Client-controlled QoS management in networked virtual environments

Non Peer-reviewed author version

MONSIEURS, Patrick; WIJNANTS, Maarten & LAMOTTE, Wim (2005)
Client-controlled QoS management in networked virtual environments. In: NETWORKING - ICN 2005. p. 268-276.

DOI: 10.1007/b107117
Handle: http://hdl.handle.net/1942/2815

Client-controlled QoS Management in Networked Virtual Environments

Patrick Monsieurs, Maarten Wijnants, Wim Lamotte

Expertise Centre for Digital Media
Limburgs Universitair Centrum
Universitaire Campus, B-3590 Diepenbeek, Belgium
{patrick.monsieurs, maarten.wijnants, wim.lamotte}@luc.ac.be

Abstract. In this paper, we propose an architecture to regulate the bandwidth usage of multimedia streams in networked virtual environments. In this architecture, intelligent proxies are placed in the network. These proxies can transcode incoming streams to lower quality versions of those streams, thereby decreasing network traffic. A network intelligence layer at the receiver controls these transcoders based on the bandwidth the streams consume, and the importance that the receiving application assigns to each stream. To access this latter information, the network intelligence layer provides an interface between the QoS management of the network and the application's interest manager. This interest manager assigns a relative importance to each individual network stream. As a result, the network intelligence layer separates application logic from network QoS management, thereby maximizing its reusability.
These concepts were implemented in an existing networked virtual environment framework, and experiments were performed to validate the ideas. The experiments demonstrate that bandwidth allocation can be changed dynamically, based on user interest, thereby maximizing network throughput and quality of experience.
Keywords: QoS, Network intelligence, Multimedia.

1 Introduction and Related Work

Transmission of multimedia content over the Internet tends to consume large amounts of bandwidth. Therefore, applications where a large number of users simultaneously send video and audio need some way to reduce the amount of transmitted data, while still maintaining an acceptable quality. This paper focuses on techniques to reduce the downstream bandwidth usage in the final segments of the network connection. By incorporating application knowledge about the importance of individual streams, we attempt to maximize the user's quality of experience.

Several techniques exist to reduce the downstream bandwidth requirements of networked multimedia applications. A first technique is to subdivide the environment in several regions, associated with multicast groups. Receivers join only the multicast groups of those regions they are interested in, and as a result will only receive the streams sent by users located in these regions [1]. Receivers with less available

bandwidth can then subscribe to fewer regions to limit the amount of data they need to receive. This allows receivers to regulate the total amount of received data. However, when using this technique, the bandwidth can only be adjusted by joining or leaving entire regions. Sometimes it is desirable to have more fine-grained control over the amount of bandwidth consumed.

Another technique consists of reducing the quality of a network stream by its sender, based on the receiver's interest in that stream [3]. While this reduces the network load of congested links, receivers with uncongested links needlessly receive lower quality streams. Also, this approach does not tend to scale well with large numbers of receivers.

The highest level of control over the bandwidth consumption is achieved when every individual stream can be either accepted, rejected or transcoded inside the network before it reaches the client. We propose an architecture where intelligent proxies are placed in the network nearby the clients. Clients notify the proxies of the relative importance of each network stream, from the client's point of view. The proxies use knowledge of the maximum available bandwidth, the bandwidth of each individual stream, the relative importance values of each stream, and the capabilities of its transcoders to decide to accept, reject or transcode those stream.

The importance of each individual stream is determined by the application's interest manager. To maintain application-independence of the network intelligence layer, a simple programming interface is provided that is linked to the interest manager of the application. Some examples of user interest detection already exist in the literature. In [3], more importance is assigned to video conferencing participants whose window is currently selected. In [4], video playback and decoding is suspended in obscured windows to save CPU resources.

Content-based transcoding of data streams by proxies in the network is not a new idea. In [5] and [6], images of HTTP streams are transcoded to different resolutions based on the capabilities of the receivers. For example, images can be rescaled or cropped when sent to devices with small displays. In [7], transcoding-enabled caching proxies are placed in the network. These proxies transcode video streams, based on the capabilities of both the client and the network. The highest quality video stream is cached at the proxy, and is transcoded for clients that require a lower quality version.

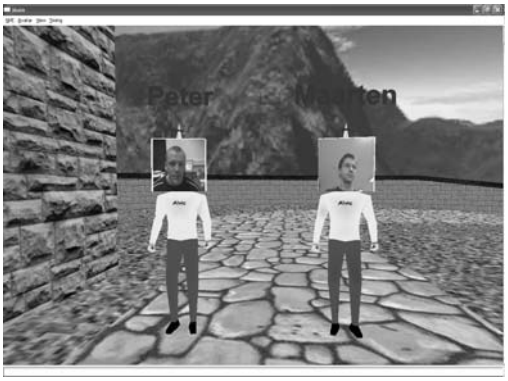

Fig. 1. Screenshot of video avatars in our networked virtual environment

Distribution of a network's link available bandwidth is discussed in [8]. All network traffic arriving at a router is divided into a number of classes, which are placed in a hierarchy. Non-leaf nodes in this hierarchy specify a distribution of minimum available bandwidth among its child nodes. Each class in the hierarchy is allocated the specified minimum bandwidth. When classes do not consume all available bandwidth, the remaining bandwidth is distributed among the other classes in the hierarchy. The difference between this approach and our technique is that this approach manages bandwidth by assigning individual network packets to appropriate queues. Our approach does not consider individual network packets, but deals with streams as a whole using the average bandwidth consumed by those streams.

2 Example NVE Application

To illustrate and test the concepts of this paper, we used our in-house developed networked virtual environment where users are represented by video avatars. A detailed description of this environment is given in [2][9][10], and a screenshot is shown in Fig. 1. In this system, web cams capture the faces of the users. These video streams are subsequently encoded in three different frame rates and/or bit rates.

The virtual world is subdivided in a number of regions. Each region has a multicast address associated with it for each video quality. Each client is interested in receiving video and positional data only from a number of regions in its vicinity. Based on the available bandwidth and the distance to those regions, an appropriate video stream quality for that region is selected, and the associated multicast address is joined. While receiving clients move around in the environment, multicast groups are dynamically joined and left.

3 Network Intelligence Architecture

When a network link becomes saturated, throughput decreases and delays become higher. As a result, in order to maintain the quality of experience in networked applications it is necessary to avoid requesting more bandwidth than is available. This can be achieved by reducing the quality and bandwidth usage of certain streams, or by completely blocking them, before they reach the client. To maintain a high quality of experience, it is essential that streams that are important to the user be allocated more resources than less important streams.

Determining the importance of individual streams is an application-specific issue. On the other hand, calculating and distributing available bandwidth is an application-independent networking task. To keep a strict separation between application logic and network management operations, a network intelligence layer is introduced between the transport and application layers, as shown in Fig. 2. This layer communicates with an intelligent proxy located inside the network. The proxy is able to transcode or block incoming network streams. The network intelligence layer queries the interest manager of the application to determine the relative importance of each stream. The importance of each stream is then communicated to the proxy. If a

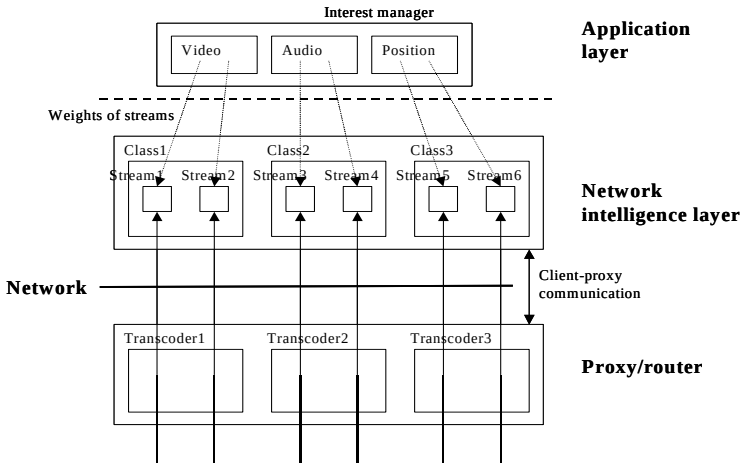

Fig. 2. Overview of the proposed network intelligence architecture. Incoming network traffic can be limited by an intelligent proxy located inside the network, by blocking or transcoding network streams

threshold of the available network capacity at the proxy is exceeded, the proxy transcodes or blocks the least important streams until the desired bandwidth usage is reached. The individual components of the architecture are discussed in the following sections.

3.1 Intelligent Proxy

In this section, we will present a short description of the intelligent proxy. A more detailed description of the proxy is given in a different paper, which is submitted to another conference [11].

The intelligent proxy, located in the network near the client, contains a number of content-based transcoders. These are able to transcode incoming network streams into less detailed streams that consume less bandwidth. The concept of a transcoder in this context is very general. It can range from a simple filter that blocks specific streams, to a process that decodes, re-encodes and retransmits an entire video stream. Because the transcoding of video streams is very processor intensive, it does not tend to scale well with large numbers of streams. It is therefore better for clients to transmit a layered version of the stream using a codec such as MPEG-4 FGS [12]. This way, transcoding a stream can be limited to filtering those packets that contain the desired quality and discarding the other packets.

The proxy measures the consumed bandwidth of the network streams used by all connected clients. When the downstream capacity of the client application is exceeded, the proxy transcodes specific streams to a lower quality until the available bandwidth is no longer exceeded. As a result, the available downstream capacity can be fully utilized. The algorithm used to distribute the available bandwidth over the different streams is described in section 3.3. The communication between client and proxy is described in section 3.4.

By placing the intelligent proxy near the receivers, the complexity of managing network traffic is located at the edge of the network. Consequently, the proxy will have to handle only a limited number of streams. We propose that these proxies are placed at DSLAM-level of xDSL networks. This approach manages the bandwidth of

the final links between senders and receivers, which is often the most problematic part of the connection. Furthermore, the intelligent proxy does not have to be located at the nearest router to the receiver, which has the advantage that not every router in the network must support the architecture.

3.2 Network Intelligence Layer

The network intelligence layer provides the application with a set of networking operations that replace the standard Winsock or Unix networking operations. Upon reception of a network stream, the application provides information about the content of the network stream to the network intelligence layer. This allows the network intelligence layer to associate the appropriate transcoder with this network stream at the proxy.

This layer also provides a generic interface to the interest manager of the client application. Using this interface, the network intelligence layer can determine the relative importance of every network stream used by the application. The importance of each stream is sent to the intelligent proxy, which will use this information to reduce or block less important streams whenever the available bandwidth is exceeded.

3.3 Distribution of Available Bandwidth

Similar to the hierarchies used by hierarchical link sharing [8], we build a hierarchy of streams and components to manage the available bandwidth by the proxy. Individual streams are represented as leaves in this hierarchy, while other nodes implement a bandwidth distribution technique. We use the following distribution techniques:

- *Priorities*: The child node with the highest priority is assigned all requested bandwidth. Any remaining bandwidth is distributed to the second-highest child, etcetera.
- *Mutexes*: All bandwidth is allocated to a single child of the collection.
- *Weighted collection*: Each child of such a node is assigned a relative weight, and bandwidth is distributed among children based on their weight and their bandwidth consumption. The algorithm used is described below.

In a weighted collection node, every stream s_i is assigned a weight w_i between 0 and 1. Every unregulated stream s_i consumes m_i bandwidth. The maximum desired bandwidth M of all streams is $\Sigma_i m_i$. If M exceeds the maximum available bandwidth B, the bandwidth of each stream must be modified using the following algorithm:

Assume that every stream s_i can be transcoded to a set of bandwidths S_i. This set can contain a number of discrete values, or the continuous range between 0 and m_i. $h(i, b)$ is defined as the highest bandwidth of a stream s_i that is less than bandwidth b, and is the highest value of S_i that is less than or equal to b. The weighted sum W of all streams is $\Sigma_i w_i m_i$.

An initial estimate of the target bandwidth for each stream is $t(i) = h(i, w_i m_i B/W)$. In an ideal case where $\forall i$: $h(i, b) = b$, $\Sigma_i t(i) = B$ and the available bandwidth would be

used optimally. It is more likely, however, that every stream consumes less bandwidth than they are allocated, resulting in a remaining bandwidth $R_0 = B - \Sigma_i\, t(i)$. The remaining bandwidth R is then allocated to all streams, starting with the stream with the highest weight. Assuming the streams are sorted by their weight, where w_0 is the highest weight, the final target bandwidth for each stream equals $T(i) = h(i,\, t(i)+R_i)$, and $R_i = R_{i-1} - (T(i - 1) - t(i - 1))$ for $i > 0$.

3.4 Communication Between Proxy and Client

The following tasks are handled by the communication between proxy and client:

- *Admission control*: The client's network intelligence layer can notify the proxy that streams destined for a specific port must be blocked by default. When a new incoming stream is received at this port by the proxy, the client is notified of this new stream.
- *Creation of a stream hierarchy*: The application uses the network intelligence layer to build the hierarchy of streams used to distribute bandwidth. When the client receives a new stream, a corresponding node is created on the proxy. The client also specifies the content type of this stream so the proxy can use the correct transcoder to modify that stream.
- *Communicating importance of streams*: Clients transmit weight values ranging from 0 to 1 that indicate the importance of individual streams to the proxy. These values are transmitted at regular time intervals. In our application, this weight value is determined by the distance to the sender's avatar.
- *Multicast tunneling*: The client's network intelligence layer intercepts the client's requests to join or leave multicast groups and transmits these to the proxy. The proxy subscribes to or unsubscribes from these groups, and unicasts the multicast traffic to and from the client.

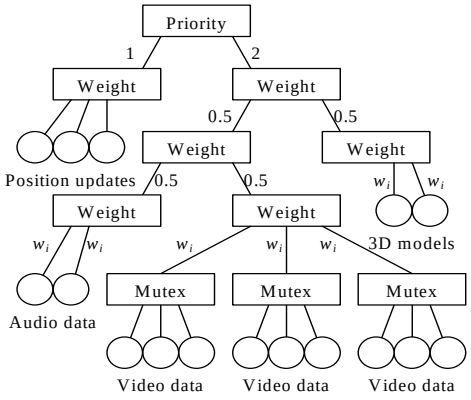

Fig. 3. Hierarchy of network streams of the client application

4 Experimental Results

In our experiments, four clients running the unmodified version of the NVE and an intelligent proxy are connected through a local area network. Behind the intelligent proxy is a different network, containing a client running a modified version of the NVE using the network intelligence layer.

The different streams of the client application are organized in the hierarchy shown in Fig. 3. Circles represent network streams, and rectangles represent distribution primitives as discussed in section 3.3. Position update streams are given the highest priority, because this information is needed by the interest manager to assign weights to other streams. The remaining bandwidth is allocated to the distribution of 3D models in the environment, and to audio/video streams. Because each client transmits three versions of the same video stream, these are grouped together using a mutex primitive.

The weight primitives assign weights according to the relative importance of the components. Therefore, even though all combined audio and video streams are both given a weight of 0.5, this does not mean they are each allocated the same amount of bandwidth. The distribution also depends on the amount of consumed bandwidth, and as audio streams typically consume less bandwidth than video streams, the audio streams will consume less bandwidth overall.

In the experiments that we performed, we only consider the video streams of the clients because these comprise the major part of the network traffic. In the traffic measurement graphs, different video streams are represented by different shades of grey. All three possible qualities of video stream are represented by the same shade of grey, but the amount of bandwidth they occupy varies. This can be observed by the height of the received traffic in the graphs.

In a first experiment, shown in Fig. 4, all clients remain stationary, but the maximum available bandwidth was gradually decreased. Initially, sufficient

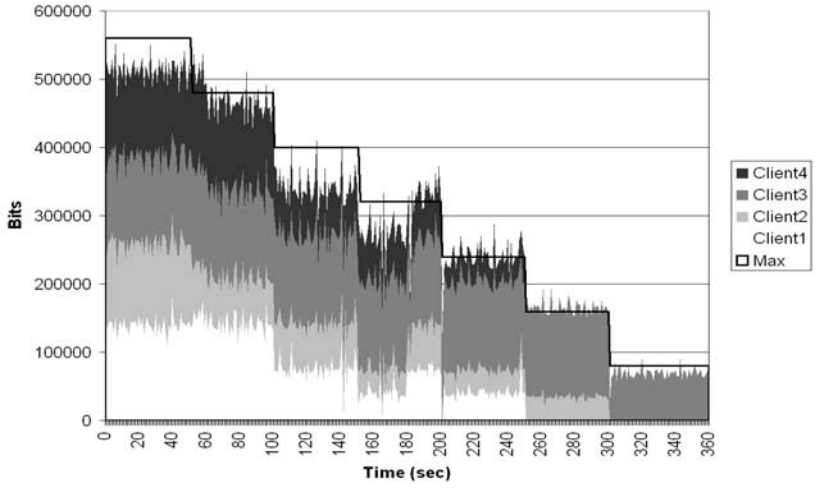

Fig. 4. Experiment 1: gradually decreasing available maximum bandwidth

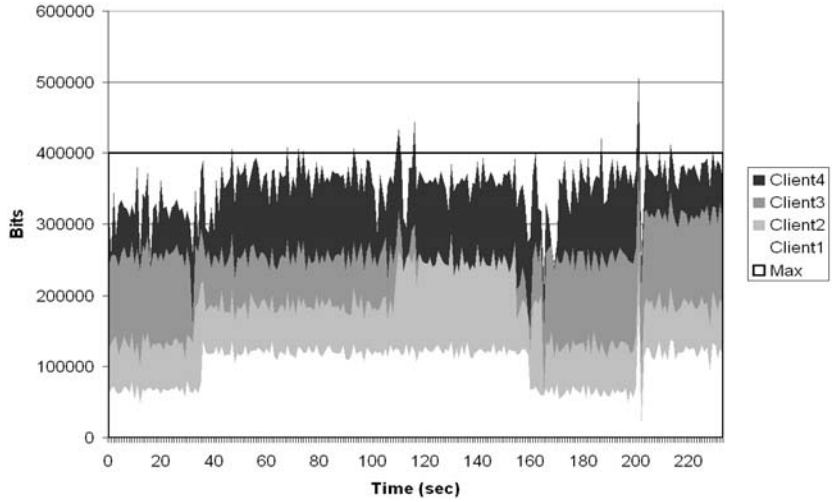

Fig. 5. Experiment 2: changing the relative importance of streams

bandwidth is available to receive all video streams at the highest quality. The avatar of client 3 is nearest to the receiver, and its stream therefore has the highest weight. The avatars of clients 2 and 4 are farthest away, and therefore the quality of their video streams is lowered first by the proxy as available bandwidth is decreased. This is apparent from the lower bandwidth usage of their network stream in the graph.

In the second experiment, shown in Fig. 5, the available bandwidth was kept constant, but not high enough to receive all the streams at maximum quality. After 30 seconds, the avatar of client 3 slowly moves away from the receiver's avatar. As a result, the relative weight of this stream decreases and more bandwidth is allocated to the other streams. After 150 seconds, the avatar of client 3 rapidly moves back towards the receiver's avatar. The dark grey stream is now assigned the highest weight, decreasing the quality of the other video streams. This demonstrates that the bandwidth allocation of the application is dynamically adjusted based on the application's needs.

5 Conclusions

In this paper, we have presented a network intelligence layer that combines application logic and quality of service of networked applications. Downstream bandwidth usage is regulated by intelligent proxies inside the network, based on the relative importance of each network stream. These proxies manage the bandwidth by blocking or transcoding incoming streams. The network intelligence layer communicates with the application's interest manager to indicate the relative importance of each individual network stream. These weights are subsequently used by the proxy to allocate bandwidth to each stream, thereby maximizing the quality of experience for the application user.

The concepts presented here were integrated and tested in an existing NVE framework. The results show that it is possible to manage individual network streams dynamically based on the application's interests. Also, incorporating the network intelligence layer in the application proved to be reasonably straightforward.

6 Acknowledgement

Part of this research was funded by the IWT project number 020659, Alcatel Bell and the Flemish Government. Part of Maarten Wijnants' work was carried out at ANDROME NV, before he moved to the Limburgs Universitair Centrum. We also wish to thank ANDROME for the use of their video codec software.

References

[1] Barrus, J., Waters, R., Anderson, D.: Locales and Beacons: Efficient and Precise Support for Large Multi-User Virtual Environments. In: Proceedings of VRAIS'96, IEEE Computer Society (1996) pp. 204–213

[2] Quax, P., Jehaes, T., Jorissen, P., Lamotte, W.: A Multi-user Framework Supporting Video-based Avatars. In proceedings of the 2nd workshop on Network and system support for games. Redwood City, CA, USA. 2003. ACM Press, pp. 137 - 147

[3] Scholl, J., Elf, S., Parnes, P.: User-interest Driven Video Adaptation for Collaborative Workspace Applications. LNCS 2816, 5th COST 264 International Workshop on Networked Group Communications, NGC, pp. 3-12, 2003

[4] Katchabaw, M. J., Lutfiyya, H. L., Bauer, M. A.: Using User Hints to Guide Resource Management for Quality of Service. The 1999 International Conference on Parallel and Distributed Processing Techniques and Applications, July 1999

[5] Smith, J. R., Mohan, R., Li, C.-S.: Content-based Transcoding of Images in the Internet. In proceedings of IEEE International Conference of Image Processing, Oct '98 (ICIP-98)

[6] Smith, J. R., Mohan, R., Li, C.-S.: Transcoding Internet Content for Heterogeneous Client Devices. In proceedings of IEEE International Conference on Circuits and Systems, May '98 (ISCAS)

[7] Shen, B., Lee, S.-J.: Transcoding-enabled caching proxy for video delivery in heterogeneous network environment. In proceedings of IASTED, Internet and Multimedia Systems and Applications Conference, Aug. 2002

[8] Floyd, S., Jacobson, V.: Link-sharing and Resource Management Models for Packet Networks. In IEEE/ACM Transactions on Networking, Vol. 3 No. 4, August 1995

[9] Quax, P., Monsieurs, P., Lamotte, W.: Performance Evaluation of Client-Based Scalable Video Stream Selection using Autonomous Avatars. In proceedings of the ACM SIGCHI International Conference on Advances in Computer Entertainment Technology (ACE 2004), June 2004

[10] Quax, P., Flerackers, C., Jehaes, T., Lamotte, W.: Scalable Transmission of Avatar Video Streams in Virtual Environments. Proceedings of the 2004 IEEE International Conference on Multimedia and Expo (ICME 2004). Tapei, Taiwan ROC. 2004. IEEE

[11] Wijnants, M., Monsieurs, P., Lamotte, W.: Improving the Client Quality of Service by Incorporating Intelligent Proxies in the Network. Submitted to Networking 2005

[12] Li, W.: Overview of fine granularity scalability in MPEG4 video standard. IEEE Trans. Circuits and Systems for Video Technology, vol. 11, no. 3, pp. 301-317, Mar. 2001