SWAN: An Indoor Wireless ATM Network

E. Hyden, J. Trotter, P. Krzyzanowski, M. Srivastava and P. Agrawal
AT&T Bell Laboratories
600 Mountain Avenue
Murray Hill, NJ 07974
{eah,trotter,paul,mbs,pa}@research.att.com

Abstract— **The SWAN[1] project investigates architectural aspects of networks containing mobile hosts. The network model includes base stations connected via a wired, ATM infrastructure and a wireless, ATM last hop to a number of mobile hosts, ranging in computational and functional abilities from personal digital assistants to notebook computers. The FAWN[2] network interface card was developed as part of the implementation of a small SWAN network. It provides wireless, ATM channels using off-the-shelf FHSS modems which operate in the 2.4 GHz industrial, scientific and medical band, and use a sequence of radio channels of 1 MHz bandwidth to provide up to 625 kb/S raw bit rate. The first phase of the implementation has been completed and provides basic wireless ATM connectivity. This paper presents an overview of the SWAN network, followed by a description of the implementation which has been built to validate our initial architectural design.**

Keywords— **Wireless Networks, Asynchronous Transfer Mode, Mobile Computing.**

I. Introduction

CURRENT wireless technology can provide data links ranging in speeds from tens of kilobits to almost a gigabit per second to meet the needs of applications as diverse as paging and local area network connectivity. The basic technology is still developing, but trends can be identified; infra red (IR) appears well suited to peripheral control and interconnection on the desktop or within a small room, and spread spectrum techniques such as Direct Sequence (DS) and Frequency Hopping (FH) are popular within the telephony and computer industries.

Miniaturisation, low power electronics and evolving display technologies allow increasing functionality to be packaged into easily portable devices such as Personal Digital Assistants (PDA) and notebook computers. The ease with which these small, lightweight devices can be transported and the desire to integrate them as seamlessly as possible into a user's computing environment using wireless technology is driving current research in mobile computing.

The SWAN project [1] was initiated within the Networked Computing Research department at AT&T Bell Laboratories to study this new aspect of computing. Broadly, the aims of the project are to gain some familiarity with the aspects of computing which are related to mobility and, in the longer term, to identify principles and engineering guidelines which can be used by developers to build successful wireless networks.

Initially, we have chosen to focus on the problem of providing indoor, wireless access to mobile hosts via a wired local area infrastructure. The network topology is restricted to one providing a wireless last hop. Figure 1 depicts a system representative of a typical SWAN network; two local area domains are connected by a wide area fabric. The aim is to provide uninterrupted network connectivity to mobile hosts as they roam, moving between the base stations within a local area domain and even between domains. The latter case may result in periods of weak connectivity or even disconnection. A SWAN base station provides coverage for an area from 10 to 20 meters in diameter, servicing typically one or two offices or a section of corridor within a building. High data rates and frequent handoffs are characteristic of such service area sizes.

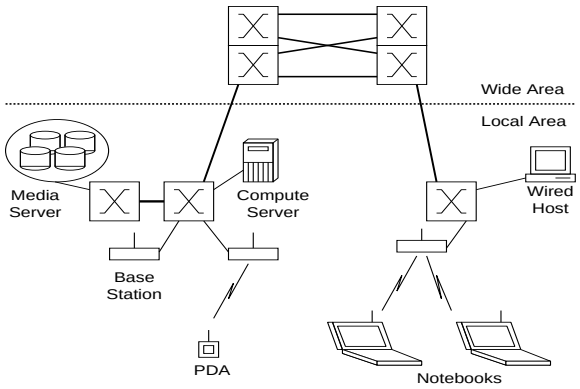

Fig. 1. Networking environment.

Endpoints in the network range in functionality and mobility from PDAs to notebook computers and include entities such as print servers, which may seldom be moved. While mobile hosts may have significant computational power, it is likely that they will be used more for communicating than for computing. Power usage restrictions will be the governing force for the foreseeable future, so the bulk of any processing will be done by servers attached to the local infrastructure, while the processors in the mobile hosts will be used primarily for information filtering, compression and decompression, encryption and decryption and capture and presentation. Audiovisual data are important communications media and multiservices networks which use the Asynchronous Transfer Mode (ATM) [3] are well suited to carrying these types of traffic, so ATM has been adopted as the networking paradigm for use within SWAN.

While SWAN has an obvious role in providing the wire-

[1] Seamless Wireless ATM Network

of applications which are available rely on Internet protocols such as IP, UDP and TCP. During the planning of the project, an early milestone was set after which it would be possible to use SWAN's ATM links to transfer IP datagrams. Achieving this raises issues such as how IP-over-ATM ought to be implemented in a wireless ATM setting. Once this milestone has been achieved, mobile hosts within our network immediately have access to a plethora of IP-based applications such as `mosaic` and the mbone tools `nv` and `vat`.

This paper describes the system as it exists after reaching the first milestone. A point-to-point Medium Access protocol (MAC) transfers cells between a mobile and a base station, an ATM Adaptation Layer (AAL) converts between IP datagrams and cells, and IP traffic is transferred on Virtual Circuits (VCs) reserved for carrying that particular type of traffic.

II. Network Interface Card

Wireless hop connectivity is provided by the FAWN (Flexible Adapter for Wireless Networking) card [4] which provides an interface between a PC card[3] bus and an RF modem. The RF modem was chosen because of availability, cost and licensing conditions. The FAWN card only requires access to transmit and receive data streams and can make use of a received signal strength indication, so the choice of modem is not critical and future versions will likely take advantage of newer, faster modems as they become available. A PC card interface was chosen so the same

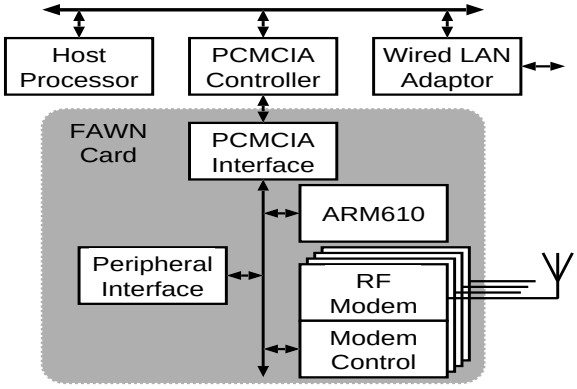

Fig. 2. Fawn network interface adaptor.

card can be used in many applications including PDAs and notebooks as well as in base stations. Because mobility is an important aspect of this research the prototype FAWN card was designed to be easily portable. Its dimensions are 10.8 cm (W) × 1.9 cm (H) × 11.4 cm (D) and it fits into the bay which holds the removable floppy drive in an AT&T Safari 3181 or similar notebook computer. The architecture of the FAWN card is shown in figure 2. The card includes an embedded processor, the ARM610, which provides the processing power to implement the MAC as well as some ATM queue management functions. Communication between the FAWN and PC card interface is implemented using a Dual Port RAM interface which allows fast access to a shared area of memory. Most of the physical level functions are implemented in an FPGA which contains packet buffers, a real time clock and provides an interface to the received signal strength. The FAWN card can be adapted for use in many different systems. In a base station several FAWN cards are used to provide the multiple channel access for FHSS mobile systems. In a notebook or PDA one card is plugged into a PC card slot and in a peripheral, such as a printer, a FAWN card is embedded in the device to provide the wireless connectivity.

The ARM610 on the FAWN implements a simple, point-to-point medium access protocol, and manages the cell queues in the DPRAM. This requires approximately 35 KB of code, which is organised as three concurrent processes. The main line program code initialises the host/fawn DPRAM interface and RF modem hardware, then becomes the idle loop. The modem hardware asserts the ARM610's Fast Interrupt request (FIQ) input whenever a transmit buffer has been emptied, the received signal strength exceeds the programmed threshold, a receive buffer has been filled, or the countdown timer has expired. These events are used to drive the modem management process which controls the bidirectional flow of cells between the cell queues in the DPRAM and the RF modem buffers. Host software can assert the low priority Interrupt Request (IRQ) input to indicate that cells have been placed in the DPRAM for transmission, and the ARM610 can generate an interrupt on the host to signal that received cells are available in the DPRAM.

III. Cell Format

The modem control hardware provides two, 64-octet cell buffers for feeding the modem transmit stream, and two for draining the receive stream. This implementation suggests a wireless cell size of 64 octets, which is sufficient to accommodate a standard BISDN cell consisting of a 5-octet header and a 48-octet payload, leaving 11 octets spare. It has been suggested that this space might be used to contain Forward Error Correction (FEC) information which could be used to recover from bit errors, but in a busy FHSS environment, burst errors will be induced when two hopping patterns collide, scrambling multiple consecutive cells and rendering FEC ineffective. Wireless bandwidth is a precious resource, so for the purpose of the first project milestone, the cell format chosen has a 4-octet header and a 60-octet payload. Figure 3 shows the format of a SWAN wireless cell. The first octet contains a frame start (FS) bit which indicates the first cell in a segment. The sequence (SEQ) field contains the ordinal of the cell in a segment and counts backwards so that the first cell in a segment has a sequence field of one less than the total number of cells in the segment, and a sequence value of zero indicates the last cell in a segment. The VCI field identifies the virtual circuit to which this cell belongs and the CRC field contains an 8-bit CRC generated from the header data. The payload contains up to 60 bytes of user data. This cell

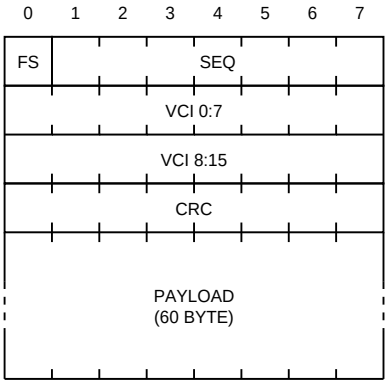

Fig. 3. Wireless cell format.

interworking with wired ATM LANs is more important; a 48-octet payload will reduce the amount of effort required to translate cells between circuits.

IV. Medium Access

The MAC divides the wireless bandwidth between receive and transmit and allows the mobile systems to establish and continue communication with a base station. The mobiles and base stations operate on one of several channels which can be hopping sequences for the case of our frequency hopping modem. The MAC communicates using frames of 10 cells each. Each cell is 64 bytes long and takes about 1 mS to send and a frame takes about 10 mS. The time to turn the channel between receive and transmit is approximately 1 mS. These figures were chosen because the maximum transmit time of our modem is 10 mS [2].

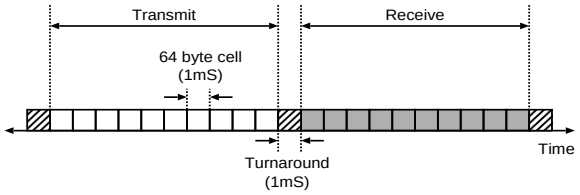

Fig. 4. MAC duplexing.

A mobile and a base station communicate by exchanging frames of 10 cells each. If a host has data to send, a frame will contain valid cells, otherwise it will contain dummy cells which are discarded by the receiver. When a mobile is powered up it beacons periodically by transmitting a frame. Once this frame is successfully received by a base station the base station acknowledges the frame by sending a frame in reply. Once communication is established the base-station and mobile continue to exchange frames of data. If no reply to a transmission is received, the receiver will timeout and begin to beacon to re-establish communication. Figure 4 shows a timing diagram of the duplexing scheme which statically allocates equal bandwidth for receive and transmit streams. Higher level software sees this as two channels which are capable of transferring 10 cells every 20 mS, giving bandwidths of approximately 300 kb/S is no association between a particular ATM virtual circuit and a MAC slot. Rather, the code responsible for multiplexing cells onto the channel sees 10 slots become available every 20 mS, and is free to transmit the next cell from any queue in the DPRAM as it pleases. While the allocation of bandwidth between transmit and receive channels is static, allocation within a transmit or receive channel is dynamic. A more complete version of the MAC which is under development allows the transmit-receive partitioning to be changed as the demands of the link vary.

V. Maintenance Software

To control a FAWN card, a small amount of support is required which is implemented as a device accessed via two special files (`/dev/fawn/mem` and `/dev/fawn/ctl` in our system). `mem` allows host applications to read and write memory and registers in the ARM610's address space, the first 4 MB of which contain the SRAM which holds the ARM610's code and data. Writing a string to `ctl` causes the string to be interpreted with the results applied to the FAWN control registers. Reading `ctl` returns a string which describes the status of the interface. The commands

```
$ echo reset > /dev/fawn/ctl
$ cat k.i > /dev/fawn/mem
$ echo run > /dev/fawn/ctl
```

when issued from the host by a suitably privileged user cause the ARM610 to be reset, the image `k.i` to be loaded at the beginning of SRAM and the ARM610 to be released from reset to begin execution.

VI. Device Driver

Figure 5 shows the basic functions of the device driver written to interface the FAWN card with the lower levels of the IP protocol stack. Segmentation code appends a

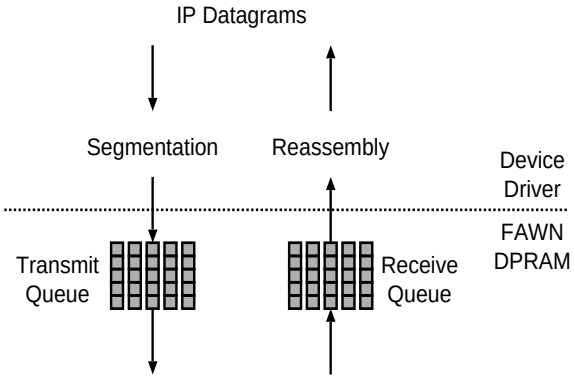

Fig. 5. Driver adaptation.

length field and checksum to transmitted datagrams then fragments them into cells which are placed into the queue associated with the VC for IP traffic. Upon reception, cells are reassembled into a single segment which, if successfully

VII. Performance

Figure 6 shows the topology of the network used during experimentation. `tun` is a 90 MHz Pentium/PCI PC,

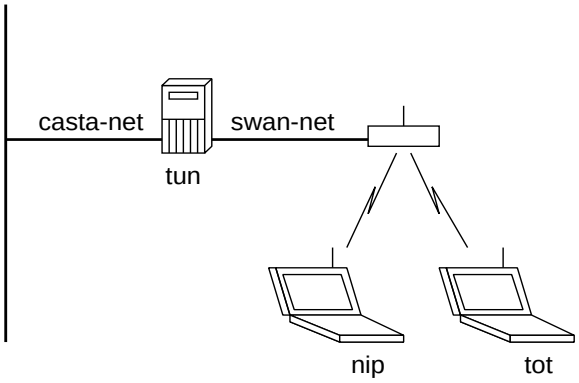

Fig. 6. Network configuration.

configured as a gateway between `swan-net` and the departmental ethernet; `nip` and `tot` are respectively a NEC VERSA 100/4 and an AT&T Safari 3181 notebooks. In the results which follow, TCP throughput was measured using `ttcp`[4] to transfer a 1 MB file from `tun` to `nip` over the wireless link, with no other traffic on the link. The antennas were placed sufficiently close to each other that the error rate on the channel was minimal. A nearby spectum analyser showed that no abnormal interference was present at the time of the experiments. Using the standard frame size of 10 cells for both transmit and receive frames, the test file was transferred in 36.97 seconds, yielding a TCP transfer rate of 28.36 KB/S (227 Kb/S).

A. MAC Frame Size

MAC frame size is an important parameter over which the system designers have significant control. Figure 7 shows the effect that varying MAC frame size has on TCP transfer rate.[5] This figure reveals that small frame sizes give rise to low transfer rates. This is to be expected because it takes some time (approximately 1 mS) to turn the wireless link from transmit mode to receive mode and *vice versa*, and small frame sizes do this more frequently, yielding lower data rates. The graph shows that a choice of 10 cells per frame is quite acceptable; transfer rate drops off rapidly for smaller frame sizes, but much larger frame sizes do not obtain significantly higher transfer rates.

B. Transfer Size

Figure 8 shows a plot of ping times for a range of payload sizes. The times are the average times calculated from 10000 individual samples for each payload size measured. Payload sizes were measured in 60-byte increments. Payload sizes smaller than `A` fit into a single cell and result

[4] Available as `ftp://ftp.sgi.com/sgi/src/ttcp/`.

[5] Although frame sizes greater than 10 mS drive the modems beyond their documented specifications, the lack of any decrease in TCP throughput shows that the pair used in this experiment did not

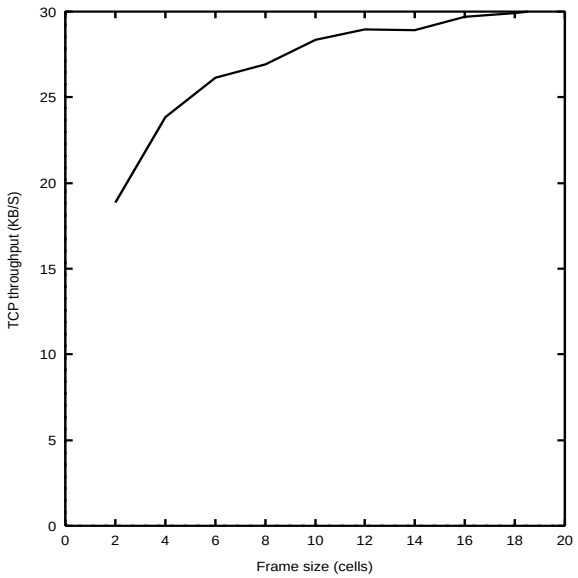

Fig. 7. TCP throughput versus frame size.

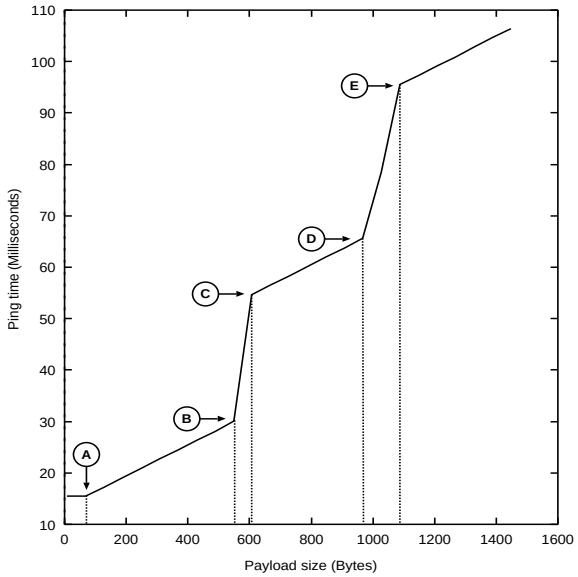

Fig. 8. Ping time in milliseconds versus payload size in bytes.

in a constant ping time. From `A` to `B`, increasing payload sizes correspond to a proportionate increase in the number of cells transmitted increasing ping time at a rate of approximately 2 mS per cell. `B` to `C` represents an increase in ping time caused by the payload starting to span two frames. The average ping time is increased by about 25 mS because the time to transmit a payload now has to include an unused receive frame between the two transmit frames. From `D` to `E` we can expect a similar increase due to the transmission now including two unused receive frames. Since the MTU is 1024 bytes, payload size at this stage is increased by one or two cells due to IP fragmentation.

C. Delay

The MAC frame structure induces noticeable effects in

ation in ping times for two sequences of 200 consecutive packets transmitted at intervals of 40 mS and 50 mS respectively. In both cases the payload consisted of two cells. In Figure 9, the smallest ping time is observed when both

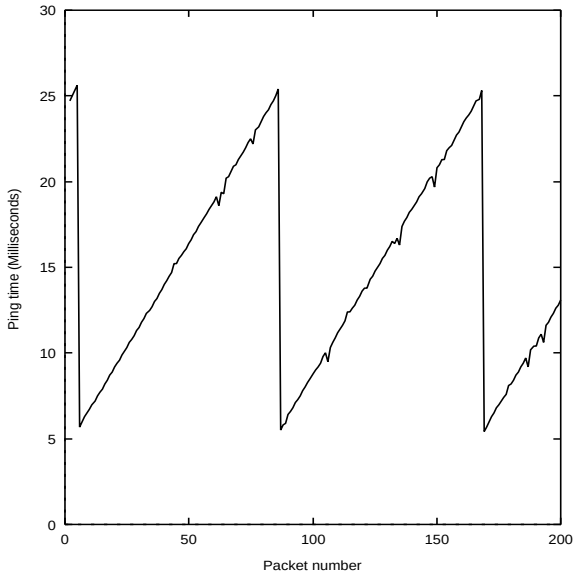

Fig. 9. Ping time in milliseconds versus packet number. Time between consecutive packets is 40 milliseconds.

ping cells are generated at the end of a transmit frame so that the corresponding reply arrives at the start of the next receive frame. The ping period is not perfectly synchronised with the frame period, so a gradual drift occurs. As the arrival of the ping cells is further removed from the end of the transmit cycle, the observed response times increase. In the worst case, the ping cells are generated at the extreme end of the transmit frame, such that one of the cells misses the current frame and has to wait for the next transmit frame. The same process is occurring in Figure

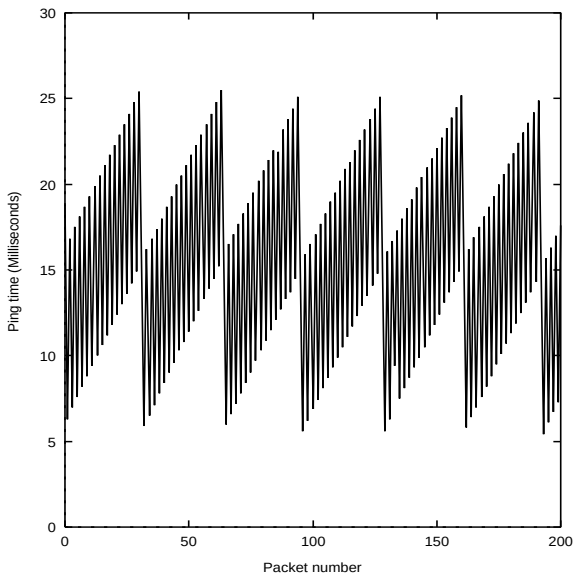

Fig. 10. Ping time in milliseconds versus packet number. Time

10, except that in this experiment, the ping period of 50 mS causes consecutive transmissions to occur on alternating transmit and receive frames. This results in a sequence of alternating shorter and longer ping times.

D. Discussion

Our early results highlight some interesting challenges. Firstly, a TCP transfer rate of 28.36 KB/S shows that we are utilising about 227 Kb/S or 73% of our 312 Kb/S bandwidth in a single direction. Secondly, wireless bandwidth is precious, and to utilize it using our hardware we have adopted a particular ATM cell format. In our next generation MAC, we expect to use a 48-byte payload for ease of interworking with wired ATM networks, ATM networks, but for efficient use of the wireless bandwidth, we may reduce the size of the cell header. The hardware cell buffer size will be altered accordingly. Thirdly, the wireless channel is implemented within a shared meduim and we have less control over it than we would have in a wired, switch based network, so the provision of any Quality Of Service (QOS) guarantees which are typically associated with ATM networks is made much more difficult. The ping times plotted in figures 9 and 10 are good examples of the types of challenges to be faced. The times are essentially an estimate of end-to-end delay, an important QOS parameter. Even on a simple point-to-point link such as the one used for our experiments, the effects of having to share the wireless medium between just transmit and receive channels shows a marked effect on the delays achievable.

VIII. Summary

We have presented the current status of SWAN — a project aimed at providing wireless network connectivity in a mobile computing environment. The results presented represent delay and throughput which make a large number of applications usable on the mobile hosts. Interactive applications such as editors and remote shells suffer no appreciable delay, and the available bandwidth is sufficient to support NFS mounted filesystems, web browsing and audio and video clients. Work is currently under way on a new MAC which provides explicit support for mobility and QOS.

References

[1] P. Agrawal, A. Asthana, M. Cravatts, E. Hyden, P. Krzyzanowski, P. Mishra, B. Narendran, M. Srivastava and J. Trotter, *An Indoor Networked Wireless Computing System*, ICC '95

[2] GEC Plessey Semiconductor, *WLAN Handbook*, GEC Plessey Semiconductors, December, 1993.

[3] D. Raychaudhuri and N. Wilson, *ATM-Based Transport Architecture for Multiservices Wireless Personal Communication Networks*, IEEE Journal on Selected Areas in Communciations, Vol. 12, No. 8, October, 1992, pp. 1401–1414.

[4] J. Trotter and M. Cravatts, *A Wireless Adapter Architecture for Mobile Computing*, Proceedings of the Second USENIX Symposium on Mobile and Location-Independent Computing, Ann Arbor, Michigan, USA, April 10-11, 1995, pp. 25–31.