MOBIWEB: ENABLING ADAPTIVE CONTINUOUS MEDIA APPLICATIONS OVER WIRELESS LINKS

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 92093-0114

***Abstract* - *MobiWeb* is 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 and handoffs. 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 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.**

I. INTRODUCTION

The major trend in the personal communications industry's efforts these days is to provide the end user with ubiquitous access to the Internet. At the same time, Web content providers want to offer uninterrupted services to the client, even when mobility is involved. The most frequently accessed Web sites, however, tend to be rich in real-time multimedia content and they were designed having in mind the abundance in resources that the wired Internet infrastructure can provide. It is the capabilities of this infrastructure that made possible the wide acceptance and utilization of real-time applications like videoconferencing, IP telephony, video on demand, etc.

Attempting to expand the access to this kind of applications into the wireless domain poses some important problems. In contrast to the core (wired) Internet infrastructure, wireless links suffer from limited resources and constantly variable characteristics. The available bandwidth can be orders of magnitude less, the imposed delay much higher and the loss rate orders of magnitude more. Furthermore, these characteristics can vary unpredictably due to frequent handoffs and various phenomena like atmospheric conditions, electromagnetic interference, physical obstacles and multi path propagation. Such conditions significantly affect the performance of real-time applications. These applications demand large amounts of, usually variable, bandwidth, pose stringent delay constraints and suffer from bursty errors. Moreover, they perform better with long-term operational stability.

Traditional Internet transport protocols, like TCP and UDP, were not designed with the peculiarities of the wireless links in mind. Therefore, the performance of real-time applications over them is severely degraded. RTP [1] is a step towards the right direction, in the sense that it allows the server to tailor the transmission of the stream according to the path's characteristics, through feedback messages sent periodically from the client to the server. The protocol, however, has several drawbacks since it cannot always cope immediately and effectively with link fluctuations due to its end-to-end nature and more importantly cannot prevent the inter-stream interference from streams competing for the resources of the wireless link.

This situation has ignited a burst of research in this area, which produced several solutions that attempt to overcome the aforementioned problems [2, 3, 4, 5, 6, 7 and 8]. Some of the solutions have concentrated in solving only part of the problem and some of these are lacking several key features. In addition, the newest standard in wireless access to the Internet, the wireless application protocol (WAP) [9], is still in very early stages of development and it doesn't provide any support for real-time multimedia applications. This has motivated us to develop *MobiWeb* [10, 11].

II. DESIGN GOALS

MobiWeb is a proxy-based network architecture designed to assist the adaptation and enhance the performance of real-time streams over wireless links. Providing support for adaptation is one of *MobiWeb's* major goals. A traditional continuous media application uses a single representation that has stringent bandwidth and timing requirements. Thus, when transmitted over a variable wireless link, it is susceptible to loose a significant amount of packets and/or frames, resulting in significant degradation of its quality. An adaptive application, on the other hand, defines several different representations. Its execution is separated in a set of Levels of Quality (LoQ). Each LoQ represents the transmission characteristics

required by the application in order to achieve the performance associated with it. The network uses this knowledge, along with the adaptation policy and the current link condition, to decide whether the application must adapt or not. This allows the application to gracefully degrade its perceptual quality to an acceptable to the user level.

MobiWeb supports adaptive applications through the means of proxies residing at the two ends of the wireless link, as illustrated in figure 1. Filtering mechanisms tailor each stream according to the characteristics of the wireless link and the capabilities of the mobile device. Selecting the proxy solution over an end-to-end one is imperative in order to minimize the response time to the variations of the link and optimize the adaptation performance. Moreover, a local solution requires minimal changes to the base station and the mobile client, in contrast with an end-to-end solution, which demands significant and costly changes to the server and potentially to the entire end-to-end path. Finally, it's worth mentioning that a proxy solution is more suitable for multicasting scenarios. The source does not have to be throttled down to the characteristics of the least potent client, which would have forced the rest of the clients in the same multicasting tree to receive the stream with the same (low) quality.

Another important goal is to prevent inter-stream interference incurred from the contention between streams for the limited link resources. Real-time applications require a Quality-of-Service that can be guaranteed by reserving a certain amount of resources throughout their execution, in contrast to best-effort applications that can operate utilizing any amount of resources. The existing solutions either lack any support for guarantees or use, at best, a weighted-fair sharing algorithm that proportionally splits the link resources between the existing streams. These solutions are inadequate for constantly variable wireless links, since resources not only need to be allocated for each transmitting stream, but they also need to be "protected" from fluctuations in link quality. Whenever degradation occurs, proportionally sharing the resources leads to smaller shares for each stream, forcing all of them to adapt at the same time. Preventing such inter-stream interference, however, forces the streams to adapt one at a time, reducing the overall adaptation workload and retaining the stability in execution for most of them.

MobiWeb contributes a unique solution that allows only one stream at a time to adapt until no further adaptation is necessary. Such a solution is based on the combination of an admission control algorithm and a dynamic prioritization scheme. Admission control reserves resources for each stream in order to guarantee at least the minimum acceptable level of quality in the stream's performance. The dynamic prioritization scheme protects

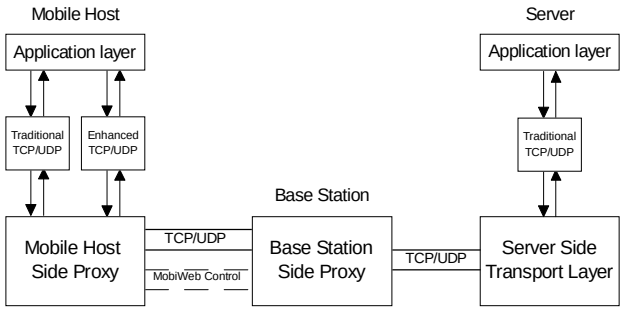

Figure 1. The Proxy-Based Architecture of MobiWeb.

the higher priority streams from losing their shares of resources to streams with lower priority. In addition, when degradation in the link quality occurs, the adaptation process will start with the stream that has the lowest priority and it will continue as long as more resources need to be freed. In contrast, a static prioritization scheme would have favored those applications that were initiated with a higher priority and it would have been unfair to the rest of them. *MobiWeb's* dynamic prioritization scheme relates the priority of a stream inversely proportionally to its Level of Quality (LoQ). Whenever a stream is forced to adapt backwards by dropping a LoQ, its priority is increased by a level and vice versa. This scheme provides robustness and fairness to streams that are already receiving a low LoQ, since they most likely have a high priority level.

Besides long-term stability, *MobiWeb* also provides shielding in the performance of real-time applications from short-term variations in the wireless link. The adaptation process must be robust to these fluctuations and avoid forcing the applications into frequent and meaningless adaptations. Drastic and significant changes, however, in the link quality must still alert the applications to immediately adapt to the new conditions, while fairly stable conditions must prevent them from frequently probing the link for a different LoQ. Previously proposed mechanisms to overcome this problem include the specification of a range in allocated bandwidth, within which the application can tolerate variations without having to adapt [4, 5, 6]. This solution succeeds in covering short-term variations of small magnitude. It interprets, however, a short but abrupt degradation in quality as a reason for adaptation, failing in this case to retain a steady state operation. In *MobiWeb* we experiment with a novel approach, utilizing a set of specialized timers that reflect the current quality of the link. These timers provide the short-term shielding in the time dimension rather than in the bandwidth dimension. They dynamically adjust to the current link conditions, which permits them to absorb all short-term variations, while allowing fast adaptation when resources become available and steady state operation when the link is fairly stable.

Finally, *MobiWeb* aims in remaining transparent to traditional best-effort Internet traffic and in providing a simple interface for the user. The scheme has to be able to be deployed gradually and transparently to the existing Internet infrastructure in order to be a viable solution. Concurrently, the user must have the flexibility to declare his preferences regarding all the streams he receives, but the actual adaptation mechanism should be hide from him. Even though most of the existing schemes remain transparent to traditional traffic, they require the user's involvement in the initiation of the adaptation process every time a change occurs. In contrast, *MobiWeb* not only transparently accommodates best effort traffic, but also provides a simple yet powerful interface to the user, successfully hiding the adaptation details.

III. MobiWeb's Features

A. Architecture

MobiWeb is intended to be deployed locally, between or near the two ends of the wireless link. In a typical setup, a Mobile Host (MH) is connected with the Internet via a Base Station (BS) via a wireless, shared or point-to-point, link. The BS uses a fixed network connection 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 several reasons. First, it allows for fast and seamless deployment over the Internet, since only two nodes have to be modified instead of the entire end-to-end path. In addition, capturing the local link characteristics is accomplished substantially easier with a local than an end-to-end approach. Finally, having the point of interaction close to the client side is essential for fast and effective adaptation. In an end-to-end solution, feedback describing the wireless link performance may arrive too late at the source, for it to adapt before flooding the path's bottleneck link.

MobiWeb's architecture is based on the proxy model. Two symmetric proxy layers are added to both the MH and the BS at the two ends of the wireless link. Vertically, the proxy layer is inserted above the transport layer of the current Internet protocol stack providing an enhanced transport interface to the application layer as figure 2 shows. If the application is adaptive, it can use this interface to take advantage of *MobiWeb's* functionality. Otherwise, the proxy layer will still accommodate traditional Internet applications, without though the benefits of adaptation.

Packets from adaptive applications are marked with their respective priority, which is utilized by *MobiWeb's* dynamic prioritization queuing algorithm. At the same time, the link layer determines the transmission rate of the packets based on knowledge about the link conditions received from lower layers. Such knowledge consists of

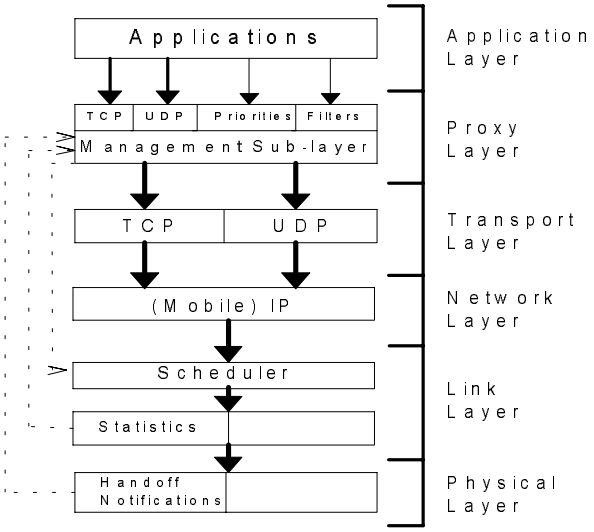

Figure 2. The Proxy organization and interface.

statistics about throughput and error rates of the link, transmission and reception power characteristics, and handoff notifications. This information is also propagated upwards to the management sub-layer, which is responsible for the accounting of the available and utilized resources, as well as the admission of new streams.

B. Admission of new streams

In order to assure performance guarantees, *MobiWeb* uses admission control applied only to adaptive real-time streams. Admission is initiated in two cases. The first is when the mobile moves into a new cell and the second is when a new stream attempts to initiate transmission. Admission is performed based on the amount of resources necessary for the stream to operate with its lowest acceptable quality. With this approach, we keep *MobiWeb's* complexity low, while we retain the functionality of the admission control. In addition, new real-time streams can still be initiated, even when the link is currently fully utilized, as long as all streams are able to operate with at least their minimum Level of Quality.

The management sub-layer of the proxy keeps track of the aggregate resources that the current streams require to operate with their minimum quality. It also keeps track of the nominal rate and the recent loss rate history of the current wireless link. Combining this knowledge it estimates the current measured resources available to the admission control. When a new stream is initiated, its minimum requirements are compared with the subtraction residue of the current measured resources from the aggregate sum of minimums. The new stream is accepted only if the residue is greater in this comparison.

C. Dynamic prioritization

Each newly initiated stream is associated with a priority value ranging from 0 to 9 (highest). The user provides the initial priority for each stream, indicating the relevant importance of this stream against the others. In addition, traditional non-adaptive streams are initialized with a default priority value (1) that allows them to participate in the dynamic prioritization scheme, even without adapting. In contrast to point-to-point links, assigning priorities in a shared link, like WLAN, requires the presence of incentive enforcement scheme that prevents the user from abusing the link. We currently examine the utilization of a costing scheme as a solution for this problem.

A new stream always starts transmitting with its lowest quality, until it gradually advances to a higher one. The lowest quality of the stream is associated with its highest priority. As long as the stream advances to a higher quality, its priority drops to a lower value. In contrast, during degradation the stream backs off to a lower quality but it is now associated with a higher priority.

The reason for applying this method is because we believe that the mentality of a user is to prefer every stream to perform fairly well rather than have only one stream performing well at the expense of the others. Thus, advancing to a higher quality means also an increase in the possibility that the stream will be a candidate for adaptation during the next degradation period. This protects the streams that already receive a low quality from further degradation. In other words, a stream that was just degraded will not be forced to adapt again, before all other streams, which previously had the same priority with it, adapt at least once. Moreover, higher priority means that when link conditions upgrade, the stream with the lowest quality will be the first to explore the available resources until it advances to a higher LoQ. This will also cause its priority to drop making another stream the one with the lowest priority, which now takes turn in exploring the resources, and so forth.

D. Specialized timers

The quality variations of the wireless link can be classified into two categories. First, there are short-term variations caused by the transmission characteristics of the air interface, which worsen when mobility is involved. Second, there are long-term variations caused mainly by handoffs and initiations/terminations of streams. Consequently, *MobiWeb's* major consideration is to enable the shielding of real-time streams from the short-term variations in link quality, while allowing for fair, fast and efficient adaptation to long-term ones.

MobiWeb shields short-term fluctuations by using a *relaxation timer* for each adaptive stream. The proxy layer at the transmitting end of the link 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 allocated resources, its relaxation timer is triggered. The duration of this timer spans over a few time periods, giving the stream the opportunity to recover from a few occasional link errors, without falling back to a lower LoQ. The expiration of the timer without improvement in quality is translated as permanent degradation of the link, which in turn forces the stream to adapt.

Conversely, when the link is fairly unoccupied or an existing stream is terminated, but mainly when the mobile host handoffs to a better wireless interface, excess resources are available and adaptive applications are expected to discover and utilize them immediately. Thus, the applications must constantly probe the link to decide whether there exist available resources able to accommodate their next LoQ. However, the timing and the frequency of the probing can have a severe impact upon their performance. *MobiWeb* alleviates the above concerns with the use of the *exploration timer* and the *exploration mode* of adaptation for each adaptive stream. If the application received all the allocated resources during the last time period, the exploration timer is triggered. As long as the quality supported by the link remains the same until the timer expires, *MobiWeb* adapts the stream to its next higher (if any) LoQ. This action puts this stream into the exploration mode of adaptation. During this mode, *MobiWeb* decides whether the newly utilized excess resources are enough to fully accommodate the higher LoQ of the stream.

In the exploration mode, when resources are available but not enough to advance a LoQ, 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 operation. However, after a handoff to a wireless link with better quality or after the termination of an existing stream, enough resources may be available for several streams to upgrade to a higher LoQ. In this situation, *MobiWeb* rushes the adaptation process by forcing the next candidate stream for exploration to adapt immediately while reducing the exploration timers of the rest of the streams. In conclusion, *MobiWeb* protects the performance of a stream from short-term fluctuations, while allowing fast exploration of available resources and a steady state operation during stable link conditions.

E. Adaptation mechanisms - Filtering

MobiWeb uses filtering as the adaptation mechanism for adaptive applications. Filters are small pieces of transformation code inserted into the proxy layer as plug-ins. Due to their position in the transmission path they illustrate several important properties. They can be changed almost immediately when adaptation occurs, they

TABLE 1
MPEG STREAMS USED IN SIMULATION SCENARIOS

Name	Duration	LoQs in FPS	Rates for each LoQ (Kbps)	Total frames	GOP
M1	80 sec	2 – 8 - 24	85K, 273K, 625K	2419	IBBBPBBBPBBBPBBB
M2	30 sec	2 – 10 - 30	215K, 514K, 785K	886	IBBPBBPBBPBBPBB
M3	44 sec	2 – 10 - 30	184K, 520K, 819K	1314	IBBPBBPBBPBBPBB

require no modifications of the server and they perform optimally in multicasting scenarios [12].

Filters can be provided by the application, the proxy layer itself or by a general filter repository residing in an Internet filter server. During the initiation of the stream, the application indicates the filters that will be used for each LoQ. The proxy layer can have a set of generic filters for the most known real-time media and the most probable representations that they can take, from which the application can choose the appropriate ones. Conversely, if the application uses proprietary filtering it can download the filters into the proxy at the base station side prior to initiation of the transmission. The final solution, if both the client and the proxy are lacking support of some filters, is to have the proxy address a well-known, third party filter server for those filters, much like plug-in repositories are currently working for the major Web browsers.

According to the way that they manipulate media, filters are classified into several categories: hierarchical, frame-dropping, codec, splitting/mixing filters, etc [13]. Even though *MobiWeb* is capable of operating with all kinds of filters, we are currently using only frame-dropping filters in our simulation environment, since they nicely exemplify *MobiWeb's* utility and are easy to implement.

IV. SIMULATION RESULTS

To illustrate *MobiWeb's* utility, we performed several simulation experiments, using NS ver. 2, the Network Simulator tool [14]. In this paper we extend our previous research with CBR real-time adaptive applications [11], with the evaluation of VBR traffic, and specifically

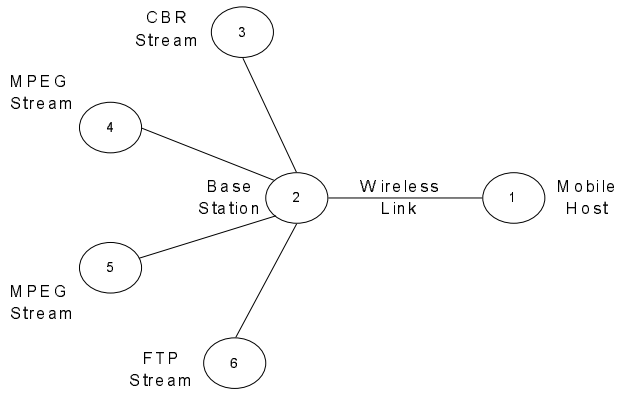

Figure 3: Simulation Topology

MPEG streams, in *MobiWeb's* context. A series of scenarios were executed on top of a simulated dynamic wireless environment, with variability in resources due to fluctuations in link quality and handoffs. Figure 3 shows the topology we used in the simulations. Over this testbed, we transmitted streams using either traditional Internet protocols or *MobiWeb*. NS is flexible enough to support the tracing of individual packets during the simulation. Such knowledge is utilized by our statistical analysis tools, which are capable of transforming it into a meaningful representation of the stream's perceptual playback quality.

For real-time adaptive traffic we used three MPEG streams with different representation characteristics (frame rate, GOP), as they are depicted in table 1. In addition, we simulated background traffic in the form of real-time CBR streams that use UDP/IP and best-effort FTP traffic transmitted over TCP/IP. When adaptation was enforced, *MobiWeb* used the frame-dropping filter to shape the transmission of the MPEG streams. The filter breaks the operation of the stream into 3 levels of quality. In the highest level all frames are transmitted. In the intermediate, B packets are discarded and in the lowest both B and P frames are discarded.

A. First simulation scenario

In the first scenario, we used the M1 MPEG stream along with a CBR real-time stream over the same wireless link. Doing so, we were able to evaluate the interference that UDP traffic introduces in M1's performance with and without the presence of *MobiWeb*. We executed the same handoff scenario, once using plain UDP with the two streams being non-adaptive, and once using *MobiWeb* with both streams adapting when handoff occurs and when resources are not enough to accommodate both of them without loss.

Figures 4a and 4b show the bandwidth allocation between the two streams over the wireless link with and without the existence of *MobiWeb*. With *MobiWeb*, we prioritized M1 higher than the CBR stream. Consequently, the initiation of the CBR stream did not severely interfere with M1. M1 retains enough resources to transmit with its highest LoQ, effectively preventing the interference from the CBR stream. In the case when plain UDP was used, the CBR stream allocates immediately all the resources it can use and M1 was forced to utilize the rest of them. This can be clearly shown at figures 5a and 5b, where the frames per second that reach the client side in time are plotted.

TABLE 2
TRACE RESULTS FROM THE SIMULATION TESTS

Test #	MPEG Stream	Good frames	Corrupted frames	Delayed frames	Dropped frames
1	M1+MobiWeb	933	74	58	15
	M1+UDP	934	251	66	642
2	M2+MobiWeb	546	34	31	17
	M2+UDP	478	3	247	158
3	M1+MobiWeb	967	39	31	53
	M3+MobiWeb	607	17	91	8
	M1+UDP	856	0	691	346
	M3+UDP	12	0	868	434

Although the figure shows that M1 looses only a small portion of frames overall, it is important that they are mainly I frames. Since I frames are the largest of all frames and have stringent timing requirements, the interference forces most of them to arrive late at the client. This means not only that they cannot be displayed, but also that they might be already late for the decoding of several proceeding frames that are inter-coded based on them.

Table 2 shows that in both cases the frames that arrived complete and in time are virtually the same. With UDP, M1 transmits more frames overall, since it cannot adapt to a lower frame rate. This however causes a significant amount of frames to be dropped at the BS. Nearly 1/3 of all frames is dropped before they can reach the client. Since a significant amount of the good frames are depending on the dropped frames for decoding, they will either not displayed or displayed with unrecognizable content. Finally, the large amount of corrupted frames with UDP implies that artifacts in the perceptual playback quality often appear and get carried over for several frames before the next correct I frame arrives.

B. Second simulation scenario

In the second scenario we replaced the CBR stream with an FTP one, in order to evaluate the interference that TCP traffic introduces to an MPEG stream. We also used M2 as our MPEG stream. Figures 6a and 6b show the results of this simulation. As it was expected, the presence of a TCP stream does not affect the transmission of M2 when resources are available. This is due to the congestion control that applies to the TCP stream, which forces it to surrender the resources in favor of the UDP stream. When handoff to a lower quality wireless medium occurs however, the interference from the TCP stream becomes noticeable.

As table 2 shows, comparing the results with the previous scenario with the CBR stream, there is a shift in the statistics from frames that are corrupted due to loss of some packets, to frames that are delayed due to the concurrent transmission of the TCP stream. The result however is even worse, since corrupted frames can be displayed even with some artifacts in them, while late frames can only aid in decoding of succeeding ones. Therefore, the playback quality of M2 is throttled down to near zero between times 14 and 28, when M2 is unable to adapt in the absence of *MobiWeb*. On the other hand, the presence of *MobiWeb* allows M2 to adapt when handoff occurs and satisfies its timing constraints by allowing at least the I frames to reach their destination in time.

C. Third simulation scenario

In the third simulation scenario, we measure the interference that two MPEG streams introduce to each other, when they concurrently transmit over the same wireless link. We used M1 and M3 for this purpose.

Figures 7a and 7b show the displayed quality of the two MPEG streams when *MobiWeb* is applied, while 8a and 8b refer in the case where UDP is used. With *MobiWeb*, M1 is initiated with a lower priority and it is the first to adapt when resources are not enough for both streams to transmit with maximum quality. Handoffs to lower quality links force both streams to gracefully surrender LoQs, retaining though the best possible performance achievable in the current environment. Finally, when handoff to a better link occurs, both streams explore the available resources quickly and efficiently.

When plain UDP is used in the same scenario, the results are completely different. As figures 8a and 8b show, the interference between the two streams is so prevailing, that none of them is able to deliver frames in time. When both streams are transmitting they are effectively splitting the available bandwidth over the wireless link. However, in the case of VBR real-time traffic, bandwidth isn't the most important resource, delay is. Hence, the delay that the packets of one stream introduce to the packets of the other causes virtually all frames to arrive late at the client. As table 2 depicts, 1 out of 3 frames transmitted by either stream is dropped while the other 2 reach late their destination.

The simulation tests showed that *MobiWeb* allows adaptive real-time streams to achieve decent quality when the wireless link is severely degraded. Instead of dropping packets indiscriminately, *MobiWeb's* filtering mechanism favors the transmission of the most important frames, achieving an acceptable quality for the MPEG streams. Furthermore, inter-stream interference due to background traffic is avoided during time of contention, without being unfair to traditional traffic when resources are available. Overall, the simulation tests proved the importance and the utility of a management scheme, like *MobiWeb*, for the transmission of real-time multimedia streams over limited wireless links.

V. Conclusions

We discussed the implications that frequent handoffs and wireless link variations introduce to the seamless execution of Internet applications, especially real-time ones. We analyzed the parameters of the problem that motivated the design of *MobiWeb*, a proxy-based architecture that substantially increases the utility of continuous media over bandwidth limited and error-prone wireless links. Using admission control and a dynamic prioritization scheme it shields the performance of important applications from the remainder of the traffic. With the LoQs and the specialized timers, it allows for both fast exploration of the available resources and efficient adaptation during handoffs, while protecting the real-time streams from frequently adapting to short-term fluctuations.

We showed, through simulation, *MobiWeb's* importance and utility to adaptive real-time streams over wireless links comparing to traditional Internet protocols. Their performance is substantially enhanced, particularly when frequent mobility and rapid fluctuations change the wireless environment.

We contribute three new features: The first is dynamic priorities designed to lower a stream's priority when it advances a LoQ, in order to give a chance to higher priority streams to both adapt **last** during degradation and explore available resources **first**. The second is the prevention of inter-stream interference by allowing only one stream at a time to adapt when the link quality changes. Finally, the third contribution is the introduction of the specialized timers in order to shield the streams from short-term variations, while allowing fast explorations and a steady state operation during link stability.

References

[1] H. Schulzrinne, S. Casner, R. Frederick and V. Jacobson, "RTP: a transport protocol for real-time applications," RFC 1889, Internet Engineering Task Force, January 1996.

[2] B.C. Housel and D.B. Lindquist, "WebExpress: A system for optimizing web browsing in a wireless environment," in *MOBICOM'96 Demonstration Session*, November 1996.

[3] A. Fox, S.D. Gribble, E. A. Brewer and E. Amir, "Adapting to network and client variability via on-demand dynamic distillation," in *Proceedings of the Seventh International ACM Conference on ASPLOS*, Cambridge MA, October 1996.

[4] J. Inouye, S. Cen, C. Pu and J. Walpole, "System support for mobile multimedia applications," in *Proceedings of the 7th International Workshop on NOSSDAV*, St Louis MO, May 1997.

[5] V. Bharghavan and V. Gupta, "A framework for application adaptation in mobile computing environments," in *Proceedings of IEEE Compsoc'97*, November 1997.

[6] M. Satyanarayanan, B. Noble, D. Narayanan, J.E. Tilton, J. Flinn and K.R. Walker, "Agile application-aware adaptation for mobility," in *Proceedings of the 16th ACM Symposium On Operating Systems Principles*, vol. 31, no. 5, December 1997, pp. 276-287.

[7] V. Bharghavan, L. Kang-Won, L. Songwu, H. Sungwon, L. Jin-Ru and D. Dwyer, "The TIMELY adaptive resource management architecture," in *IEEE Personal Communications*, vol. 5, no. 4, August 1998, pp. 20-31.

[8] O. Angin, A.T. Campbell, M.E. Kounavis and R.R.-F. Liao, "The mobiware toolkit: programmable support for adaptive mobile networking," in *IEEE Personal Communications*, vol. 5, no. 4, August 1998, pp. 32-43.

[9] The WAP Forum, Wireless application protocol http://www.wapforum.org

[10] M. Margaritidis and G.C. Polyzos, "Application-assisted adaptation of real-time streams over wireless links," in *Proceedings of the Second Annual UCSD Conference on Wireless Communications*, San Diego, California, March 1999, pp.

[11] M. Margaritidis and G.C. Polyzos, "Wireless network support for adaptive real-time applications," in *Proceedings of the 1999 Advanced Simulation Technologies Conference (ASTC99)*, San Diego, California, April 1999, pp.

[12] J. Pasquale, G. Polyzos, E. Anderson and V. Kompella, "Filter propagation in dissemination trees: Trading off bandwidth and processing in continuous media networks," in *Proceedings of the 4th NOSSDAV*, Lancaster, UK, 1993, pp. 269-278.

[13] Nicholas Yeadon, Francisco Garcia, David Hutchison and Doug Shepherd, "Filters: QoS support mechanisms for mulipeer communications," in *IEEE Journal on Selected Areas in Communications special issue on Distributed Multimedia Systems and Technology*, vol. 14, no. 7, September 1996, pp. 1245-1262.

[14] The VINT Project, Network Simulator version 2, http://www-mash.CS.Berkeley.EDU/ns/ns.html

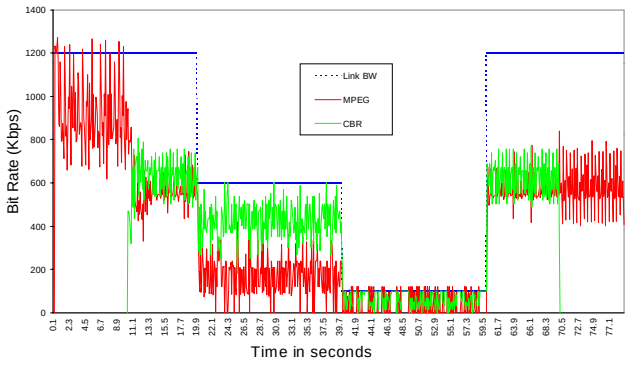

Figure 4a: BW used on first simulation with MobiWeb

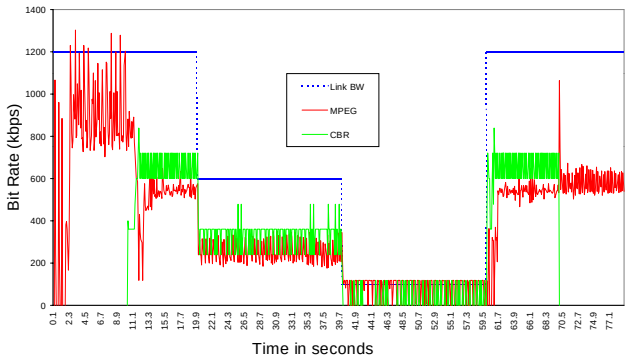

Figure 4b: BW used on first simulation w/o MobiWeb

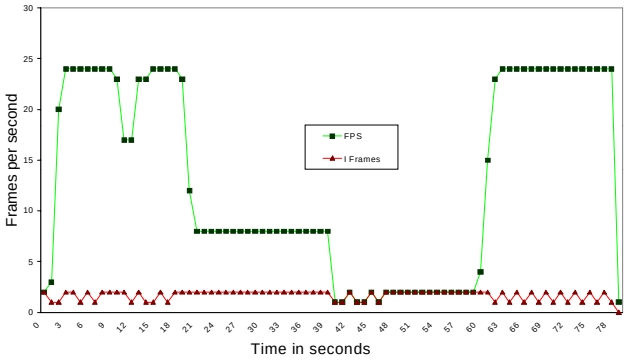

Figure 5a: FPS of first simulation with MobiWeb

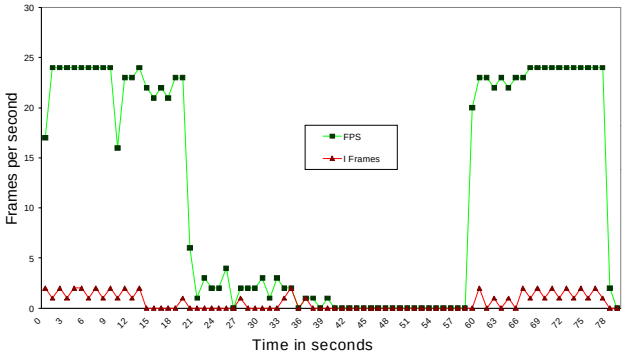

Figure 5b: FPS of first simulation w/o MobiWeb

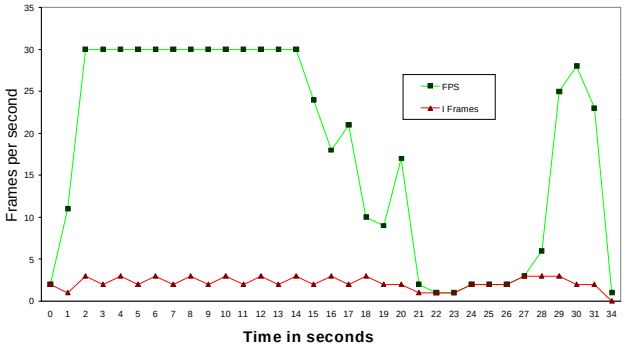

Figure 6a: FPS of second simulation with MobiWeb

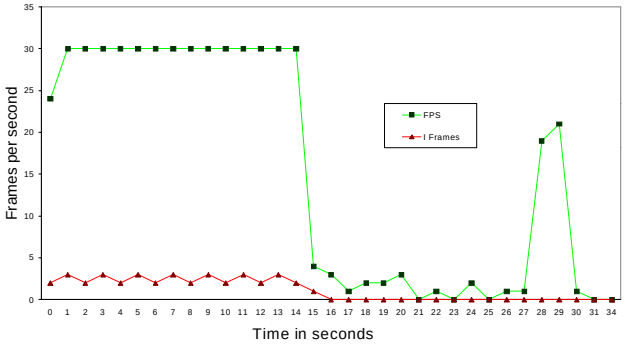

Figure 6b: FPS of second simulation w/o MobiWeb

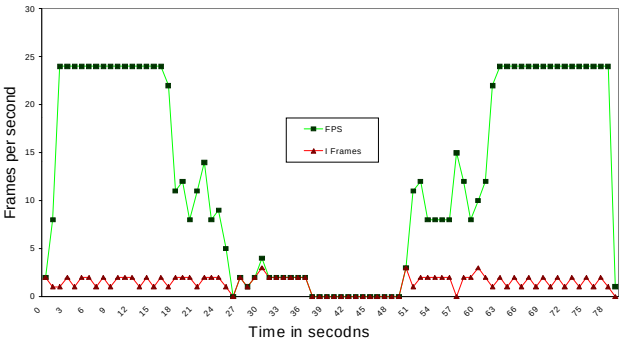

Figure 7a: FPS of M1 in third simulation with MobiWeb

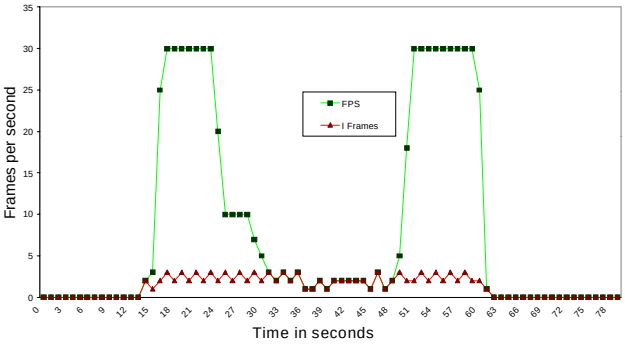

Figure 7b: FPS of M3 in third simulation with MobiWeb

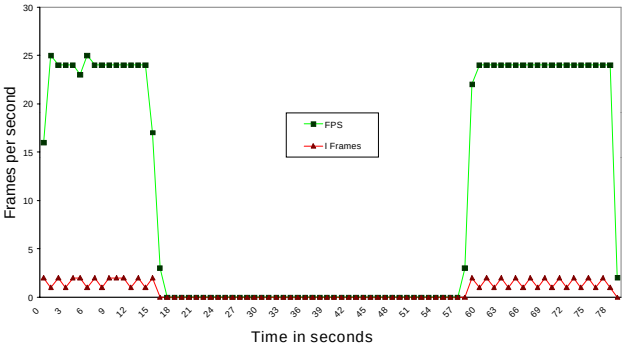

Figure 8a: FPS of M1 in third simulation w/o MobiWeb

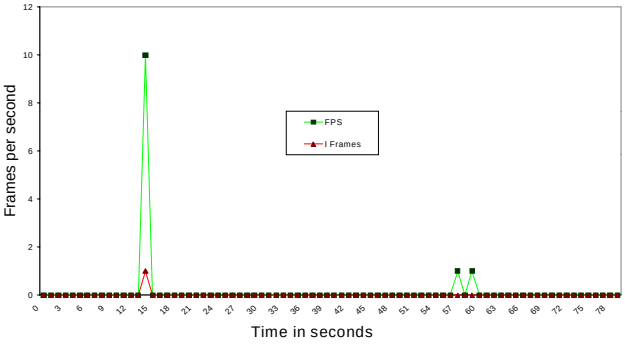

Figure 8b: FPS of M3 in third simulation w/o MobiWeb