing, and packet encapsulation parts).

By understanding the characteristics of an incoming flow, a relay node can alter the outgoing streams to face congestion and information loss. Altering the characteristics of an outgoing stream happens on a per-stream basis. In this way, the relay node is able to produce many versions out of one original flow, each one adapted to the specific requirements of a downward receiver. The adaptation process happens exclusively on the relay node and without the need of interference from the original streaming server. Any additional information needed is provided by the NDM. In fact, the platform does not require the streaming servers to distinguish relay nodes from plain client applications. In this way, we can introduce to the platform other (for example, commercial) streaming servers without the need of adapting them first to the platform and still maintain QoS benefits.

Differentiating outgoing streams can take many forms. For example, in a DiffServ-enabled network, the relay node can mark outgoing packets after consulting the NDM about the client/receiver profile.

In the proposed architecture, differentiating outgoing streams is accomplished by altering bandwidth. The relay node can adjust the bandwidth required by a flow and adapt it to the requirements of each downward client. The ability to adjust bandwidth allows the servicing of clients with different network equipment and capabilities, and also allows dynamically facing network congestion and packet loss.

In the following paragraphs, we present the algorithm for dynamically adapting bandwidth, as well as the simulation results and MPEG-4 specific adaptation to accommodate object-based representation. The algorithm was first presented in [15].

1. Rate-Adaptation Algorithm

We assume that a relay node for a given incoming flow is able to produce n variants of the flow, each variant having different (and lesser) bandwidth requirements from the original flow. The way a relay node produces these variants is explained in section IV.3, which deals with the actual MPEG-4 specific implementation.

Clients are initially served from the original flow, which obviously carries the best possible stream quality. Whenever a client suffers packet loss, the relay node dynamically switches the appropriate outgoing stream to a less bandwidth-demanding variant until no packet loss is observed. If network congestion is temporary, a client can try switching back to a better and more bandwidth-demanding variant. Packet loss information is conveyed to the relay node by means of RTCP reports.

We define the process of analyzing stream quality (conveyed from the RTCP reports) for a specific time interval as 'quality control'. Also, we define the process of moving a client to a higher (in bandwidth) variant as a 'switch-up' and the process of moving a client to a lower variant as a 'switch-down'.

We define time interval I. This interval is the frequency between quality controls. Also, we define two limits, U and L. These limits represent the percentage of packet loss a client can suffer. The U limit defines the maximum packet loss a client can suffer before a switch-up can be attempted. In practice, the U value is usually zero, or close to zero, meaning switching up is an option only for clients with minimal or no packet losses at all.

The L limit defines the minimum packet loss a client can suffer before a switch-down should be attempted. Practical values for L vary between 0.05 and 0.10, meaning packet losses between 5% and 10%.

Each time a quality control takes place in a relay node, the mean packet loss of each client is compared to U and L. We use two different percentages, the short-term MPL_S and the long-term MPL_L.

The short-term MPL_S is calculated as the mean value of all losses that occurred during interval I since the last quality control.

The long-term MPL_L is produced from previous packet loss metrics that are weighted according to the time that has occurred since their calculation. For calculating the MPL_L, older metrics weigh less than recent ones. The mechanism that defines the weight a metric should have uses variables n and s, where n is the volume of metrics available and s is a 'slope'. The slope variable can take values between 0 and $2/\{n(n-1)\}$ and ultimately defines how important older metrics are in relation to recent ones. A larger value for s means

that newer values are more important. The weight is calculated as follows:

$$w_i = s \cdot i + \frac{1}{n} - \frac{s(n+1)}{2},\ i = 1, \cdots, n. \quad (1)$$

From (1), for every positive integer n, we can verify that

$$\sum_{i=1}^{n} w_i = 1. \quad (2)$$

If we have n_0 available packet loss metrics PL_i, where i=1,⋯, n_0, the mean long-term packet loss MPL_L is

$$MPL_L = \sum_{i=1}^{n_0} w_i \cdot PL_i. \quad (3)$$

If the slope s is 0 (zero), meaning all metrics are equally important, then

$$s = 0 \Rightarrow w_i = \frac{1}{n},\ i = 1, \cdots, n_0. \quad (4)$$

And

$$MPL_L = \sum_{i=1}^{n_0} \frac{1}{n_0} \cdot PL_i = \frac{PL_1 + \cdots + PL_{n_0}}{n_0}. \quad (5)$$

Equation (5) is the mean packet loss over all available metrics.

The short-term packet loss MPL_S is compared to limit L and defines which clients will be switched-down. Long-term packet loss MPL_L is compared to limit U and defines which clients will be switched-up.

In practice, clients will be switched-down when they suffer packet losses, even short-term. On the other hand, for switching-up, the algorithm takes into account the long-term quality behavior of the client. This particular model is similar to the model of additive-increase/multiplicative-decrease that is used by the TCP protocol; a rate decrement is immediate whereas a rate increase takes time.

2. Simulation

For verifying the above algorithm, we simulated it using the NS-2 platform [14]. We used a relay node with three streams at 256, 128, and 64 kbps. The streams had a constant bitrate throughout the simulation (CBR flows). The mean time between client arrivals was 0.5 seconds, exponentially distributed (Poisson arrivals).

For the simulation, we used the topology in Fig. 7 with the parameters depicted in Table 1. With the aforementioned configuration, we programmed 200 clients to request a

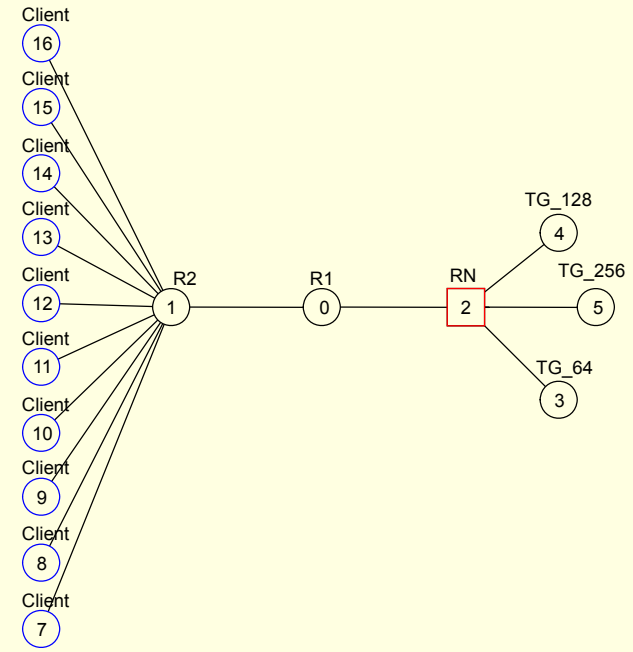


Fig. 7. Simulation topology.

Table 1. Simulation parameters.

Link	Bandwidth (Mbps)	Queue size (packets)	Queue type
RN → R1	40	300	Drop tail
R1 → R2	40	300	Drop tail
Clients → R2	384	NS default	NS default

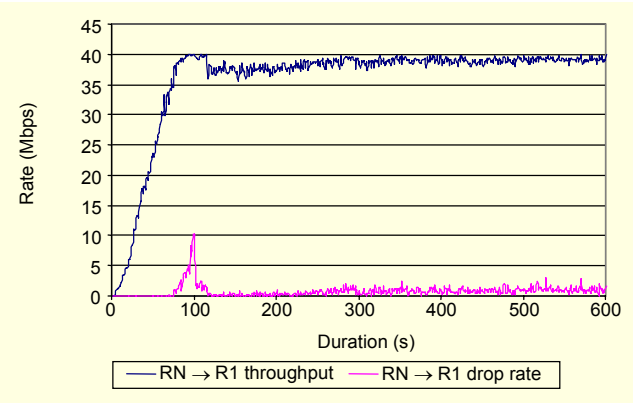


Fig. 8. Throughput and drop rate for RN→R1 link.

256 kbps stream over a 40 Mbps connection for a simulation time of 600 seconds. Limits L and U had 0.08 and 0.01 percent packet losses, respectively.

Figure 8 displays the throughput of the RN→R1 link and the respective drop rate over the course of the simulation.

As we can observe, during the early transitional phase where

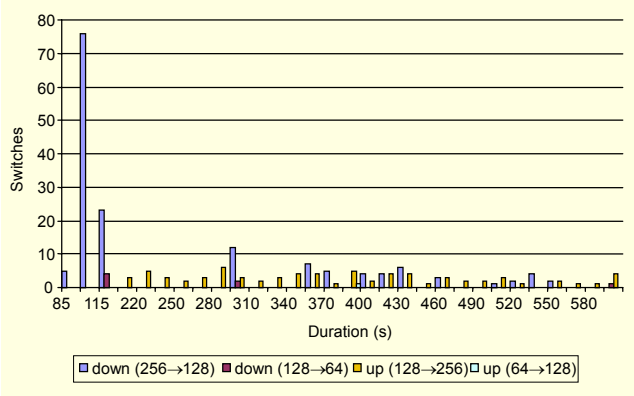


Fig. 9. Number of switches.

the relay node analyzes the RTCP reports and before the first quality control, packet loss is significant. From that point on, the relay node makes the necessary switches and the system comes to a balance, maximizing bandwidth usage on the link and minimizing packet loss.

Figure 9 displays the number of switches that take place in the relay node during the simulation. We observe that, in the beginning, almost half of the clients are switched down since the total bandwidth required surpasses the link capacity. After the few first quality controls, the system comes to a balance, mainly switching clients up so as to maximize bandwidth usage.

Summarizing, we conclude the following:

- The utilization of the network link is very close to the link's capacity, accomplishing almost the maximum feasible throughput.
- In parallel, we observe that the packet drop rate – though high as expected in the beginning – drops significantly in a matter of a few transitions, achieving an acceptable packet loss.
- In addition, as soon as the transitional stage is over, the number of switches in relation to the number of clients is minimized and the system stabilizes.

3. Applying the Rate-Adaptation Algorithm

Other multimedia platforms such as Apple's QuickTime and RealNetworks' Helix also incorporate rate-adaptation mechanisms. These mechanisms work in a similar way to the algorithm described in the previous paragraphs, that is, whenever packet loss is experienced, the client is switched to a less bandwidth-demanding version of the stream. All variants are pre-encoded and available together with the original stream in a single 'fat' stream. Although this method is obviously feasible and relatively easy to implement, it has a major drawback when coupled with the overlay network.

As depicted in Fig. 10, the aforementioned method consumes network resources in the core distribution tree as we need to stream, in addition to the original flow, all the available variants as well, thus requiring more bandwidth from our delivery network. In addition, there is a strong possibility that only a subset of the variants is actually used at the edge nodes, and the extra network resources consumed in the core are wasted.

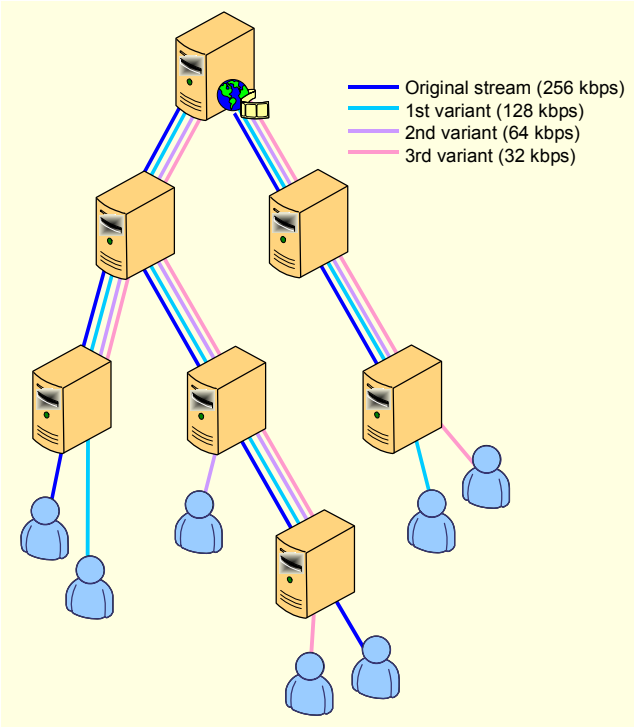


Fig. 10. Rate adaptation using pre-encoded variants.

An MPEG-4 Specific Implementation

The MPEG-4 standard incorporates the concept of object-based representation of media scenes as described previously. We utilize this concept for applying the rate-adaptation mechanism described before.

In MPEG-4, each scene can be described as the combination of individual objects. The amount of data required for representing each of these objects contributes to the total bandwidth required by the flow. The relay node can derive variants of the original flow, thus controlling the required bandwidth, by selectively transmitting a subset of the objects that compose a scene.

As portrayed in Fig. 11, an incoming MPEG-4 flow consists of four different objects with specific bandwidth needs. The relay node selectively discards some of these objects and creates the variants needed by the rate-adaptation process.

To put the aforementioned technique in effect, we require a

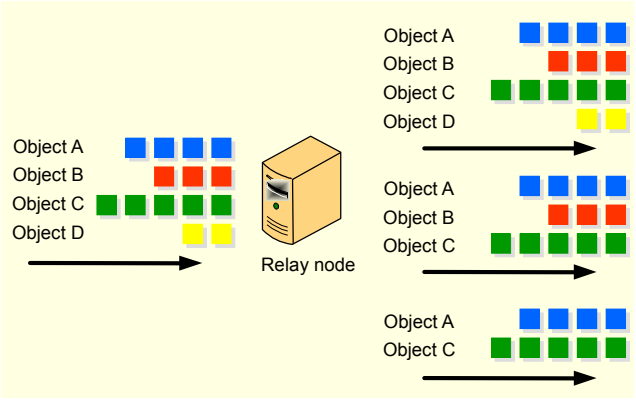


Fig. 11. Object-based rate adaptation.

special processing to be applied during the encoding phase, where objects are defined. Specifically, we record the bandwidth required by each object, and then we classify the objects according to their significance in the representation of the scene. Classifying the objects effectively defines the variants. In Fig. 11, we can see that Object A and Object C are deemed as very important and are present in every variant. Following in significance is Object B, which is present in two out of the three possible cases, and finally, Object D is the least important and is present only in the original stream.

The NDM carries this information and conveys it to the relay nodes before the rate-adaptation algorithm is put into effect. More technically speaking, the relay node parses the BIFS and OD descriptors from the incoming stream and breaks down the scene to its object-defined representation. Whenever a switch-up or a switch-down is hinted by the rate-adaptation algorithm, the relay node strips or sets up the appropriate DMIF channels that carry the elementary streams describing objects, according to the NDM-provided bandwidth classes, that were pre-defined during the encoding process.

V. Implementation and Use

The client software is based on Osmo4 [16], which in turn is based on the MPEG-4 Systems reference software. The server software is based on the ApadanaServer [17]. Both the server and client run on a Microsoft Windows operating system. The overlay modules (the relay node and the NDM) are implemented from scratch over Microsoft's .NET platform [18], which is based on the notion of the common language runtime (similar to a virtual machine in Java terms) against which applications are compiled and executed. Therefore, a .NET application can run on any machine that supports the runtime. Although the primary operating system for our platform is Windows (2000/XP/2003), where the official .NET implementation runs, we have also successfully deployed the overlay network over Linux though Ximian's Mono Initiative [19], which provides a .NET implementation for Unix-like operating systems such as Linux, FreeBSD, and Mac OS-X.

In the following paragraphs, some real-use scenarios for the platform are presented and a performance evaluation is given.

1. Example Scenarios

Suppose we have to cover a major sports event like the Olympic Games in Greece. The platform users have the ability to tune either to the main program that consists of live streaming (broadcast service) or select a certain pre-recorded content that is made available in a VoD fashion (on-demand service).

A. Broadcast-like Service

The streaming server is situated in Athens where the action takes place. We know that we expect many viewers from Athens and Thessaloniki (the largest cities in Greece) so we have two relay nodes in these cities to balance the load. We also expect viewers in Corfu and Crete, where there are many tourists from abroad who would probably like to watch the Olympics, too.

We manually deploy the stream to a relay node in Athens and also in a relay node in Thessaloniki since we definitely know that there will be demand for content. The NDM default overlay profile has *Locality* and *Resource Availability* as parameters. The relay node overlay profile consists of *Stream Availability*, *Network Proximity*, and *Stream Quality*.

Users coming from Athens will be redirected to one of the relay nodes in Athens (since *Locality* is the first criterion) and the choice would perform load-balancing since the currently less-loaded relay node will be selected (*Resource Availability* is the second criterion). The same holds true for users from Thessaloniki.

If a user from Corfu wishes to participate, he will be redirected to the only relay node in Corfu. That particular relay node does not currently have the stream, so a second round of overlay evaluation will take place. The profile for the relay node dictates *Stream Availability* as the first criterion, so every relay node in Athens and Thessaloniki that has the stream is a candidate. The second criterion is *Network Proximity*, so a ping-like experiment is performed and the choice is further limited to relay nodes in Thessaloniki[1]. The final choice would be the relay node that offers the best stream quality (this is the third criterion). The relay node fetches the stream from Thessaloniki and delivers it to the local client. Any users following from Corfu will directly get the stream from the local relay node. In the same fashion, users from

1) We suppose that, network-wise, Thessaloniki is closer to Corfu than Athens.

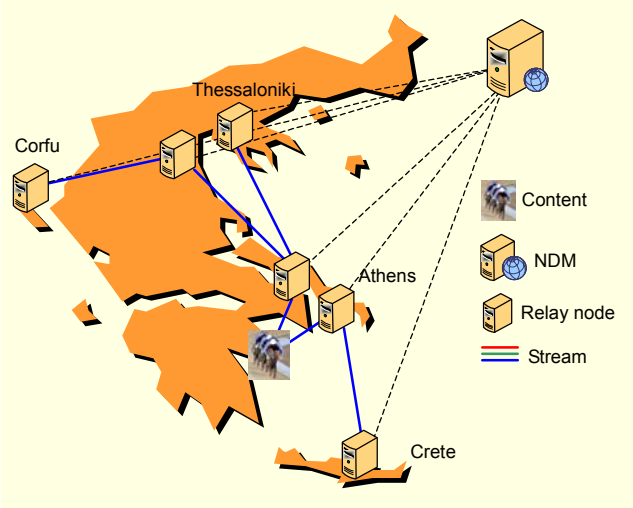


Fig. 12. Example of the broadcast service.

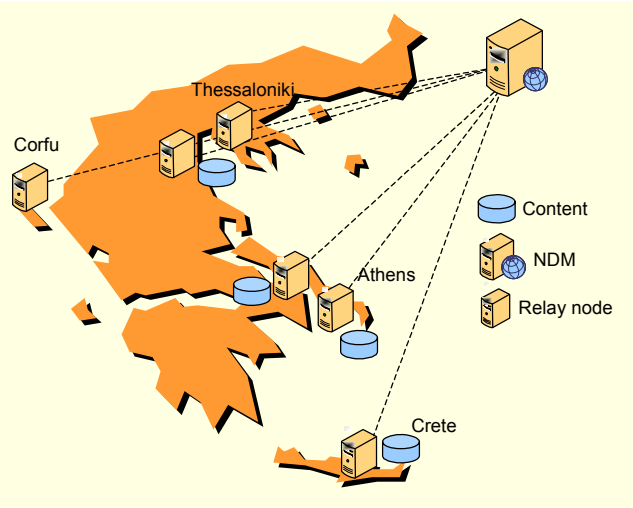


Fig. 13. Example of the on-demand service

Crete would select the local relay node that will, in turn, get the stream from Athens.

If, at some time, there are no more users from Corfu or Crete, the respective relay node will prune the connection to Thessaloniki/Athens so as not to waste network resources.

Since this is a broadcast scenario, the relay nodes can additionally enforce the rate-adaptation techniques described in the previous chapter.

B. On-Demand Service

For the on-demand service, we assume the deployment of a popular file (for example, the opening ceremony) at two streaming servers in Athens and one in Thessaloniki. The default overlay profile for the NDM is *Locality* and *Resource Availability*.

Users from Athens will get connected to the less-loaded streaming server in Athens. Users from Thessaloniki will connect to the only streaming server in Thessaloniki. Users from Corfu or Crete will get connected to the less-loaded streaming server, either in Athens or Thessaloniki.

If we observe, through the administrative interface, that the particular content is very popular in Crete for example, we could dictate the local relay node to fetch the content and make it available from its adjacent streaming server. From this point forward, any new users from Crete will be able to get the video directly from the local server without burdening the remote hosts.

2. Performance Metrics

In order to evaluate the efficiency of our implementation, we tested the capability of the relay node in relaying media streams to a large number of clients. We were particularly interested in

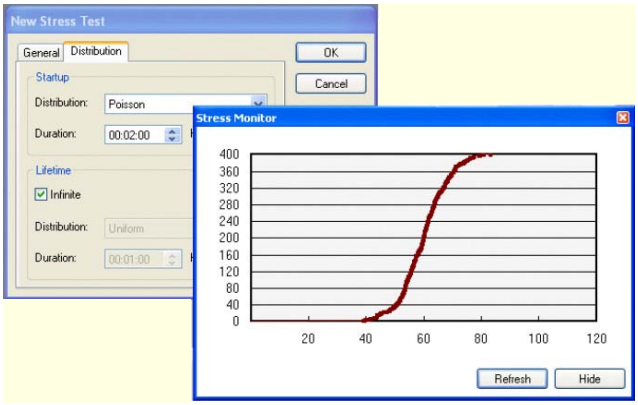


Fig. 14. Stress simulation software.

evaluating processor occupancy and memory usage as the number of clients increases. This kind of information is essential for planning the distribution architecture and the placement of relay nodes in the network. Obviously, measuring streaming performance refers only to the broadcasting scenario as the relay nodes are not used for streaming in the VoD scenario, as explained before.

For the stress tests, we used a PC with relatively low memory and processing power capabilities (Intel Pentium 4 at 1.9 MHz with 256 MB RAM equipped with Windows 2003) acting as a relay node. This allowed us to reach saturation points in processor occupancy and memory usage much faster, without ever reaching the maximum bandwidth that is supported by the network infrastructure (in our case a 100 Mbps switch). This way, we can avoid any side effects in our measurements such as extreme packet loss due to inadequate bandwidth.

For simulating client requests, a special simulation application, shown in Fig. 14, was developed that was capable of creating RTP/RTSP sessions to a server. The application

provides a series of models (uniform, Poisson, and exponential) for distributing client sessions over time.

We conducted stress tests using three different streams with average bitrates of 64, 128 and 256 kbps. The simulation software created 400 RTP/RTSP sessions for each bitrate using the Poisson distribution over a ten minute time frame.

Figure 15 displays the CPU occupancy of the relay node over the number of clients connected for the three streams. The processor occupancy starts from an offset of about 20%, which is dedicated to the supporting of the relay node process and other processes (not related to streaming) running on the relay node host. Following this, we have a linear increase of the processor occupancy in relation to the number of clients, reaching up to the point of saturation (naturally, around 100%).

The linear increase of the processor occupancy is due to the careful threading design of the relay node. It is a well-known design approach to avoid spawning numerous processing threads as it may quickly drench the operating system resources by the constant switching between them (also known as thread thrashing). The relay node keeps a constant number of processing threads (around 20) by using thread pooling and taking advantage of the asynchronous features of the .NET platform.

To verify that the relay node behaves efficiently even with a limited number of threads, we prepared the chart in Fig. 16. The figure displays the outgoing rate (in MB/s) of the relay node in conjunction with processor usage and the number of client sessions for a 256 kbps stream. We can easily observe that the rate is linear (equaling 256 kbit multiplied by the number of clients) and steadily increasing with the number of sessions until the processor usage reaches the saturation point. From then on, the relay node does not perform adequately as the processor is unable to sustain the number of clients.

Figure 17 displays the memory usage of the relay node for the 256 kbps stream in relation to the number of clients. We observe a small and steady rise that continues well after the processor saturation point (which is around 150 clients for the 256 kbps stream). When memory usage reaches the hardware limit of 256 MB of the test PC it falls back to the starting offset and starts rising again.

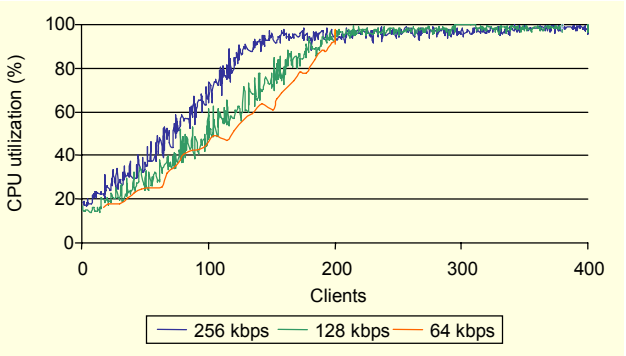


Fig. 15. Percentage of processor occupancy in relation to number of clients.

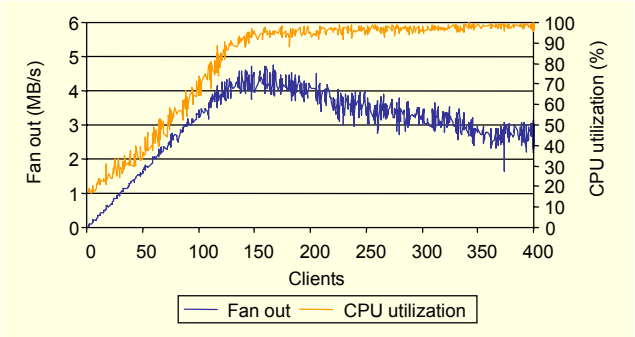


Fig. 16. Fan out and processor occupancy in relation to number of clients.

Fig. 17. Memory consumption for a 256 kbps stream.

The sharp and repeated falls in memory consumption are explained by the memory model of the .NET implementation of the relay node. .NET manages memory with a garbage collector mechanism. The garbage collector reclaims memory periodically or when absolutely needed. The falls in the graph indicate garbage collection invocations. Since available memory reaches the starting offset after each invocation, it is safe to assume that the number of clients have no or very limited effect on the memory usage of the relay node.

The above results indicate that the relay node can efficiently be used in real-world scenarios. With the limited processing power of the test machine, we could serve up to 150 clients in 256 kbps and over 200 for 64 kbps. What is more important is the predictable nature of the relay node behavior in relation to the number of client sessions. For real-world deployment, we could rely on stress tests with a small number of clients that would allow us to accurately predict saturation points and appropriately load-balance our network.

VI. Conclusions and Future Work

In this paper we presented a new interactive multimedia streaming platform utilizing overlay networks for the

distribution of media streams. The driving force behind our design has been MPEG-4 and its advanced offerings. The platform is scalable and incorporates QoS functionality. Furthermore, the overall design is standards-based, and parts of the platform can be re-used with alternative/commercial solutions for media delivery.

Regarding future work, we mainly concentrate our efforts on two areas: scalable coding and DCCP. Both technologies are expected to greatly enhance the platform's QoS offerings.

We are particularly interested in scalable encoding as supported by the MPEG-4 standard in the form of fine granularity scalability [20], [21], and on more recent advances made in the relevant area such as wavelet-based video encoders. Special focus will be made on the adoption of multiple description coding [22], which allows content encoding to a set of hierarchically equal bit-streams. The receiver can then reconstruct the original signal in accordance with the amount of the total available descriptions it has received. Scalable coding is a natural candidate for the rate-adaptation algorithm, and its integration with the current method of object-based adaptation is an intriguing subject.

Furthermore, we intend to adopt the DCCP [23], which is a new transport protocol specifically suited to cover the needs of applications making use of real-time media. DCCP is intended to be used as a replacement for the traditional UDP and TCP protocols for the purpose of carrying media streams, and it incorporates an advanced feedback mechanism that is expected to render feedback-related protocols (such as RTCP) redundant.

References

[1] Jong Myeon Jeong et al., "Development of Interactive Data Broadcasting System Compliant with ATSC Standards," *ETRI Journal*, vol.26, no.2, 2004, pp 149-160.

[2] ISO/IEC 14496-1, *Information Technology - Coding of Audio-Visual Objects - Part 1: Systems,* International Standard, 2001.

[3] Kyuheon Kim et al., "Interactive Broadcasting Contents Authoring System," *Telecommunications Review,* SK Telecom, vol. 11, no.5, 2001, pp 688-698.

[4] Bong-Ho Lee et al., "A Framework for MPEG-4 Contents Delivery over DMB," *ETRI Journal*, vol. 26, no.2, 2004, pp 112-121.

[5] ISO/IEC 14496-6, *Information Technology - Coding of Audio-Visual Objects - Part 6: Delivery Multimedia Integration Framework(DMIF)*, ISO/IEC, Mar. 2000.

[6] Apple Computer, Inc., QuickTime Streaming Server, http://www.apple.com/quicktime/streamingserver/

[7] RealNetworks, Inc., Helix, http://helixcommunity.org/

[8] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications," *RFC3550*, July 2003.

[9] H. Schulzrinne, A. Rao, and R. Lanphier, "Real Time Streaming Protocol (RTSP)," *RFC2326*, Apr. 1998.

[10] J. van der Meer, D. Mackie, V. Swaminathan, D. Singer, and P. Gentric, "RTP Payload Format for Transport of MPEG-4 Elementary Streams," *RFC3640*, Nov. 2003.

[11] ISO/IEC 14496-10, N5555, *Information Technology - Coding of Audio-Visual Objects - Part 10: Advanced Video Coding*, FDIS, 2003.

[12] ISO/IEC 14496-3, *Information Technology - Coding of Audio-Visual Objects – Part 3: Audio*, 1999.

[13] Akamai Technologies, Inc., http://www.akamai.com/

[14] The Network Simulator - ns-2, http://www.isi.edu/nsnam/ns/.

[15] Ch. Z. Patrikakis, Y. Despotopoulos, P. Fafali, Jihun Cha, and Kyuheon Kim, *Proc. of SPIE*, vol. 5558, Nov 2004, pp 371-381.

[16] ENST, Osmo4, http://www.comelec.enst.fr/osmo4/

[17] University of British Columbia, ApadanaServer, http://lan.ece.ubc.ca/apadana.html

[18] Microsoft Corp., .NET framework, http://www.microsoft.com/net/

[19] Ximian Inc., Mono Project, http://www.mono-project.com/.

[20] ISO/IEC 14 496-2/FPDAM4, *Coding of Audio-Visual Objects, Part-2 Visual, Amendment 4: Streaming Video Profile,* July 2000.

[21] Weiping Li, "Overview of Fine Granularity Scalability in MPEG-4 Video Standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 11, pp. 301-316, Mar. 2001.

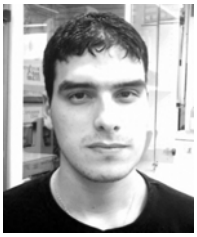
[22] V. K. Goyal, "Multiple Description Coding: Compression Meets the Network," *IEEE Signal Processing Magazine*, vol. 18, no. 5, Sep. 2001, pp. 74-94.

[23] Eddie Kohler, Mark Handley, and Sally Floyd, *Datagram Congestion Control Protocol*, Internet Draft, Mar. 2005.

Hyun-Cheol Kim received the BS and MS degrees in electronics engineering from Kyunghee University, Suwon, S.Korea in 1998 and 2000. Since 2000, he has been a member of the Engineering Staff in Electronics and Telecommunications Research Institute (ETRI), S.Korea. He has been engaged in the development of an MPEG-4 authoring system and DMB terminal. His research interests include video processing, multimedia communications, and interactive broadcast systems.

Charalampos Z. Patrikakis was born in Athens, Greece, in 1970. He received the Dipl.-Ing. degree and the PhD from the Electrical Engineering and Computer Science Department of the National Technical University of Athens (NTUA), Greece. He is working in the field of computer networks research in national, European and international projects. He has participated in several European Union projects (ESPRIT, RACE, ACTS, IST, eContent). He has acted as a work-package leader and as technical coordinator to several European projects. His main research interests are in the areas of IP service design and implementation, multicasting in IP networks, IP transport protocols and Peer to Peer networking, and streaming media distribution algorithms and protocols. He is a member of IEEE, a member of the Greek Computer Society, a certified trainer by the National Accreditation Centre of Vocational Training Structures and Accompanying Support Services, and a member of the Technical Chamber of Greece.

Nikos Minogiannis was born in Athens, Greece, in 1977. He received the Dipl.-Ing. degree from the Computer Engineering and Informatics Department from the University of Patras in July, 2000. He is currently a PhD candidate in the Electrical and Computer Engineering Department of NTUA. During this time, he has been a Research Associate in the Telecommunications Laboratory of the department. His research interests include peer-to-peer networking, streaming media distribution protocols, and multimedia for portable devices. He has participated in several European and international projects on the aforementioned subjects. He is a member of the Technical Chamber of Greece.



Pantelis N. Karamolegkos was born in Athens, Greece, in 1978. He received his Dipl.-Ing. degree from the Electrical Engineering and Computer Technology Department of the University of Patras. He is currently a PhD candidate and research associate of the Telecommunications Laboratory of NTUA, working in national and European research projects. His main research interests are in the areas of peer-to-peer networking, autonomic networking, and streaming media distribution algorithms and protocols. He is a member of the Technical Chamber of Greece.

Alex Lambiris was born in Athens, Greece, in 1980. He is studying mathematics at the University of Athens (UOA), Greece. He has worked in the field of computer networks research and has participated in national and European research projects. His main research interests are in the area of peer-to-peer networking.

Kyuheon Kim received the B.Sc in electronic engineering in 1989 from Hanyang University, Seoul, S.Korea, and the PhD in electrical and electronic engineering from University of Newcastle upon Tyne, UK, in 1996. From 1996 to 1997, he worked as a research fellow for University of Sheffield, UK, and from 1997 to 2006 as a head of interactive media research team in Broadcasting Media Technology Department, ETRI, S. Korea. Also, he was a head of the Korean delegates to the MPEG International Standard Organization, and a task group chairman of the convergence between broadcasting and telecommunications in AWF (Asian-Pacific Telecommunity Wireless Forum). Currently, he is an Associate Professor in Kyunghee University, S. Korea. His interests are in digital signal and video processing, and interactive multimedia broadcasting systems.