Wireless Network Support for Adaptive Real-Time Applications

Margaritis Margaritidis and George C. Polyzos
Center for Wireless Communications
and
Computer Systems Laboratory
Department of Computer Science and Engineering
University of California, San Diego
La Jolla, California 92093
e-mail: margarit@cs.ucsd.edu, polyzos@cs.ucsd.edu

Abstract - We evaluate MobiWeb, a proxy-based network architecture designed to enhance the performance of adaptive real-time streams over wireless Internet links. MobiWeb includes a priority scheme that preserves media smoothness despite short-term fluctuations of the link, as well as an adaptation mechanism applied to streams responding to long-term link changes. A set of timers permits the rapid exploration of the link, increasing the utilization of resources when they are abundant, while forcing enhanced streams to back-off to more moderate resource usage when resources are scarce. MobiWeb uses admission control for real-time traffic in order to provide a base level of quality, while remaining fair and transparent to unaware, best-effort traffic. This paper validates the enhanced performance of adaptive real-time streams when MobiWeb is used, through the evaluation of several scenarios simulated on NS – the Network Simulator tool.

Keywords: Wireless networks, adaptive applications, real-time streams, proxy, NS-Network Simulator

Introduction

We are now witnessing the rapid emergence of multimedia applications utilizing real-time streams over the Internet. Videoconferencing, video-on-demand and Internet Phone are a few to mention. Even mainstream carriers started lately to favor the Internet solution, transmitting voice in packets through statistically multiplexed Internet connections rather than the traditional circuits (G3 1999; Net2Phone 1999).

The transmission of real-time streams over packet-switched wire-line networks is not a novelty anymore and several proposed techniques and optimizations improve it substantially. However, the use of wireless channels can severely impact real-time streams' performance, because of the unpredictability of the link quality. Real-time streams are known to demand large amounts of bandwidth and impose stringent timing requirements. Although development of new wireless technologies leads to devices that work with data rates of a few Mbps, they still are bandwidth limited, compared to similar wired devices and technologies, which imposes difficulties in accommodating demanding real-time applications. Thus, bandwidth and delay constraints remain key issues for the transmission of real-time streams over wireless channels.

The other important issue is dealing with the significant and unpredictable error rates of wireless links. Atmospheric conditions, electromagnetic interference, physical obstacles, multi-path propagation, and other phenomena interfere with transmissions over the wireless channel, effectively introducing bursts of errors. Many real-time applications can tolerate scarce and scattered errors, but long lasting error bursts severely impact their presentation quality, forcing larger application units, e.g., video frames, to be dropped, significantly affecting the user perceived quality and possibly even shutting down the application for a noticeable period of time.

In order to overcome these problems, we are developing *MobiWeb* (Margaritidis and Polyzos 1999), an Internet-based communications architecture that intends to support the seamless operation of real-time streams over wireless links. *MobiWeb* uses an inter-stream priority scheme to address the problems introduced by wireless links. Each adaptive stream is assigned a priority and a portion of the link's resources, which are protected from being used by lower priority streams. Priorities can change dynamically to reflect the relative importance of each stream, as new streams are initiated and old ones are terminated or as the focus of the users and their actions change. The adaptation interface of *MobiWeb* allows applications to indicate their preferred adaptation mechanism to be applied onto a stream in order to change its traffic and presentation characteristics. The interface provides enhanced features for adaptive applications, while it remains transparent and fair to traditional best-effort traffic.

In addition, *MobiWeb* shields the application's performance from the rapid changes and errors of the wireless channel by utilizing a relaxed adaptation method. It uses a set of timers that capture the current condition and the dynamics of the channel, providing the necessary information for *MobiWeb* to decide whether a change in the quality is temporary or permanent. Such a decision assists applications to quickly

explore available resources and backoff during severe link degradation, while retaining their steady state operation during short and temporal quality fluctuations.

The next section of this paper describes the novel features of *MobiWeb* for supporting real-time streams over wireless links. Following is a *presentation* of the *MobiWeb* architecture and implementation. The subsequent section discusses several simulation tests that prove *MobiWeb's* validity and effectiveness. Finally the last section summarizes and concludes this paper.

MobiWeb's features

MobiWeb's first important feature is the provision of support for adaptability. As mentioned before, the limited resources of a wireless channel can severely impact the performance of a traditional, non-adaptive real-time application. Moreover, the multiplexing of other traffic through the same link leaves fewer resources available for a demanding real-time stream. If the stream cannot adapt, it has to either drop a significant amount of packets, which results in jerky performance, or wait and transmit later, which unacceptably delays the playback of the application.

MobiWeb's support for adaptability comprises a set of Levels of Quality (LoQ) for each stream and a priority value associated with each one of them. During the initiation of the adaptive stream, the application provides *MobiWeb* with the characteristics of each LoQ of its operation. This includes the priority value, the bandwidth and delay requirements, as well as the mechanism of adaptation. This mechanism can be a filter to be applied on the stream or a callback to the application that can adapt the stream from its source. The initiation of the stream is subject to admission control, since *MobiWeb* is expected to provide minimum guarantees of operation to real-time streams. These guarantees are selected to be soft, since providing hard guarantees, especially in a wireless environment is impossible due to the unpredictability of the availability of resources. Thus, admission is based on the average statistical requirements extracted from the lowest LoQ of the stream.

During its operation, *MobiWeb* monitors the performance of each application along with the constantly changing channel quality. When new streams are initiated or old ones are terminated and when the channel's quality changes, *MobiWeb* adapts the transmission of streams according to their adaptation method utilizing the appropriate adaptation mechanism. The selection though of which stream to adapt is a challenging and non-straightforward task. Other proposals for adaptation leave the issue of selection open or let all streams adapt, unnecessarily affecting the performance of most of the streams (Angin *et al.* 1998; Inouye *et al.* 1997; Satyanarayanan *et al.* 1997).

MobiWeb instead uses a fair and intelligent method to select the next candidate for adaptation. Whenever a stream is initiated, it does so in its lower quality of operation and its higher priority value. As long as resources are available the stream advances LoQs and its priority is reduced. When adaptation is necessary, the stream subject to it will be selected from those with the lower priority and, accordingly, the higher quality and the most utilized resources. This ensures that each stream will retain its base quality of operation as long as there are enough resources for the base quality of all existing streams. If the degradation is so drastic that some streams are forced to adapt below their base LoQ, those streams are terminated so that the others can survive.

MobiWeb remains operationally transparent to traditional best-effort traffic, even though the performance of this traffic is affected by the priority structure introduced. Because of the nature of best-effort traffic, admission control is not performed and all such streams are accepted. Even though LoQs are not applied in this case, each best-effort stream is assigned a common predefined priority value. This is selected to be the median of the range of available priority values. Adaptive applications are usually initialized with higher priority in order to give themselves precedence over the best-effort traffic. It is expected though that after some advances in LoQ, their priority will equal that of best-effort traffic, giving traditional applications enough resources to operate effectively.

Another of *MobiWeb's* important features is its ability to shield the real-time streams from the short-term fluctuations of the wireless channel. Real-time applications are capable of tolerating occasional errors without a noticeable effect in their performance. But as mentioned previously, they must adapt to drastic channel changes to retain their performance to an acceptable level. Those two conflicting requirements are not known to be captured from any of the schemes available in the literature. The original solution of not adapting at all allows the stream to tolerate scattered errors, but it is ineffective over a wireless channel. Subsequent solutions are forcing the streams to immediately adapt to changes (Fox *et al.* 1996). This results in streams that constantly adapt their performance following the channel changes. Since real-time streams perform best when they are in a steady state of operation, even if that is not the one with the best quality, the best solution is a scheme that will keep the benefits of both previous solutions. Some schemes introduced in this direction so far are designed to accommodate static, non real-time traffic (Bharghavan *et al.* 1998). *MobiWeb's* solution is focused mainly on real-time streams, while preserving functionality and fairness for the traditional traffic. It does so with the use of a pair of specialized timers that assist *MobiWeb's* decisions on whether to adapt a stream or not.

MobiWeb shields small fluctuations on the wireless channel from the adaptive applications by using a relaxation timer. *MobiWeb* checks the performance of each stream over short periods of time. If at the end of such a period a stream did not receive its assigned resources, its relaxation timer is triggered. The duration of this timer spans over a few time periods, giving the stream the opportunity to recover without falling back from an occasional channel error. Expiration of the timer without improvement is translated as severe degradation of the channel, which forces the stream to adapt. The actual value of this timer must be carefully selected during the setup of *MobiWeb* in order to retain its functionality. In addition, a change in the transmission environment (i.e. a handoff to a different wireless

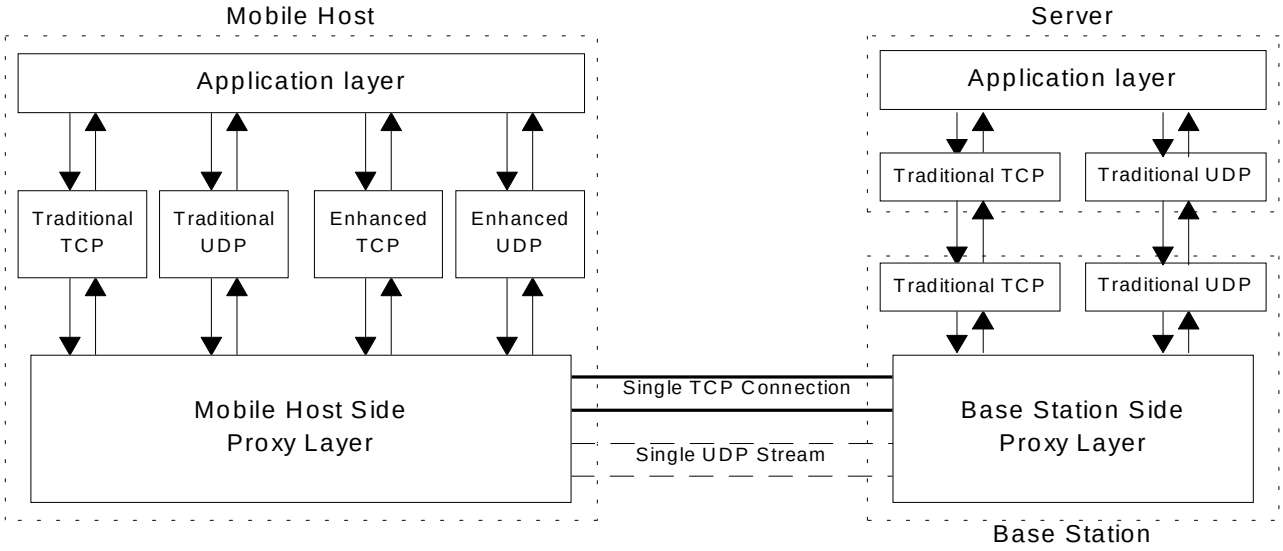

Figure 1. The Proxy-Based Architecture of MobiWeb.

medium) requires a recalculation of the timer value for it to reflect the inherent characteristics of the new medium.

When the channel is fairly empty or an existing stream is terminated and resources are freed, the adaptive applications are expected to discover and utilize them immediately. Thus, the applications must probe the channel to decide whether the available resources can accommodate their next LoQ. The timing and the frequency of the probing can have severe implications for the performance of the adaptive applications. If all streams probe the channel simultaneously, they will overflow it and none of them will claim the available resources. Instead, every probing stream will have to immediately adapt backwards, causing unnecessary fluctuations to its performance. Moreover, if each stream probes the channel very often, in order for the excess resources to be utilized instantly, it is impossible for them to reach a desired steady state of operation. On the other hand if the probing is infrequent, recently freed resources might remain unused for an unacceptably long period of time.

MobiWeb alleviates the above concerns with the use of the exploration timer and the exploration mode of operation. As in the case of the relaxation timer, *MobiWeb* checks the performance of each adaptive stream over short periods of time. As long as the application received all the resources assigned to it during the last time period, the exploration timer is triggered. If the quality supported by the channel remains the same until the timer expires, *MobiWeb* adapts the stream to its next available (if any) LoQ. This action puts this stream into the exploration mode of operation. During this mode *MobiWeb* decides whether the newly utilized excess resources are enough to fully accommodate this higher LoQ of the stream. The stream remains in the exploration mode for a period of time significantly higher than the duration of the relaxation timer. If the resources are enough to accommodate this LoQ, the relaxation timer will not trigger, allowing the stream to exit the exploration mode and continue its normal operation. If the resources are not enough though, the relaxation timer will trigger and expire first and *MobiWeb* will adapt the stream back to its previous LoQ.

Such a fallback during the exploration mode is interpreted differently compared to a typical fallback caused by channel degradation. In the exploration case resources are available but not enough to advance a LoQ, so the next exploration timer is set to be exponentially higher than the previous one. This will put the stream less often in the exploration mode, effectively allowing it to enter a steady state of operation. In the other case though, fallback was caused by interference from other streams forcing this stream to give up some of its resources. In order to preserve fairness, *MobiWeb* sets the next exploration timer for this stream to its lower value so that it will be the first to acquire any resources when they will become available in the future.

The concurrent operation of the relaxation and the exploration timer permits *MobiWeb* to combine the advantages of both techniques mentioned above. It can protect the performance of a stream from short fluctuations, while allowing fast exploration of available resources and a steady state operation during stable channel conditions.

MobiWeb's architecture

MobiWeb is intended to be deployed locally, at the two ends of the wireless channel. In a typical setup, a Mobile Host (MH) is connected with the Internet via a Base Station (BS) by using a wireless, shared or point-to-point, channel. The BS uses a wired network to relay traffic to the Internet, so the point of contention is expected to be the wireless link between the BS and the MH. Such a setup favors a local solution for two reasons. First, it allows for fast and seamless deployment over the Internet, since only two nodes have to be modified instead of the whole path from the source to the destination. Second, *MobiWeb's* decisions are based on local channel

characteristics, which is very difficult to capture with an end-to-end solution.

MobiWeb's architecture is based on the proxy model, as illustrated in figure 1. Two symmetric proxy layers are added to both the Mobile Host (MH) and the Base Station (BS) at the two ends of the wireless channel. Vertically, the proxy layer is inserted between the application and the transport layer of the current Internet protocol stack. At both ends the proxy layer intercepts each TCP or UDP stream from an application. If the application is *MobiWeb* aware, the proxy layer applies first the appropriate adaptation mechanism on each stream initiated from this application. This effectively adapts the stream's performance according to the currently selected LoQ. If the stream is initiated by a traditional application, each packet is only checked for time expiration and is being queued, while the default priority for best-effort traffic is assigned to it.

MobiWeb's scheduler is responsible for arranging the actual transmission of the packets. In order for the scheduler to be able to apply the prioritization scheme, a single TCP connection and a single UDP stream are permanently open between the MH and the peer BS (Floyd and Housel 1998). The reason for this is twofold. First, it eliminates setup overhead between the two ends for each new TCP connection. Second and more important, it enables the scheduler to enforce the priority scheme. *MobiWeb* is designed to be dynamic and transparent to traditional traffic, therefore compatibility with the current TCP and UDP protocols is imperative. Since there are no appropriate provisions for priorities for TCP/UDP traffic, the transmission of multiple streams through a single TCP connection or UDP stream is the best solution for realizing the prioritization scheme.

The selection of the next stream to be served is based on the prioritization scheme. *MobiWeb* breaks down time into short time periods and enforces the scheme in each period. Streams with the highest priority take turns in transmitting until their assigned resources are consumed. Following them, the streams belonging to the next level of priority start transmitting, until all streams are satisfied or the time period ends. In the first case, streams that still have data to transmit, do so in a round-robin fashion until the time period ends. In the second case, some of the lower priority streams received fewer resources than the amount that was assigned to them. That triggers their relaxation timer, which will force them to eventually fallback to a lower LoQ.

After a stream is selected for transmission, each packet is been encapsulated in a new packet that carries the stream's ID before transmission through the single TCP or UDP stream. The peer *MobiWeb* proxy decapsulates the packet, classifies it according to its flow ID and routes it to its final destination, while it updates the stream's performance statistics in the case of an adaptive one.

At the end of the time period of transmission, *MobiWeb* checks the performance statistics gathered for each stream and initiates the appropriate adaptation mechanism, if required. The adaptation mechanism depends on the nature of the transmitting application. *MobiWeb* aware applications are expected to provide callbacks to the proxy layer or a set of filters, each one associated with a LoQ. In the first case the application itself regulates the performance of the stream according to the information received from the callback. Such an implementation proves to be quite costly, since each application has to integrate the adaptation mechanisms. Moreover, this solution is preferable only when the transmitting end is close to the wireless link, since the time interval introduced by the round trip delay of the callback can severely impact the performance of the stream before it is able to adapt.

The second case is more general and it is based on the use of a set of media transformation filters. The filters are applied to the stream and transform it in a way that can be transmitted utilizing the assigned link resources. For example a video filter can scale down the resolution or the color depth or both. An audio stream can be re-sampled, using a lower frequency or using fewer bits per sample. The advantage of using filters is that they can be implemented once for each medium type and get standardized and utilized by several different applications. The drawback though is that they increase the computation complexity of *MobiWeb*, since *MobiWeb* is now responsible for the real-time transformation of each stream. *MobiWeb* is designed to support both mechanisms of adaptation, providing flexibility and support for the best solution that applies to each configuration.

In addition, *MobiWeb* has an alternative mode of operation. In cases where the MH is a lightweight device, the implementation of the proxy layer at the MH end can be unaffordable, both from computational complexity and from power consumption perspectives. In such a configuration the proxy layer at the MH can be omitted. *MobiWeb* still operates at the BS and manages only the incoming streams to the MH, while the outgoing streams use the traditional Internet transport methods. This solution can be acceptable because a lightweight device is expected to be used to mainly receive real-time media rather than transmit much data, in which case the proxy layer at the MH is basically redundant. For example, a small PDA can probably be used to receive audio and video over the Web, but might not be expected to participate in full-blown videoconferences.

Finally, note that the proxy layer uses cross-layer information in order to operate properly. The interface to the application layer is similar to a typical transport layer one, with the only difference being that it is enhanced with optional fields for adaptive applications, in order for them to be able to assign priority values to each of their streams. They can also provide an association between each LoQ of their operation and each filter or callback that the proxy will use, in order to transform the stream to meet this LoQ.

At the lower end, *MobiWeb's* scheduler requires information about the current quality of the wireless link, in order to decide which streams it will allow to transmit and which it will not. The channel statistics can be gathered by the link layer and recorded continuously in files directly accessible by the scheduler, e.g., as suggested in (Polyzos and Xylomenos 1998). These statistics not only affect the decisions of the scheduler as part of the short term drop packet policy, but also play an important role in the admission policy for new streams and in the long term adaptation of the existing *MobiWeb* aware applications.

Evaluation

To evaluate the performance of *MobiWeb*, we performed several simulation tests, two of which are presented here. The simulations were executed by NS – the Network Simulator, a tool that was designed for the VINT Project (VINT 1998) and is freely distributed by UC Berkeley (NS 1998). NS's open architecture provided the methods to integrate additional modules, necessary to implement the functionality of *MobiWeb* and produce simulation traces used to create the graphical chart interpretations of the tests.

The purpose of the performed simulations is to illustrate *MobiWeb's* ability to assist the adaptation of real-time streams, while being transparent to traditional traffic. Therefore, the simulation configuration was designed to introduce three streams simultaneously over the same channel: a video stream, an audio stream and an FTP transfer. The three streams terminate at the end of a single wireless link, with a nominal data rate of 1.6 Mbps, as depicted in figure 1. The *MobiWeb* proxy agent is installed on node 2 and is responsible for managing the packets that are destined for node 1.

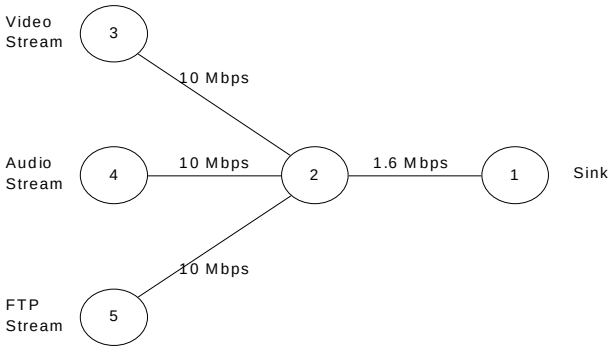

Figure 1: Simulation Topology

The real-time streams are both initiated by adaptive applications. The audio stream has a base layer of 400 Kbps and it can advance two more layers, each adding 200 Kbps to the total required bandwidth. The video stream has a base layer of 750 Kbps and the addition of an enhancement layer leads to a total of 1 Mbps transmission rate. In each simulation the video stream is initiated 5 seconds after the audio and the FTP streams, and is deliberately terminated earlier than them, in order to study the effects that both the entry and the departure of a stream has to the remaining ones.

The first simulation test reflects a scenario where the real-time streams are required to have absolute priority over the FTP transfer. Since the range of priority values is set from 0 (lower) to 4 and the FTP, being traditional traffic, is assigned a value of 2, both the real-time streams are started with priority 4. As figure 2 shows, the audio stream advances immediately its two LoQ levels to reach peak rate and stay there until the video stream is initiated. The FTP stream utilizes the available bandwidth, but it has to fallback and effectively shut down when the higher priority video stream starts transmitting. At this moment the audio stream has priority 2 and the video 4, so the first one gives up a level while the second advances one, so that both of them reach a stable level of operation, where they utilize all the link's bandwidth unaffected from the presence of the FTP stream. It is important to notice that *MobiWeb* recognizes that the link is fully occupied and prevents both streams from attempting to advance further.

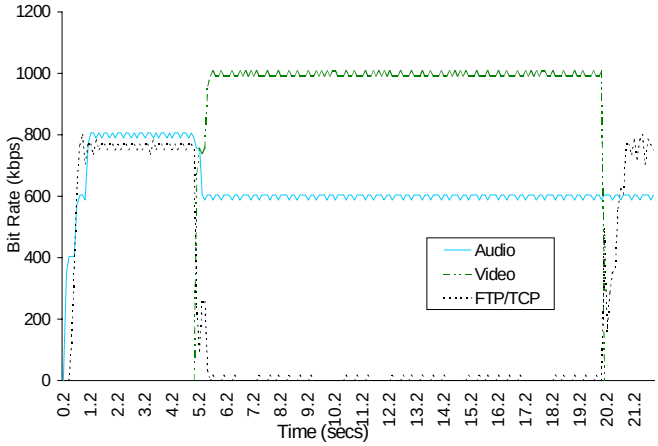

Figure 2: Trace from the first simulation

The second simulation scenario is slightly different from the first, in that some resources from the higher quality video steam are sacrificed in favor of other traffic and in particular of the FTP transfer. This is accomplished by initializing the video stream with a priority of 3 instead of 4. Thus after its first advancement its priority equals that of FTP, with which it shares the link resources available at this priority level. Consequently every attempt from the video stream to advance a LoQ leads to a failure, since it cannot claim enough resources to accommodate that level. As figure 3 shows, these attempts are frequent at the beginning but become scarcer later, to eventually let the stream reach a stable state of operation.

On the other hand, the audio stream falls back a LoQ when the video stream starts, though it immediately probes the channel to claim it back, since *MobiWeb* indicates that there are available resources not reserved for adaptive applications. The audio stream's attempts have the same effect as those of the video stream, since once again it has to share the available resources with the FTP transfer and potentially with the video stream too. This is clearly depicted in figure 4, which is a snapshot of figure 3 around the first 8 seconds of the simulation. At times 5.5 and 6.6, the video stream first and then both the adaptive streams, unsuccessfully probe the channel for a higher LoQ. The spikes in the plot associated with the FTP transfer at the same time marks represent brief opportunities for the FTP to transfer more packets, preventing this way the other streams from claiming resources at this priority level. After a few unsuccessful attempts, the audio stream reaches a stable state of operation, in which it stays until the video stream is terminated. This allows the audio stream, as well as the FTP one, to advance a LoQ by successfully utilizing the freed resources.

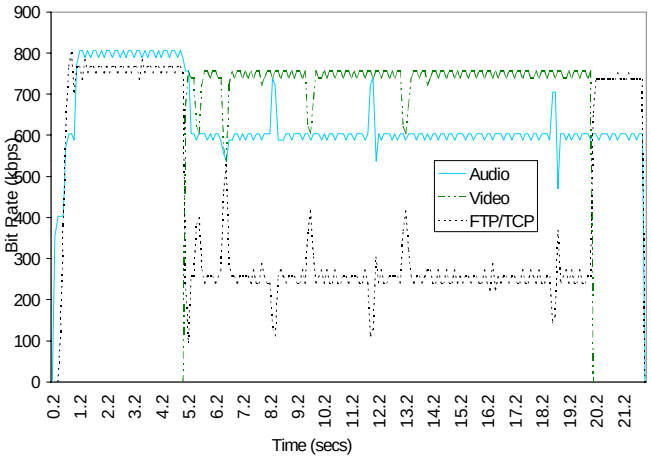

Figure 3: Trace from the second simulation

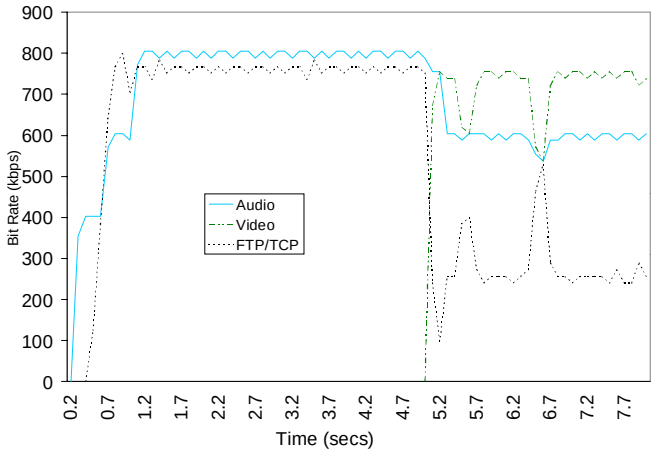

Figure 4: Adaptation with MobiWeb

Conclusions and future work

The simulation results demonstrated that *MobiWeb* substantially enhances the performance of adaptive real-time streams over limited wireless links. When a stable state of operation is required, the priority scheme can provide the shielding that the performance of a real-time stream needs from other traffic. *MobiWeb* remains fair though to traditional traffic, when adaptive streams attempt to claim excessive resources without having a higher priority, by fairly sharing the link between all competing applications.

When resources are available, *MobiWeb* uses its adaptation mechanisms to assist existing streams in rapidly exploring those resources. However, when the link is fully utilized, the unsuccessful exploration attempts lead eventually the adaptive real-time streams to a desired steady state of operation, unaffected from short-term fluctuations in the quality of the channel.

The simulation tests presented in this paper were performed to evaluate the unique features of *MobiWeb* over a fairly stable wireless link. We intent to further exploit the usability of *MobiWeb* by enhancing our simulations to incorporate error models that reflect the real environment conditions that exist on a wireless link.

References

Angin, O.; A.T. Campbell; M.E. Kounavis; and R.R.-F. Liao. 1998. “The mobiware toolkit: programmable support for adaptive mobile networking.” *IEEE Personal Communications*, vol. 5, no. 4, (Aug.): 32~43.

Bharghavan, V.; L. Kang-Won; L. Songwu; H. Sungwon; L. Jin-Ru; and D. Dwyer. 1998. “The TIMELY Adaptive Resource Management Architecture.” *IEEE Personal Communications*, vol. 5, no. 4, (Aug.): 20~31.

Floyd R. and B. Housel. 1998. “Mobile Web Access Using eNetwork WebExpress.” *IEEE Personal Communications*, (Oct.): 47~52.

Fox, A.; S.D. Gribble; E. A. Brewer; and E. Amir. 1996. “Adapting to Network and Client Variability via On-Demand Dynamic Distillation.” In *Proceedings of the Seventh International ACM Conference on ASPLOS* (Cambridge, MA, Oct.).

G3 Global Gateway Group. 1999. http://www.gxc.com

Inouye, J.; S. Cen; C. Pu; and J. Walpole. 1997. “System Support for Mobile Multimedia Applications.” In *Proceedings of the 1997 NOSSDAV* (May). 143~154.

Margaritidis, M. and G.C. Polyzos. 1999. "Application-Assisted Adaptation of Real-Time Streams over Wireless Links." To appear in *Proceedings of the Second Annual UCSD Conference on Wireless Communications*, (San Diego, California, Mar.).

Net2Phone. 1999. http://www.net2phone.com

Network Simulator. 1998.
http://www-mash.CS.Berkeley.EDU/ns/ns.html

Polyzos, G.C. and G. Xylomenos. 1998. “Enhancing Wireless Internet Links for Multimedia Services.” In *Proceedings of the 5th International Workshop on Mobile Multimedia Communications* (Berlin, Germany, Oct.). 379~384.

Satyanarayanan, M.; B. Noble; D. Narayanan; J.E. Tilton; J. Flinn; and K.R. Walker. 1997. “Agile Application-Aware Adaptation for Mobility.” In *Proceedings of the 16th ACM Symposium On Operating Systems Principles*, vol. 31, no. 5, (Dec.): 276~287.

VINT Project. 1998. http://netweb.usc.edu/vint/