Control of Multimedia Wireless Communication Network

MARIUS-CRISTIAN NICULESCU*) and IONUT RESCEANU*)
Department of Automation and Mechatronics*)
University of Craiova
Al. I. Cuza Street, No. 13, Craiova, RO-200585
ROMANIA
toros@rdscv.ro

Abstract: - This paper presents a new management technique aimed at increasing the energy efficiency of client-server multimedia applications running on wireless portable devices. Multimedia communication can be defied as the field pertaining to the formation, storage, retrieval, dissemination and usages of multiple "media" such as images, texts, graphics, animation, video and audio. The recent explosive growths of the Internet and multimedia research have raised the interest in distributed multimedia systems over Internet. In this paper, we propose an architecture for capturing, distributing and displaying multimedia content through existing Internet and Intranet infrastructures. An object-oriented architecture for the building of customized multimedia clients and servers facilitates software maintenances and updates as application requirements evolve and code reuse. The architecture is consisted of client/ server model.

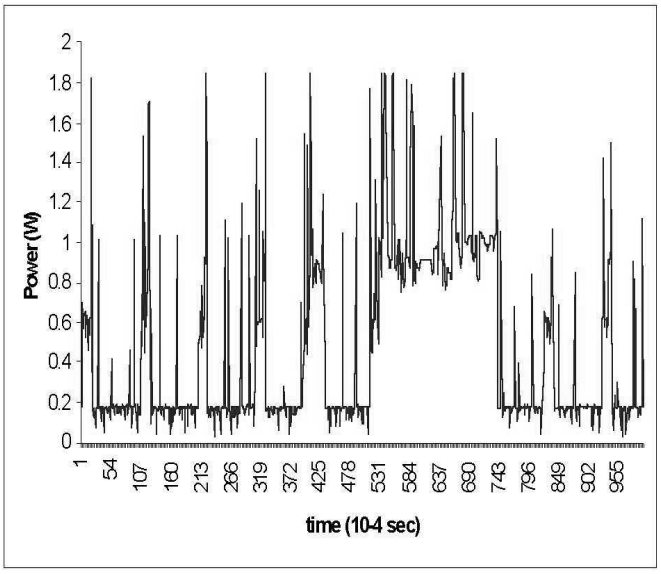
Key-Words: - WLAN, MPEG4, 802.11b PM, Server PM, Internet.

1 Introduction

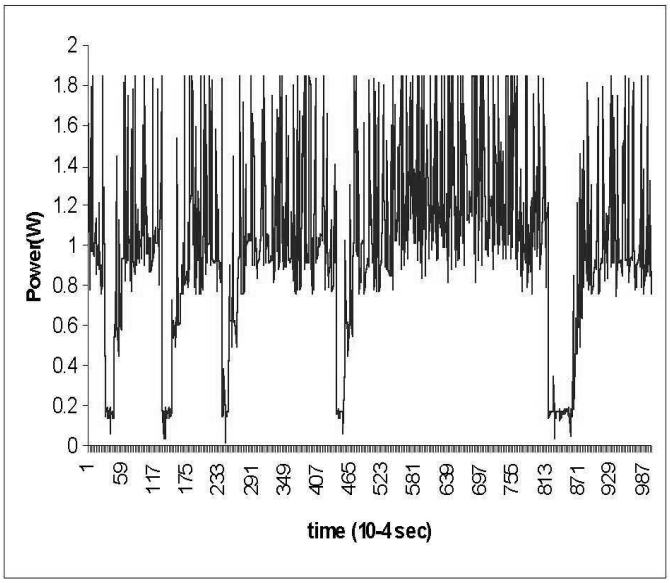
The popularity of multimedia streaming on the Internet, combined with the growing deployment of wireless access networks, augurs the converging usage of these two technologies in the not-too-distant future. Experience with wireless multimedia streaming on today's networks can provide valuable insights into the design of future wireless multimedia networks and applications. In this paper, we present a measurement study of RealMedia streaming traffic on an indoor IEEE 802.11b wireless LAN. The traffic is analyzed hierarchically, from the application layer to the network layer to the data link layer. We focus on the traffic structure at each layer, and on interaction effects across layers. Our main observation is that streaming quality is quite robust in all but the poorest channel conditions, despite the inherent burstiness of both the RealMedia application workload and wireless channel errors. Several factors contribute to these good results. First, although RealVideo is typically variable-bit-rate (VBR) at the application layer, it is often streamed as constant-bit-rate (CBR) at the network layer, reducing burstiness and thus the chances of packet losses due to buffer overflow in the network path. Second, while the wireless channel has bursty error characteristics, MAC-layer retransmission in 802.11b hides most errors from higher-layer protocols. Finally, the application layer's NACK-based error control is effective in recovering missing packets when needed. Our results demonstrate the viability of multimedia streaming on future wireless LANs.

Portable devices spend a considerable amount of energy in order to support power hungry peripherals e.g. wireless local area network (WLAN) interfaces, and liquid crystal displays (LCD). A good example of such a device is the Smart-Badge IV [21] wearable computer. It consists of a StrongARM-1110 processoi and SA-1111 coprocessor, memory, WLAN interface, audio codec and 2.2" LCD. Depending on the network traffic, the WLAN accounts for as much as 63% oi the overall system power consumption. One way to reduce the energy consumption is to use the power management included in the 802.11b standard (802.11b PM) [4]. In the standard, an access point (AP) transmits a beacon every 100 rns, followed by a traffic indication map (TIM). Each client checks the TIM for its turn to send or receive data. When not communicating, the WLAN goes into the doze mode until the next beacon. flnfinrtiinatftlv. the iirotor.ol nower management is not verv effective. First, the energy efficiency of the 802.11b PM decreases and receiver wait times increase with more mobile hosts, since multiple concurrent attempts at synchronization with the beacon cause media access contention. Second, the response time oi the wireless link with 802.11b PM grows because of the delay imposed by sleep periods [22]. These two issues can be resolved by careful scheduling of communication between the server and the client WLAN. Lastly, in a typical wireless network, broadcast traffic can significantly reduce the chances to enter the doze mode.
The Internet and Multimedia are technologies, which have been around us for a several years, and both are expanding in terms of use and technological

advancement. The two technologies are quickly converging because the infrastructure often is already in place and adopting existing datagram networks and Internet connections will save expensive software development. On the other hand, this approach seems more economical and easier to manage then separate datagram and real-time networks. Figure 1 shows the power consumption of a WLAN card with 802.11b PM enabled under light and heavy network broadcast traffic conditions. Clearly, as the amount of broadcast traffic increases, the WLAN spends a large amount of energy listening to it, even if no other application is running on the device. As a result, very little or no energy savings are obtained. One way to solve this problem is to turn off the card. It is important to schedule data transmission carefully, since the overhead of waking up the WLAN from the off state is large. Many current wireless local area networks arc organized in a client-server fashion. Multiple WLAN clients connect to wired servers via APs. Servers are great candidates for efficient scheduling of data transmission to clients as they are not power constrained, and know both wired and wireless network conditions. In this work, we present a server controlled power management strategy. Our technique exploits server knowledge of the workload, traffic conditions and feedback information from the client in order to minimize WLAN power consumption. Our methodology is applicable to a wide variety of applications, ranging from video and audio streaming, to web browsing and e-mail. We define two new entities: a server power manager (server PM) and a client power manager (client PM). Server PM uses the information obtained from the client and the network to control the parameters of 802.11b PM and to perform energy efficient traffic reshaping so that WLAN can be turned off.



a) light traffic



b) heavy traffic

Fig. 1. 802.11b PM under different broadcast traffic conditions

Client PM communicates through a, dedicated low-bandwidth link with the server PM and implements power controls by interfacing with device drivers. It also provides a set of APIs that client programs can use to provide extra information to the server. In order to illustrate the effectiveness of our approach, we tested our methodology with a streaming video application. By using our approach with this application, we can exploit the server knowledge of stream characteristics. We show that when our methodology is implemented on both the server end, and on the client end, we measure savings of more than 67% in power with respect to leaving the card always on, and more than 50% relative to using default 802.11b PM, Even larger savings are possible for applications that inherently have longer idle periods, such as e-mail or web browsing. Our methodology can also be easily extended to manage other system components (e.g. CPU).

The rest of the paper is organized as follows. Section 2 presents some related work. Section 3 gives an overview of the proposed methodology and describes server and client power managers. Section 4 presents experimental results and Section 5 concludes the paper.

2 Related Work

Recent advances in compression technology for video and audio has also prompted the introduction of multimedia integrated to Internet. But the problem is that voice and video are time sensitive and they have to be sent in a stream. Also the current packet technologies

like TCP/IP can't guarantee that the traffic will get where it needs to go without being jumbled and jerky. To cope with this problem, the Integrated Services working group in the IETF (Internet Engineering Task Force) developed an enhanced Internet service model that includes best-effort service and real-time service. The Resource Reservation protocol (RSVP) together with Realtime Transport protocol (RTP), Realtime Control protocol, Realtime Streamming protocol provides a working foundation for IP networks to carry voice and video in addition to the data traffic.

The wireless network power optimization problem has been addressed at different abstraction layers, starting from physical, to system and application level. Energy efficient channel coding and traffic shaping to exploit battery lifetime oi portable devices were proposed in [3]. A physical layer aware scheduling algorithm aimed at efficient management of sleep modes in sensor network nodes is illustrated in [17]. Energy efficiency can be improved at the data link layer by performing adaptive packet length and error control [8]. At the protocol level, there have been attempts to improve the efficiency of the standard 802.lib, and proposals for new protocols [5,6,19]. Packet scheduling strategies can also be used to reduce the energy consumption of transmit power. In [14] authors propose the E^2WFQ scheduling policies based on Dynamic Modulation Scaling. A small price in packet latency is traded for the reduced energy consumption. Traditional system-level power management techniques are divided into those aimed at shutting down components and policies that dynamically scale down processing voltage and frequency [20,1]. Energy-performance trade-offs based on application needs have been recently addressed [7]. Several authors exploit the energy-QoS trade-off [12, 19,11]. A different approach is to perform transcoding and traffic smoothing at the server side by exploiting estimation of energy budget at the clients [16]. A new communication system, consisting of a server, clients and proxies, that reduces the energy consumption of 802.lib compliant portable devices by exploiting a secondary low-power channel is presented in [18]. Since multimedia applications are often most demanding of system resources, a few researchers studied the cooperation between such applications and the OS to save energy [9,2,15,10]. We present a new methodology, where server knowledge of the workload is exploited to control the power configuration of the radio interface. Moreover, with respect to previous application-driven policies, our infrastructure can be used with a wide range of applications, since it exploits very common parameters.

3 Server Controlled Power Management

The management of the server functions is effected by an ApplicationController, a ServerInterface, a DataHandler, a Session-Handler, and a ConnectionHandler objects. The ApplicationController serves as the controller of the server functions. All major events, such as the event of reception of data from user is reported to the ApplicationController. The ApplicationController then invokes the corresponding methods of -the DataHandler, the SessionHandler or the ServerInterface to inform them of the event.

The Server-Interface handles the interaction with the user. Therefore, it provides GUI objects for configuring and monitoring architecture components and triggers the notification of major application events to the user. The DataHandler manages the internal and intra-applications dataflow. It uses DataForwarder object to handle data forwarding. SessionHandler controls the whole session of a dialogue between clients. It receives information about sessions initiated by remote clients, stores the session information and exports information about status of the sessions to the clients. The SessionHandler uses session description and announcement protocols for these purposes. It stores the session description in a session directory.

Our methodology exploits server knowledge of the workload, traffic conditions and feedback information from the client in order to minimize power consumption. The server schedules communication sessions with the client based on the knowledge of both, the wireless and wired networks, e.g favorable channel conditions or channel bandwidth capabilities. When broadcast traffic needs to be monitored, the server can enable the 802.11 PM. Alternatively, it can coordinate the shut down of WLAN and perform on-time wake-up, thus avoiding the performance penalty typically incurred by client-centric approaches. Our application driven infrastructure can be also be used to manage power consumption in stand-alone and ad-hoc applications. The power control strategy can easily be extended to include other system components, such as peripherals or the CPU.

In order to exploit the extra information available at the server and the client, the traditional client-centric power manager model has to be extended. Two different power managers are defined: one running on the client (Client PM) and the other on the server (Server PM). The two PMs exchange power control information through a dedicated TCP connection. As shown in Figure 2, the client PM interfaces directly with the drivers on the portable device. It also collects client application dependent information.

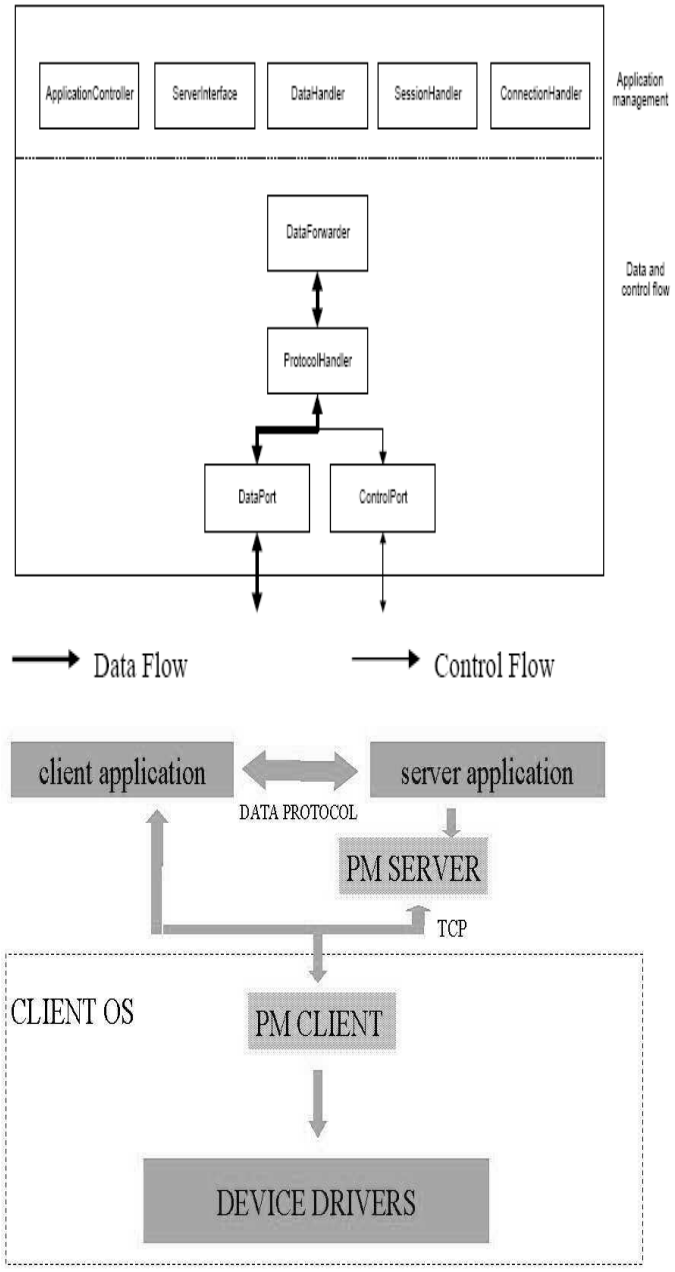


Fig. 2. Server Controlled PM Architecture

The server PM interfaces with the server application and the client PM. Figure 3 shows the communication protocol between the server and client. Control commands are issued by the server PM, interpreted by the client PM, and translated into appropriate device driver calls. Upon request from the server PM, device specific information is fetched by the client PM. Application specific information can also be retrieved bv the client PM via API calls.

3.1 Server Power Manager

The DataPort, ProtocolHandler and Control-Port are the objects involved in dataflow in server. The receive() method of DataPort object is responsible for getting data from network. Data is then goes through ProtocolHandler and Dataforwarder objecs. Dataforwarder objects forward() method forward the data to other destination. ProtocolHandler only bypass it to Data-Forwarder object. But for session data, ProtocolHandler's protoencode() method performs encoding the session data. Data-Forwarder after getting the data, decides its destination and sends again to the network through ProtocolHandler and DataPort object.

The server PM interfaces with the application in order to exploit information available on the host system, in addition to the knowledge of the overall network conditions and the specific feedback information provided by each client. Based on all this knowledge, the server PM controls the power configuration of the WLAN card by communicating with the client PM. The power control command used include:

i) switch-off the WLAN;
ii) set the off time;
iii) enable the 802.11b PM policy;
iv) set 802.11b PM parameters, e.g. the period between wake ups and the timeout before going back to doze mode. The server PM enables:

Client Adaptation.

When the user wants to talk with a remote user first it connects to server to a predefined port. Also it specifies the address of the remote user. The server maintains a directory of online users who have registered to it. The user can choose a partner from the directory of server also. After requesting to the server the user waits for response to receive from server. The server informs the presence of remote users availability for conversation. If available then the user opens a new data connection to server and continues conversation. The server manages the forwarding of data. As the core Internet protocols like TCP/IP and UDP/IP do not provide any realtime functionality, therefore they are not able to fulfill the requirements needed for transmitting multimedia data. TCP is a connection-oriented protocol that is reliable including flow control and supports bytestream in full duplex mode. But it is not a good basis to transport real-time multimedia data over a network. Multimedia demands a constant bit rate even if sometimes a packet gets lost and that is exactly what TCP cannot guarantee. In multimedia applications multicast methods are often used to serve a couple of clients at the same time. So the connection oriented ability of TCP is not very helpful, allows it only a point-to-point connection on application level between a server and one client. The multicast method to send data to more than one client is not supported by the TCP protocol.

On the other hand, UDP is a connectionless protocol

that tries best effort delivery with no flow-control and included message support. But UDP is not reliable and that is one of the key advantages for real-time transmissions. It is a best effort delivery protocol. That is exactly what is needed for multimedia data. UDP also supports multicast methods. In the manner of broadcasting a video to more than one client this is exactly what is needed. Multicast simply multiplies the data on a network segment to a group of clients. It would be necessary to set up sessions with each client and to multiply the data on application level. So it would generate a multiple of streams on a single network segment. But there are not only good things on UDP. This protocol does not support flow-control and that is actually very important for real-time applications. So UDP is just a good base that goes in the right direction but it lacks a couple of key functions. The Integrated Services working group in the IETF (Internet Engineering Task Force) developed an enhanced Internet service model called Integrated Services that
includes best-effort service and real-time service. The real-time service will enable IP networks to provide quality of service to multimedia applications. Resource Reservation Protocol (RSVP), together with Real-time Transport Protocol (RTP), Real-Time Control Protocol (RTCP), Real-Time Streaming Protocol (RTSP), provides a working foundation for real-time services. Integrated Services allows applications to configure and manage a single infrastructure for multimedia applications and traditional applications. It is a comprehensive approach to provide applications with the type of service they need and in the quality they choose. The Real-time Transport Protocol (RTP) provides support for the transport of real-time data such as video and audio streams. The services provided by RTP include time reconstruction, loss detection, security and content identification. RTP is designed to work in conjunction with the auxiliary control protocol RTCP to get feedback on quality of data transmission and information about participants in the on-going session.

After the application server initializes the server PM. an additional set of APIs becomes available. The initialization routine places the server PM into a state of waiting for incoming client PM requests. Each accessing client PM has a dedicated TCP connection to the server PM. The client PM informs the server PM about the application e.g. input buffer size, the expected value and variance of the service rate, as well as network interface specific information e.g. the WLAN card status and its on/off transition time.

Traffic Adaptation. Server PM monitors both the wired and wireless traffic conditions with minimum overhead. By accounting for the broadcast packet rate, the server decides when to enable the 802.11b PM. For example, in very light traffic conditions, the 802.11b PM might be used instead of a switch-off policy.

Traffic Shaping. The server PM schedules transmissions to the client in bursts, in order to compensate for the client performance and energy overheads during transitions between on and off states. The client WLAN card is switched off once the server has sent a burst of data designed to keep the application busy till the next communication burst. The burst size and delay between bursts are precomputcd at the server. The delay should be large enough to almost empty the client input buffer, while the burst size should avoid overflow while keeping the buffer sufficiently filled. This maximizes the off time of the card and reduces the number of transitions between on and off states. The time for the buffer to emtv. D_{burst} is the ratio of the total number of packets in the burst. $SIZE_{burst}$, to the average service rate (or buffer depletionrate) at the client, λ_S ,mean:

$$D_{burst} = \frac{SIZE_{burst}}{\lambda_S mean} \quad (1)$$

The total delay between bursts, $D_{burst}^{|}$ is:

$$D_{burst}^{|} = D_{burst} - T_{tran} - T_{cushion} \quad (2)$$

Where T_{tran} is the time needed for the transitions between WLAN on and off states, $T_{cushion}$ is the "cushion" time so that the buffer does not ever empty completely. The cushion time helps accommodate variations of the service rate and arrival rate, 10% was found to be sufficient for most test cases. The total energy saved if the client WLAN is turned off is:

$$E_{tot} = P_{on} * T_{on} + P_{off} * T_{off} + P_{tran} * T_{tran} \quad (3)$$

Here P_{tran} is the transition power (hardware dependent). T_{tran} can be computed by multiplying the duration of one transition to the active state, $T_{wake\text{-}up}$, by the total number of transitions, N_{tran}. The number of transitions depends on the size of the burst and the total size of streamed file.

If $\lambda_{a,mean}$ is the average arrival rate at the client input buffer, the total time to send a burst of data to the client will be:

$$T_{on} = \frac{SIZE_{burst}}{\lambda_a, mean} \quad (4)$$

Payload Type	Sequence Number
Time Stamp	
SenderSourRCe (SSRC) Identifier	

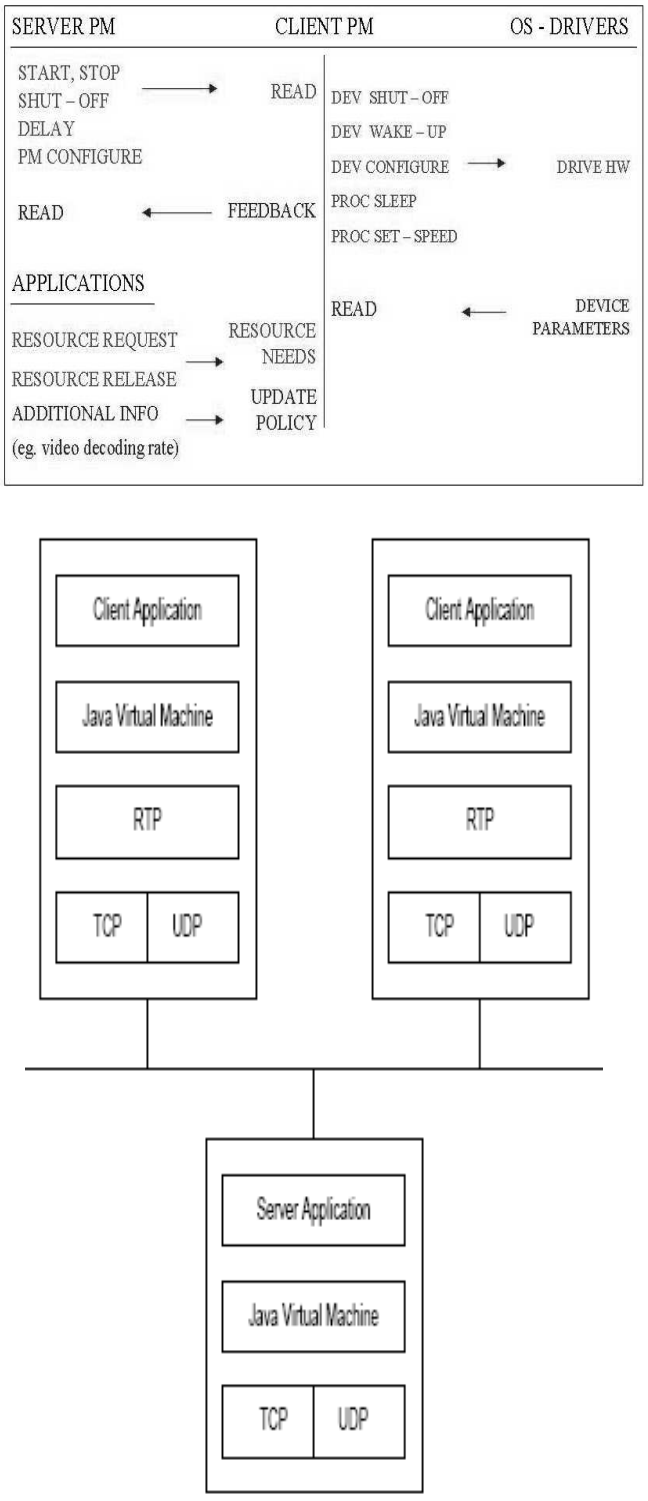


Fig. 3. Communication protocol

Finally, we can obtain the total energy consumed with our methodology:

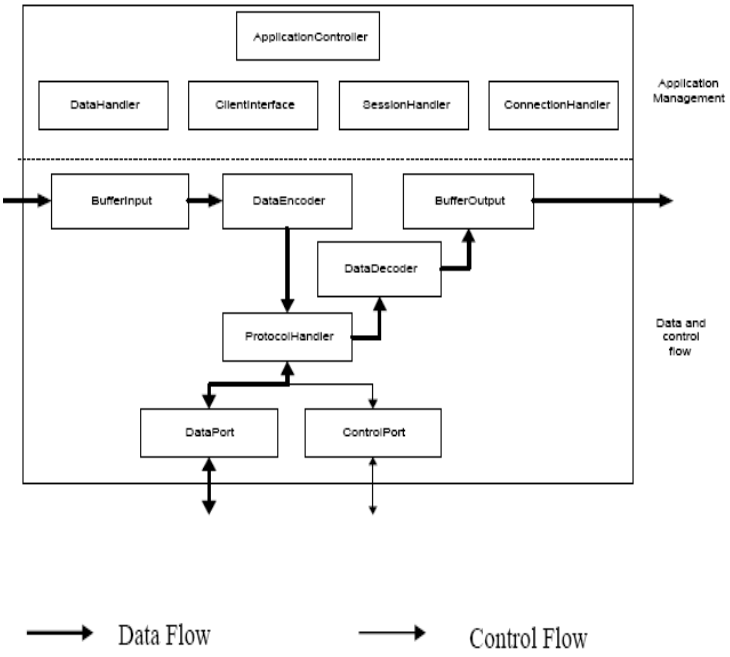
$$E_{tot} = P_{on} * \frac{SIZE_{burst}}{\lambda_{a,} mean} + P_{off} * \frac{SIZE_{transfer}}{\lambda_{S,} mean} + P_{tran} * T_{tran} \quad (5)$$

In addition to scheduling communication so that the client WLAN can be turned off with no performance overhead, we can also perform the same scheduling while 802.11b PM is enabled. This allows us to save energy during the burst periods, but the overall burst time grows because of the decreased responsiveness and increased contention probability between data and broadcast packets.

3.2 Client Power Manager

The client power manager communicates with the server PM, and also interfaces with the device drivers and client applications. The main client PM tasks are:

Server Interface The client application decides when to set-up the communication between the server PM and the client PM. Once established, the client application provides the information to be forwarded to the server e.g. the buffer size and the deoletion rate. The device drivers reoort the main characteristics of the devices to be managed, e.g. the transition time between on and off states for client WLAN.



Device Interface The client PM calls the appropriate device driver function depending on the command sent by the server PM. Possible actions taken by the client include changing the parameter of the 802.11b PM, switching the WLAN on and off, and reading the interface statistics such as the signal to noise ratio. The client PM can also interact with the CPU by changing its power mode oi setting its clock speed.

Application Interface Applications can feedback information that can be exploited by both the server and the client PM. Examples include sending the current backlog level to the

server, so that thee server knows exactly how much data to provide in a burst in order to refill the buffer; or providing the buffei size. The application could directly request a WLAN wake-up when its input buffer reaches a minimum value.

Application Driven Infrastructure The client PM can also be used in stand-alone mode (with no server). While in this mode, the applications provide their resource needs to the power manager. The PM then turns on or off devices appropriately. Some of the overhead needed to turn on a device can be masked, as many applications have extra latency due to the initial set-up.

4 Experimental Results

The streaming media server used for this work is a research prototype developed at Hewlett-Packard Laboratories. Real Time Streaming Protocol (RTSP) is used for session initiation and termination between the client and server. The media data units are carried using Real-time Transport Protocol (RTP) over User Datagram Protocol (UDP). Transmitting media data across the net in real-time requires high network throughput. It's easier to compensate for lost data than to compensate for large delays in receiving the data. This is very different from accessing static data such as a file, where the most important thing is that all of the data arrive at its destination. Consequently, the protocols used for static data don't work well for streaming media. The HTTP and FTP protocols are based on the Transmission Control Protocol (TCP). TCP is a transport-layer protocol[1] designed for reliable data communications on low-bandwidth, high-error-rate networks. When a packet is lost or corrupted, it's retransmitted. The overhead of guaranteeing reliable data transfer slows the overall transmission rate. For this reason, underlying protocols other than TCP are typically used for streaming media. One that's commonly used is the User Datagram Protocol (UDP). UDP provides no guarantees to the upper layer protocol for message delivery and a UDP sender retains no state on UDP messages once sent (for this reason UDP is sometimes called the Unreliable Datagram Protocol). UDP adds only application multiplexing and checksumming of the header and payload. If any kind of reliability for the information transmitted is needed, it must be implemented in upper layers.

Lacking reliability, UDP applications must generally be willing to accept some loss, errors or duplication. Some applications such as TFTP may add rudimentary reliability mechanisms into the application layer as needed. Most often, UDP applications do not require reliability mechanisms and may even be hindered by them. Streaming media, real-time multiplayer games and voice over IP (VoIP) are examples of applications that often use UDP. If an application requires a high degree of reliability, a protocol such as the Transmission Control Protocol or erasure codes may be used instead. Lacking any congestion avoidance and control mechanisms, network-based mechanisms are required to minimize potential congestion collapse effects of uncontrolled, high rate UDP traffic loads. In other words, since UDP senders cannot detect congestion, network-based elements such as routers using packet queuing and dropping techniques will often be the only tool available to slow down excessive UDP traffic. The Datagram Congestion Control Protocol (DCCP) is being designed as a partial solution to this potential problem by adding end host TCP-friendly congestion control behavior to high-rate UDP streams such as streaming media. While the total amount of UDP traffic found on a typical network is often in the order of only a few percent, numerous key applications use UDP, including: the Domain Name System (DNS) (since most DNS queries only consist of a single request followed by a single reply), the simple network management protocol (SNMP), the Dynamic Host Configuration Protocol (DHCP) and the Routing Information Protocol (RIP).

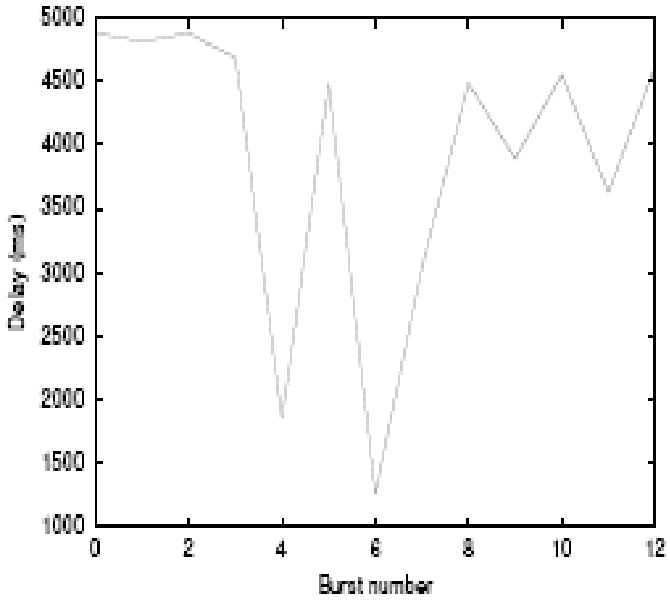
UDP is an unreliable protocol; it does not guarantee that each packet will reach its destination. There's also no guarantee that the packets will arrive in the order that they were sent. The receiver has to be able to compensate for lost data, duplicate packets, and packets that arrive out of order. Like TCP, UDP is a general transport-layer protocol--a lower-level networking protocol on top of which more application-specific protocols are built. The Internet standard for transporting real-time data such as audio and video is the Real-Time Transport Protocol (RTP). RTP is defined in IETF RFC 1889, a product of the AVT working group of the Internet Engineering Task Force (IETF).

Timestamps of the individual video frames are used to determine the deadline for sending data packets sent from the server to the client. The server can exploit client buffering capabilities to reschedule packet transmission times based on a cost function. In our experiments, the server transmits MPEG4 video data to a portable client device via WLAN. We use two different benchmarks in our tests. Benchmark 1 has 12 bursts and runs at 15 frames/sec, while Benchmark 2 has 402 bursts at 30 frames/sec. The first benchmark has a large number of single packet frames, while the second benchmark has

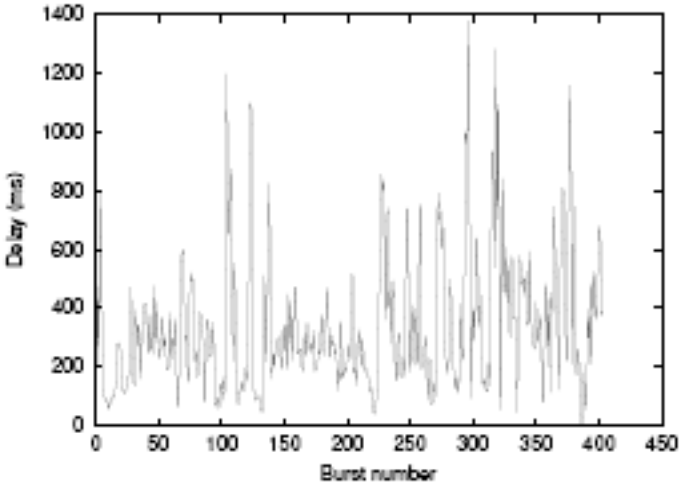
many large multi-packet frames. The delay between bursts is:

$$D_{burst} = frame\_time * (\frac{n_1}{1} + \frac{n_2}{2} + \frac{n_3}{3} + ... + \frac{n_M}{M}) \quad (6)$$

Here *frame_time* is the interval between frames, HJ is the number of frames consisting of *i* packets and *M* is the maximum number of packet per frame. Figure 4 shows that the Benchmark 1 has a delay of around 4 seconds, while the Benchmark 2 has a shorter delay because the frames are more complex. Based on the delay computation, the server transmits a burst of data to the client followed by the command to switch off WLAN and the length of time until the next transmission. The client rcsnonds bv turnine off the WLAN and turning it



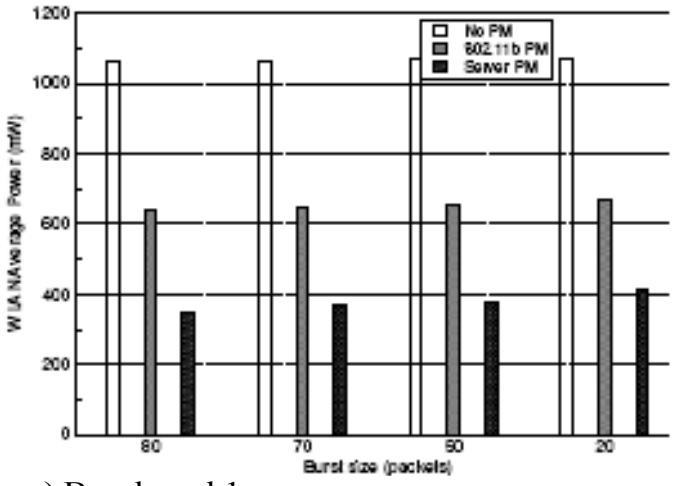
a) Benchmark 1



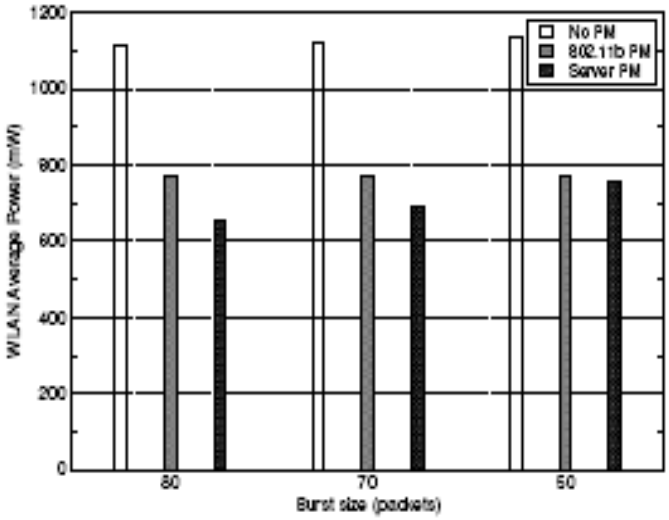
b) Benchmark 2

Fig. 4. Computed burst delays for each benchmark

back on at the pre-scheduled time. As a result, no performance penalty due tc longer turn on time is incurred. The approach was tested by measuring power consumption of a Smart-Badge IV wearable device, equipped with a CISCO Aironet 350 PCMCIA WLAN card. The off/on transition time was measured at 300 ms, with an average transition power consumption of 0.1 W. The server is connected directly to an AP. and the network is completely isolated in order to perform repeatable experiments. Broadcast traffic is introduced in a controlled manner by using real traces collected from other open networks. Power measurements are performed using a DAQ board accumulating and averaging current and voltage samples (10 ksamples/sec). All measurements include the overhead imposed by our power management protocol.



a) Benchmark1



c) Benchmark2

Fig. 5. Power consumption for each benchmark

First we computed the average power consumed by the WLAN card when receiving the video stream with different burst sizes for medium broadcast traffic conditions. The experiment is performed for multiple situations, WLAN with no PM. onlv 802.11b PM. and with server PM. From Fismre 4. for Benchmark 1. the server controlled approach saves 67% of average power compared to no PM, ario 50% compared to the default 802.11b PM. The average power savings increase as the burst size increases, since this enables longer times between bursts, thus better compensation for the transition delay between the WLAN on and off states. In all three cases, the video plays back in real time, thus the reported average power savings directly correspond to energy savings. For Benchmark 2, the delays are very short, however, we still save 41% of power with respect to leaving the card always on and 15% with respect to the standard 802.11b power management protocol with bursts of 80 packets. We cannot use a burst size smaller than 50 packets, since the computed delays arc too short and the card can never be switched off. Figure 6 presents the average power consumed during streaming video under different broadcast traffic conditions. In the plot we show the difference between using only the default 802.11b PM and server controlled switch off policy for the WLAN card (with no 802.11b PM). The switch off policy is almost insensitive to the traffic conditions, while the 802.11b PM performs better only in light traffic conditions. Clearly, our server controlled power management approach is more efficient than either policy alone as it always selects the best of the two policies.

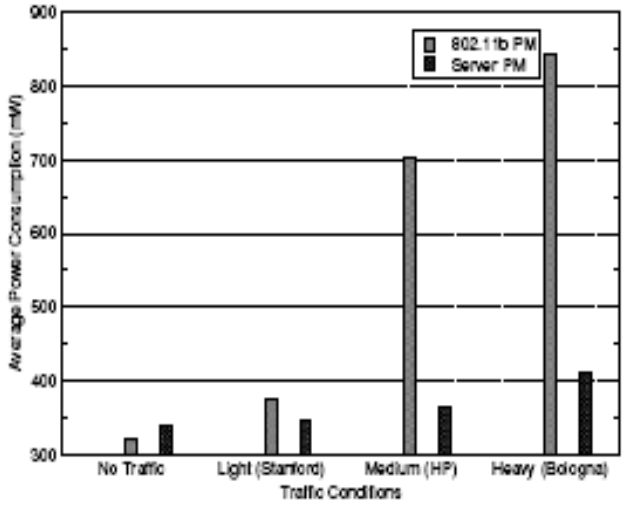


Fig. 6. Power consumption with different traffic levels

5 Conclusion

In this work we presented a new methodology to improve the energy efficiency of portable wireless systems that operate in a client-server fashion. Our system enables the server to exploit knowledge of the workload, traffic conditions and feedback information from the client in order to minimize client WLAN power consumption. We tested our methodology on the SmartBadge IV wearable device running a streaming video application. Our server controlled approach saves 67% of energy as compared to leaving the client WLAN card always on, and 50% as compared to the simple 802.11b PM policy.

As an application, a real-time interactive client/ server system has been implemented. Text and audio have been used as communication media. The software system is written multithreaded and offers a great flexibility in opening and closing connections and adding new modules because of a consequent use of objectoriented design methods. For audio exchange the system showed poor performance due to straight-forward transmission and reception of audio data. In future we hope to use some state of the art encoding techniques for audio data. Video data transmission can also be included in future applications.

.

References:

[1] S. Cuk, Discontinuous Inductor Current Mode in the Optimum Topology Switching Converter, *Proc. of the IEEE Power electronics Specialists Conference, IEEE PESC'*78, 19

[1] A. Acquaviva, L. Benini, B. Ricco, "Software Controlled Processor Speed Settinj for Low-Power Streaming Multimedia," *IEEE Trans, on CAD,* Nov. 2001.

[2] F. Bellosa, "Endurix: OS-Direct Throttling of Processor Activity for Dynamic Powei Management," *Technical Report TR-I4-99-03, University of Erlangen,* June 1999.

[3] C. Chiasserini, P. Nuggehalli, V. Srinivasan, "Energy-Efficient Communication Pro tocols", *Proc. of DAC,* 2002.

[4] IEEE LAN/MAN Standards Committee, "Part 11: Wireless LAN MAC and PH\ Specifications: Higher-Speed Physical Layer Extension in the 2.4 GHz Band", 1999.

[5] C. Jones, K. Sivalingam, P. Agrawal, J. Chen, "A Survey of Energy Efficient Network Protocols for Wireless Networks", *Proc. of DATE,* 1999.

[6] R. Krashinsky, H. Balakrishnan, "Minimizing Energy for Wireless Web Access with Bounded Slowdown", *Proc. of MOBICOM,* 2002.

[7] R. Kravets, P. Krishnan, "Application-Driven Power Management for Mobile Com munication," *Proc. of WINET,* 1998.

[8] P. Lettieri, C. Schurgers, M. Srivastava, "Adaptive Link Layer Strategies for Energy Efficient Wireless Networking", *Wireless Networks,* no. 5, 1999.

[9] J. Lorch, A. J. Smith, "Software Strategics for Portable Computer Energy Management," *IEEE Personal Communications,* June 1998.

[10] Y. Lu, L. Benini, G. De Micheli, "Operating System Directed Power Reduction," *Proc. of ISLPED,* July 2000.

[11] C. Luna, Y. Eisenberg, R. Berry, T. Pappas, A. Katsaggclos, "Transmission Energy Minimization in Wireless Video Streaming Applications," *Proc. of Asilomar Conf. on Signals, Systems, and Computers,* Nov. 2001.

[12] R. Min, A. Chandrakasan, "A Framework for Energy-Scalable Communication in High-Density Wireless Networks," *Proc. of ISLPED,* 2002.

[13] T. Pering, T. Burd, R. Brodersen, "Voltage Scheduling in the IpARM Micropro cessor System", *Proc. of ISLPED,* July 2000.

[14] V. Raghunathan, S. Ganeriwal, C. Schurgers, M. Srivastava, *"E^2WFQ:* An Energy Efficient Fair Scheduling Policy for Wireless Systems", *Proc. of ISLPED,* 2002.

[15] J. Flinn, M. Satyanarayanan, "Energy-aware adaptation for mobile applications," *Proc. of SOSP,* Dec. 1999.

[16] P. Shenoy, P. Radkov, "Proxy-Assisted Power-Friendly Streaming to Mobile Devices," *Proc. of MMNC,* Jan. 2003.

[17] E. Shih, P. Bahl, M. Sinclair, "Dynamic Power Management for non-stationary service requests", *Proc. of MOBICOM,* 2002.

[18] E. Iancu „*Modeling and simulation of failures in dynamic systems"* WSEAS Transactions on Circuits and Systems - Special Issue: Optimization, Simulation, Modeling and Control in Systems Applications, Issue 4, volume 2, October 2003, ISSN: 1109-2734, pag. 716-721.

[19] Niculescu, E., Small-signal properties of the modified boost topology, *WSEAS Transactions on Electronics*, Issue 2, Volume 1, 2004, pp. 436 – 441

[20] Purcaru, D., Niculescu, E., Purcaru, I., Experimental Measuring System with Rotary Incremental Encoder, *WSEAS Transactions on Advances in Engineering Education*, Issue 9, Vol. 4, September, 2007, pp. 180-186