An experimental indoor wireless network

SWAN: A Mobile Multimedia Wireless Network

Prathima Agrawal, Eoin Hyden, Paul Krzyzanowski, Partho Mishra, Mani B. Srivastava, and John A. Trotter

SWAN (Seamless Wireless ATM Network) is an experimental indoor wireless network that investigates the combination of wireless access with multimedia networked computing in an indoor setting. It is based on room-sized pico-cells and mobile multimedia endpoints. It enables users carrying multimedia endpoints, such as personal digital assistants (PDAs), laptops, and portable multimedia terminals, to seamlessly roam while accessing multimedia data resident in a backbone wired network. The network model of SWAN consists of basestations connected by a wired asynchronous transfer mode (ATM) backbone network, and wireless ATM last hops to the mobile hosts. SWAN is one of the first systems to realize the concept of a wireless and mobile ATM network. Mobile hosts as well as basestations are embedded with custom-designed ATM adapter cards called FAWN (Flexible Adapter for Wireless Networking). FAWN uses off-the-shelf 2.4 GHz industrial, scientific, and medical (ISM) band radios.

After giving an overview of the SWAN network model, and discussing the challenges in making ATM wireless and mobile, the article describes the first phase implementation of SWAN hardware and software. This initial implementation provides connectivity over the wireless last hop. We have investigated both native-mode end-to-end ATM communication across the wired ATM backbone and wireless ATM links, and Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) communication using Internet Protocol (IP)-over-wireless-ATM in the wireless link with IP forwarding and segmentation-and-reassembly modules at the base stations. The latter mode of communication has allowed experimentation with various readily available TCP- and UDP-based multimedia applications such as *nv*, *vat*, and so on. Typical TCP throughput is 227 kb/s, with the raw channel bandwidth equally divided into 312 kb/s each in the transmit and receive directions, while round-trip delays over the wireless hop range from 5 ms to 25 ms. Performance metrics such as throughput, delay, and delay jitter are affected by various wireless link attributes. This article presents experimental data that explores the relationship between performance and wireless link attributes such as the frame size used to transfer data over the air, and the number of mobiles sharing the link.

Toward Integrated Service Wireless Networks

"Anytime, anywhere" information access and processing are much cherished in modern society because of their ability to bring flexibility, freedom, and increased efficiency to individuals and organizations. Businesses can use ubiquitous information access to streamline workflow by allowing employees to access and update databases, and receive work orders, directly at their logical work site or point of service. Examples include access to patient charts and medical data for doctors and nurses from handheld terminals, and electronic delivery of trade orders directly to traders carrying PDAs on a busy stock exchange floor.

Wireless access technology, by providing ubiquitous and tetherless network connectivity to mobile users, has to some extent already realized the vision of anytime, anywhere access in two specific domains: cellular voice telephony and wireless data networks. Cellular telephone networks have extended the domain of circuit-switched telephone service over a wireless last hop, while wireless data networks such as WaveLAN [1] for local area networks (LANs), Metricom's Ricochet for metropolitan area networks (MANs), and cellular digital packet data (CDPD) from various cellular carriers for wide area networks (WANs) do the same for users of TCP/IP data packet networks.

The degree to which wireless access has succeeded in the voice and data domains has clearly been different. Despite many technical weaknesses, cellular telephone networks have been a market success, enjoying market growth rates between 35 and 60 percent per year for the past decade and a subscriber base of many tens of millions in the U.S. alone [2]. The now mature analog cellular network technology is evolving into digital cellular networks, low-earth orbit satellite-based cellular networks, personal basestations, and wireless private branch exchanges (PBXs). Wireless data networks, on the other hand, are relatively immature, with a much smaller market penetration. A multitude of factors are responsible for this, ranging from low data rates and high cost in outdoor data networks, to the rather chaotic standards situation and consequent user frustration that exist in higher-speed wireless LANs. While the users of wireless data networks are not quite as numerous as those of wireless voice networks, their number is expected to triple from 1995 to 1997 as wireless data networks become easier and cheaper to use, and more powerful and less expensive mobile devices arrive [3].

Technically, realizing the SWAN vision is not just a bandwidth or system capacity issue. Equally important are controllable bandwidth and a quality of service (QoS) environment for multimedia in the presence of wireless and mobility.

The state of wireless access can thus be summed up as a choice between cellular telephone networks that provide voice bit rates with rigid service quality, and wireless data networks that look like the wired data networks of yesteryear, with limited data rates and no notion of service quality. In recent years wired networks have begun to evolve toward integrated service data networks (ISDNs) with a multimedia-oriented network model. This is due to the availability of user devices capable of multimedia communication (such as multimedia PCs and workstations), and the need for service providers to flexibly and efficiently multiplex multimedia connections in their networks.

The goal of the SWAN mobile computing environment being developed in the Networked Computing Research Department at Bell Laboratories is to drive wireless networks toward a similar integrated service model. The dream is to enable mobile context-aware multimedia services to users carrying heterogeneous portable endpoints with varying degrees of smartness. For example, in indoor settings, which is our initial focus, we want a multimedia networked computing and communication system to subsume the functionality and calling models of wireless LANs and PBXs. When deployed in a campus or an office, this integrated network will support audio/video calls on wireless communicators (with services such as call forwarding and conferencing); multimedia messaging and paging; multimedia applications on wireless PDAs, laptops, and terminals; and access to movable wireless gadgets such as video monitoring cameras.

The SWAN Approach

Technically, realizing the SWAN vision is not just a bandwidth or system capacity issue. Equally important are controllable bandwidth and a quality of service (QoS) environment for multimedia in the presence of wireless and mobility. Clearly, technology trends are encouraging. Continual progress in wireless technology will soon make it possible to economically provide per-user data rates of several megabits per second, at least inside buildings. Already the readily available radio modems operating in the 900 MHz and 2.4 GHz ISM bands, and in the newly allocated 1.91–1.93 GHz data personal communications services (PCS) band offer 1–2 Mb/s data rates per channel. Higher-speed radios, such as those operating in the 5.8 GHz band, will be available in the not too distant future. When operated in a pico-cellular configuration with spatial channel multiplexing and reuse, such multichannel radios can support a reasonable user density. Such wireless networks, together with the emerging integrated service wired network infrastructure and progress in audio/video compression algorithms, will permit seamless delivery of packetized multimedia information to a mobile user, at least indoors. Progress in packaging, display, and low-power circuits [4] has resulted in portable multimedia endpoints [5, 6] that seamlessly integrate into a user's networked computing environment.

SWAN, as its expanded name *Seamless Wireless ATM Network* suggests, seeks to provide continual network connection to mobile heterogeneous ATM endpoints. We are building a prototype system by exploiting the synergy between the above-mentioned technology trends in multimedia information access, wireless access technology, and portable multimedia endpoints. However, the available off-the-shelf radio technology is somewhat immature for our needs. Besides obvious bandwidth limitations for large user densities, typical ISM band radio cards often have limitations such as few if any software tunable channels, and fixed high radio frequency (RF) power levels which are more suitable for building-wide wireless LANs than for room-sized pico-cells. Realizing such limitations of the available off-the-shelf radio technology, we have attempted to keep our system design largely independent of any specific radio.

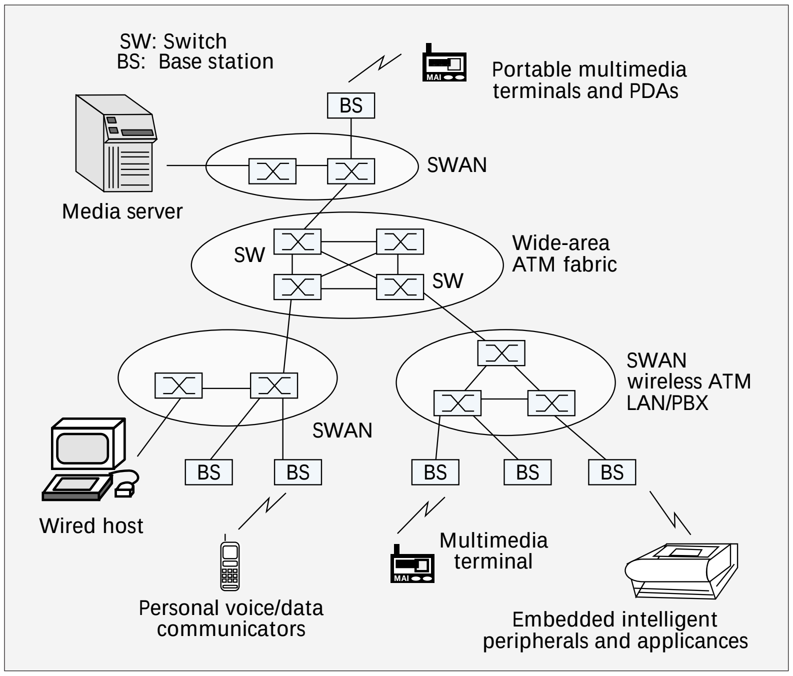
The network model of SWAN consists of base stations connected by a wired ATM backbone network, and wireless ATM last hops to the mobile ATM hosts. The system and network software in SWAN, called *Etherware*, uses novel connection management protocols developed by us to handle endpoint mobility and schedule wireless resources. By allowing applications to register interest, in the form of call-back handlers, to low-level events such as an impending handoff or a change in wireless channel bit error rate, Etherware provides a platform for a variety of mobile context-aware multimedia applications (i.e., applications that adapt to changes in their operating environment).

The mobile hosts as well as the base stations incorporate custom-designed adapter cards called FAWNs (Flexible Adapters for Wireless Networking) [7]. Currently, an off-the-shelf 2.4 GHz ISM band radio is interfaced to FAWN. FAWN provides embedded programmable processor and reconfigurable hardware resources for implementing medium access control (MAC) and air-interface subsystems to transport the cells[1] in ATM virtual circuits (VCs) over the air. Software programmability and miniaturized physical size enable the development of heterogeneous portable endpoints with varying degrees of smartness.

Enhancing ATM to make it wireless and mobile is a key feature of SWAN, which is among the first systems to implement this concept. SWAN's choice of ATM not only produces a homogeneous end-to-end network simplifying the architecture, but also has attributes such as QoS-specifiable VCs, which are quite useful in a wireless and mobile environment. ATM allows cell scheduling, error control, and handoff management based on QoS parameters, instead of treating all the traffic uniformly as current wireless data LANs do. Compared to research projects such as Berkeley's Infopad and UCLA's WAMIS, which also seek to provide a mobile user with audio, video, graphics, and data connectivity, a distinctive aspect of SWAN is indeed its use of ATM. For example, Infopad uses a TCP/IP backbone with a separate custom connection-oriented wireless hop transport mechanism between "dumb" terminals and base stations. WAMIS, which has adopted a packet-radio-type ad hoc networking model with endpoints also acting as relays, supports TCP/IP and VCs.

Besides our and other [8, 9] efforts within Bell Laboratories, NEC's WATMNet [10] and ORL Cambridge's wireless

[1] This article uses the word "cell" in some instances to refer to an ATM cell (unit of data), and in other instances to refer to the geographical area surrounding a base sation in a cellular network. Both are well accepted standard terms, but the intended usage will usually be clear from the context.



SW: Switch
BS: Base station
Portable multimedia terminals and PDAs
BS
SWAN
Media server
SW
Wide-area ATM fabric
SW
SWAN wireless ATM LAN/PBX
SWAN
BS
BS
BS
BS
BS
Wired host
Multimedia terminal
Personal voice/data communicators
Embedded intelligent peripherals and applicances

■ **Figure 1.** *Network communication model of the SWAN wireless ATM network.*

ATM system [11] are the two other concurrent wireless research systems that are implementing wireless and mobile ATM, though with different approaches and scope. Any mobile and wireless ATM network obviously requires solutions to problems such as the appropriate hardware for transmitting and receiving ATM cells over the air, and VC connection and QoS management in the presence of mobility. SWAN's solutions to such lower-level problems have been presented elsewhere [7, 12, 13]. Compared to other wireless and mobile ATM efforts, the SWAN system is distinguished by novel low-latency VC rerouting algorithms based on performance-triggered rebuilds [12], custom reconfigurable and miniature wireless ATM adapter hardware [7], and support for heterogeneous end systems ranging from laptops to dumb multimedia terminals [5].

In this article we describe the overall system architecture of SWAN, its first-phase implementation, and initial performance data. We present SWAN's ATM-based network model in the next section, and discuss the rationale behind and challenges of wireless and mobile ATM in the following section. Next, the first-phase implementation of SWAN is described. This implementation provides connectivity over the wireless last hop. We have investigated both native-mode end-to-end ATM communication across the wired ATM backbone and wireless ATM links, and TCP and UDP communication using IP-over-wireless-ATM in the wireless link with IP forwarding and segmentation-and -reassembly modules at the base stations. The latter mode has allowed experimentation with various readily available TCP- and UDP-based multimedia applications. Our experience and performance data on the initial system implementation are presented at the end.

Network Communication Architecture of SWAN

Figure 1 shows a high-level view of the network communication model adopted by SWAN [14]. A hierarchy of wide-area and local-area wired ATM networks is used as the backbone network. Wireless access is used in the last hop to mobile hosts. In addition to connecting conventional wired server hosts and client endpoints, the wired backbone also connects to special switching nodes called *base stations*. The base stations are conventional PCs and workstations, and are equipped with the FAWN [7] custom wireless adapter cards. These act as gateways for communication between nearby mobile hosts and the wired network. Functionally, the base stations are special mobility-aware ATM switches that are located at the edge of the wired ATM cloud and have wireless link interfaces on one or more of the switch ports. In practice, given the slow speed of wireless links, the switching functionality may be realized in software instead of ATM switching hardware. The mobile hosts are also equipped with the FAWN adapter. The geographical area for which a base station acts as the gateway is a room-sized pico-cell. Network connectivity is continually maintained as users carrying a variety of mobile hosts roam from one cell to another. Although mobile hosts in SWAN may have different local general-purpose computing resources, all of them must have the ability to participate in network signaling and data transfer protocols. A mobile in SWAN sends and receives all its traffic through the base station in its current cell.

Endpoints in SWAN range considerably in functionality and mobility. They include "smart" PDAs and laptops, "dumb" multimedia terminals, and wireless entities such as print servers and cameras that move infrequently. While dumb endpoints are not expected to compute locally, even the smart endpoints that perform considerable local computation are likely to off-load the bulk of the processing to servers on the wired network. This is necessary to conserve available power and enhance battery life. Thus, the processing at the smart endpoints will be dominated by communication-intensive tasks such as information filtering, compression and decompression, encryption and decryption, and capture and presentation.

Our model uses end-to-end ATM over both the wired network and the wireless last hops. This is in contrast to the use of connectionless mobile-IP [15, 16] in present-day wireless data LANs such as WaveLAN [1]. The choice of ATM was motivated by advances in video compression algorithms and the availability of higher-bandwidth RF transceivers which permit the transmission of packetized video to a mobile. Support for multimedia traffic over the wireless segment was a driving force in SWAN. Adopting the connection-oriented paradigm of ATM VCs over the wireless hop allows QoS guarantees associated with VCs carrying multimedia traffic to be extended end to end. Other researchers, such as Raychaudhuri and Wilson [17], have also advocated the use of ATM-based transport in multiservices wireless personal communication networks. Rajagopalan [18] has proposed an architectural option for integrating wireless access and mobility into existing ATM networks by using special mobility-aware anchor nodes.

Extending ATM's VC paradigm all the way through to a mobile host necessitates continuous rerouting of ATM virtual circuits as a mobile host moves [19–22]. Indeed, Toh's measurements at Cambridge [22] show that VC rerouting can be quite fast. The small cell sizes and the presence of QoS-sensitive multimedia traffic make this problem particularly important in SWAN. VCs carrying audio or video, as far as possible, need to be immune from disruptions as a mobile host *hands off* from one base station to a neighboring one. ATM signal-

Advocating mobile and wireless ATM, particularly when existing products such as wireless LANs and CDPD are IP-based, naturally makes one a participant in the ongoing serious and somewhat religious ATM versus IP debate in the data networking community.

ing protocols need to accomplish the task of VC rerouting with minimum latency. At a lower level, the MAC and air-interface layers must support efficient transport of ATM cells, low-latency mobile handoffs, per-VC error control, and cell scheduling according to VC QoS parameters. Chandler *et al.* [23] have described an air interface for ATM cell transport in a code division multiple access (CDMA) wireless network. SWAN's solutions to these connection management, signaling protocol, MAC, and air-interface problems can be found in [12, 13], and form the algorithmic basis of our system implementation.

Rationale and Challenges of Mobile and Wireless ATM

Before attempting to discuss SWAN's approach to mobile and wireless ATM, one needs to answer the questions of what the rationale is for using mobile and wireless ATM, and what the challenges are in implementing it. These questions are interesting for at least two reasons. First, ATM was designed for an environment where the hosts do not move and are connected to a switch via a relatively error-free and high-speed point-to-point wired link. It is not obvious what enhancements are needed for ATM to work well in a mobile and wireless setting. Second, there is the inevitable comparison with using IP instead of ATM in a mobile and wireless environment. Advocating mobile and wireless ATM, particularly when existing products such as wireless LANs and CDPD are IP-based, naturally makes one a participant in the ongoing serious and somewhat religious ATM versus IP debate [24] in the data networking community.

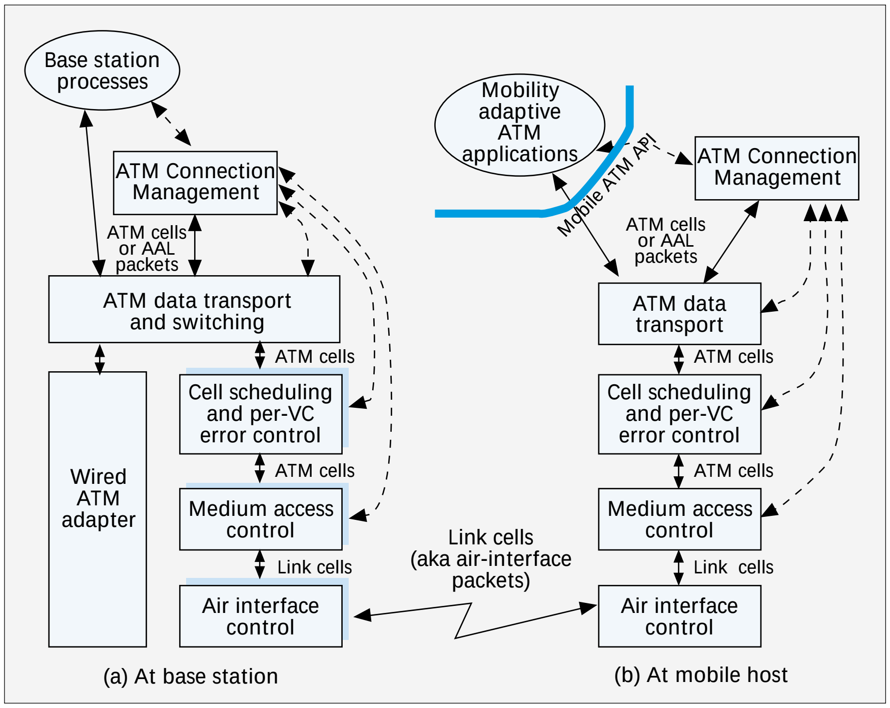
Rationale

Before delving deep into the rationale question, it must be pointed out that there is a homogeneity argument in favor of wireless and mobile ATM as envisioned in SWAN. ATM is a scalable technology, and appears likely to be at the core of future multimedia networks. Therefore, extending the QoS-specifiable ATM VC model over the wireless hop leads to a homogeneous end-to-end network, simplifying the architecture. In the wired world, extending ATM to the desktop has faced such hurdles as an entrenched and cheap last-hop access technology (Ethernet) and the higher cost of fiber and optical transceivers, which were required until the recent emergence of twisted-pair ATM. There are no such entrenched standards or obvious cost disadvantages in using ATM for wireless last-hop access.

The rationale for using wireless and mobile ATM can be explored along two dimensions that distinguish ATM: "cellification" and QoS-specifiable VCs. By "cellification" we refer to the fact that ATM uses fixed and small-sized cells (48-byte body with a 5-byte header). QoS-specifiable VCs refer to the use of switched VCs whose service quality parameters are negotiated at setup time by the endpoints with the network, and the network goes through a process of admission control and resource allocation before the connection is set up.

Superficially, cellification appears to be a complete disaster for wireless ATM, with a terrible waste of bandwidth for the last hop due to a large ATM cell header relative to the cell body. The reality, however, is more subtle. First, the fine-grained multiplexing provided by ATM cells is well suited to slow-speed links because it leads to lower delay jitter and queuing delays. Second, wireless links have high bit error rates. Therefore, one cannot use very large packets because that would lead to unacceptably large packet loss probability and would not work well with retransmission-based error control. Third, radios also impose limits on maximum continuous transmit burst. For example, this limit with the radio currently used in SWAN (next section) is 10 ms at 625 kb/s, or 6250 bits = 781 bytes, which poses a natural size limit on the data transmission unit. If one were to use IP transport, with a maximum transmit unit (MTU) of 781 bytes, the header overhead with IP datagrams would in the best case be 20/781 = 2.6 percent; by comparison, the overhead in the case of ATM is 9.4 percent. In reality, the two would be closer once overheads such as the MAC-level header, synchronization bits, and so on are taken into account. Fourth, one can be smart with ATM cell headers. By limiting the number of active VCs to a mobile (something that many wired ATM adapters also do), one can use a smaller number of bits to represent VCs for the duration of the wireless hop. For example, the 24-bit virtual channel identifier/virtual path identifier (VCI/VPI) can be transparently replaced by an 8- or 16-bit id over the wireless link between a base station and a mobile. A second technique that may be used is to cluster multiple ATM cells on the same VC into a variable-sized MAC frame and share the VC id fields. In essence, in these two techniques the base station acts as a gateway that can change cell format for the duration of the wireless link. Aside from the bandwidth waste argument, another cellification-related argument against ATM in the wired world is that the small cell sizes lead to cell processing constraints due to the limited time available per cell in a high-speed fiber optic network, clearly not an issue with wireless networks. In short, it suffices to say that the "cellification" issue over the wireless last hop is quite complex, with many trade-offs.

The rationale for wireless and mobile ATM is quite strong from the QoS-specifiable VC perspective. ATM VCs provide a useful traffic flow id for use at the wireless link level. For example, the MAC layer can use the VC ids to meaningfully allocate and schedule the shared wireless channel resources. MAC layers in current IP or internetwork packet exchange (IPX)-based wireless data LANs cannot do such intelligent wireless resource allocation. Similarly, the link-level error control mechanism can be suitably adapted on a per-VC basis depending on the characteristics of the individual VC. Furthermore, ATM's notion of specifying per-VC QoS is quite useful in a wireless and mobile network carrying multimedia traffic. QoS specification can be used, for example, to give some connections higher priority during handoffs, or to use rerouting policies tailored to the traffic characteristics. In general, VCs with QoS parameters in wireless and mobile ATM provide the ability to meaningfully distinguish the data packets being sent over the air, and not treat all of them according to one generic policy. Wireless and mobile ATM suffers from neither the rigid synchronous structure of cellular voice and PCS systems nor the inherently best-effort delivery model of wireless data LANs. It must be pointed out that the IP-based internet protocols are also being extended to allow multimedia applications that need service quality guarantees to express those needs to the network via resource reservation protocols such as RSVP [25]. To make the network honor traffic guarantees, it has been proposed that routes with reservations should be pinned or locked in place. This points to a



■ **Figure 2.** *Wireless and mobile ATM stacks.*

possible convergence of the IP and ATM service models. Research problems with QoS-specified pinned IP routes in a wireless and mobile network will be similar to those with QoS-specifiable ATM VCs.

Challenges

ATM was designed for an environment where the hosts do not move, and the transmission medium is a relatively error-free and high-speed point-to-point wired link. However, in a SWAN-like environment, the hosts move and the transmission medium is a relatively noise- and burst-error-prone shared medium of modest bandwidth. The change from a wired static world to a wireless mobile world can have unexpected pitfalls, as the experience of various researchers [26] with TCP performance in a mobile-IP wireless environment has shown — packet loss in the wireless hops due to noise, burst error, or handoff is falsely interpreted by TCP as being due to congestion, leading to poor performance. Clearly, it is not obvious what enhancements are needed for ATM to work well in a mobile and wireless setting.

The issues that need to be successfully addressed in making ATM work in a SWAN-like network fall into two somewhat orthogonal categories: mobility-related problems at the higher level and wireless-related problems at the lower level. From a mobility perspective, the key ATM issue is that of VC management in the presence of host mobility. Obviously, the VC route needs to be continually modified as the hosts at either end move during the lifetime of a connection. The rerouting must be done fast enough to cause minimal disruption to the transport layer. The signaling protocols for VC establishment, rerouting, and tear-down must function properly even in the presence of host mobility at both ends during the signaling phase itself. Any rerouting must maintain the sequence of ATM cells at the endpoints. Clearly, there would appear to be "obvious" solutions to some of these problems from analogous problems in the connection-oriented cellular and PCS world. However, the scale of the problem here is different in many dimensions, rendering the naive solutions too inefficient. The handoff frequency is much larger due to small cell size — imagine a person strolling down a hallway. Each terminal can terminate a large number of VCs, leading to a large number of VC reroutes on each handoff. The statistical multiplexing inherent in ATM and multimedia traffic requires a QoS renegotiation at the new base station, compared to the relatively simple frequency or time slot allocation in cellular and PCS base stations.

Although wireless is a last-hop issue, end-to-end ATM requires various components of the last hop to be aware of ATM traffic. From a wireless perspective, the key ATM issues are providing lower-layer support for ATM QoS, and efficient transport of ATM over a slow, noisy air medium. The first issue, QoS support, arises from the need for the MAC subsystem to participate in admission control at the time of VC admission and renegotiation at the time of handoff. Usually, MAC protocols are targeted at coordinating access of the shared air resource among multiple mobiles. With ATM, the MAC protocols need to schedule the air resource among multiple VCs at multiple mobiles. The second issue, efficient transport of ATM over the air, arises from the high header-to-payload ratio of ATM cells in an already low-bandwidth air medium, and the high wireless link noise and burst error rate. To address the former, one can use header compression and cell clustering techniques, mentioned previously. To address the latter, link-level error control schemes are needed. Here, ATM VCs offer the advantage that link-level error control schemes can be selected appropriately for each VC.

Finally, besides the specific mobility and wireless issues, there is a third related problem of which services application programming interfaces (APIs) a mobile and wireless ATM network should offer to native-mode ATM applications. Current ATM APIs are tailored for a static wired environment, and only offer basic services for VC establishment, tear-down, and data read/write. For a SWAN-like network, the API ought to be able to hide wireless and mobility from applications that wish to operate transparently, and to send wireless- and mobility-related events to applications that wish to adapt to changing network conditions. For the latter class of mobility and wireless adaptive applications, the lower layers of the network, such as MAC, must allow the higher layers to register interest in specific low-level events, such as an imminent soft handoff, and be able to send events back to the higher layers and the application when the specified low-level event occurs.

The rest of the article describes the architecture and implementation of SWAN, and how it addresses some of the above-mentioned challenges in making ATM wireless and mobile.

Mobility Management and Wireless Access Architecture

An initial system design of SWAN is complete and functional. Its focus is on the local-area domain and the wireless last hop. The wireless last hop consists of base stations and mobiles. SWAN uses room-sized cells, each of which has a base station equipped with one or more radio ports. Generic PCs and Sun workstations are used as base stations by plugging in a wired ATM adapter card and one or more FAWN RF wireless ATM adapter cards, and running necessary ATM switching and signaling software. The use of PCs and workstations also allows the base stations to host application and system processes if need be. The mobiles, at the other end of the wireless hop, include portable computers with an adjunct ATM wireless adapter or multimedia terminals with embedded ATM wireless adapters. The wired ATM backbone is currently composed of Fore's ATM switches. The software part of SWAN, called Etherware, consists of software modules running on the base station central processing unit (CPU), on the mobile host CPU, and on an embedded CPU on the wireless adapter card.

Taking a top-down view, in this section we describe the high-level architecture in terms of the protocols and algorithms embodied in Etherware. The first-phase hardware and software implementation is described in the next section.

Layers in Etherware

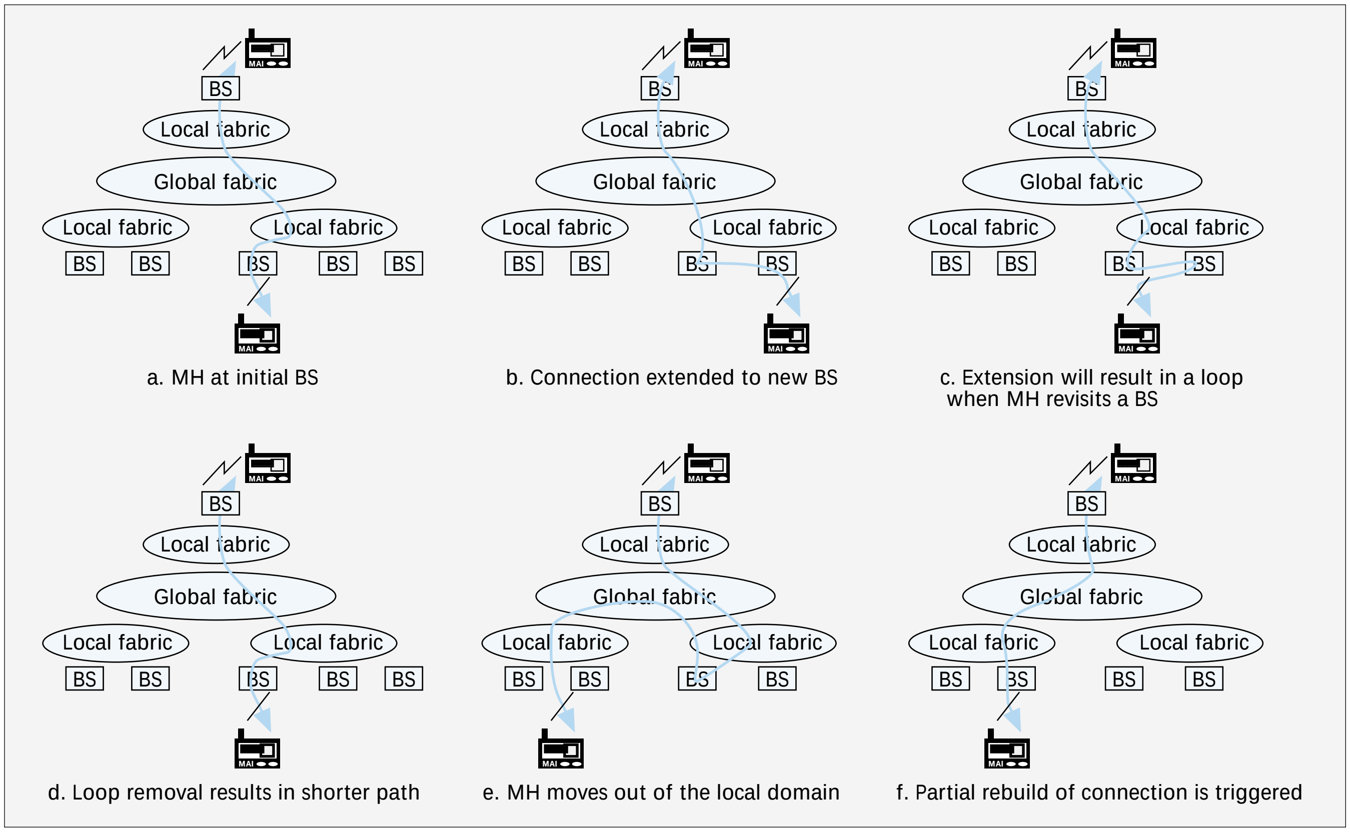
The key algorithms in Etherware are those used to handle ATM mobility and wireless access. These algorithms use a model in which the QoS requirements of applications are used to guide the operation of the protocols. There are four principal entities in SWAN: mobile hosts, base stations, wired switches, and wired hosts. These entities communicate with each other via a "signaling" protocol which handles addressing, location, and data forwarding functions. Being directly interfaced to the wireless link, of particular interest are the mobile hosts and the base stations. Figure 2 shows a logical view of the wireless and mobile ATM stack realized by Etherware at these two entities.

At the highest layer in the stack is the ATM Connection Management function which offers a mobile ATM API to the local applications and embodies state machines that participate in the necessary signaling to establish, maintain, and tear down VCs with specified QoS parameters in the presence of mobility. The signaling is done not only with the peer ATM Connection Management state machines at other mobile hosts, base stations, wired switches, and wired hosts, but also locally with the lower layers to negotiate resource allocation at VC setup and reroute, configure per-VC entries, and register interests in low-level events. The layer below does the necessary functions to transport data between the local applications and the available ATM link interfaces. At the mobile hosts this is simply segmentation and reassembly to the single (wireless) interface. At the base station, however, an ATM cell switching functionality is also needed to route cells among the multiple wired and wireless ATM link interfaces. Below this data transport and switching layer are functions associated with the wireless link for:

- Scheduling of cells in the VCs on the basis of the specified rate
- Link-level error control using forward error correction (FEC) and/or selective retransmission optionally specified on a per-VC basis at setup
- Medium access control (MAC) that supports QoS
- Air interface for efficient transport over the air

ATM Mobility Management

The design of the signaling software assumes the existence of separate signaling protocols for the wired and wireless networks, respectively, with base stations acting as gateways between the two logically distinct networks [26]. This allows us to minimize the changes required to the signaling infrastructure used in the wired network while allowing the signal-


MAI
BS
Local fabric
Global fabric
Local fabric
Local fabric
BS
BS
BS
BS
BS
a. MH at initial BS
b. Connection extended to new BS
c. Extension will result in a loop when MH revisits a BS
d. Loop removal results in shorter path
e. MH moves out of the local domain
f. Partial rebuild of connection is triggered

■ **Figure 3.** *VC rerouting using extension, loop removal and rerouting.*

Since carrying multimedia traffic to the mobiles is a major goal in SWAN, the two important drivers for the MAC and physical layer control subsystem are low-latency handoffs and support for multiple simultaneous channels in a given cell.

ing in the wireless network to be customized for the unique requirements of that environment. The signaling protocol in the wired network is used to establish, tear down, and rebuild connections. The signaling protocol in the wireless network is used to extend routes, remove route loops, and trigger route rebuilds, in addition to the establishment, teardown, and rebuilding of connections.

SWAN uses modifications to conventional signaling in wired networks to support the provision of end-to-end connectivity in the presence of mobility. The protocols are designed assuming small cells, with mobiles moving frequently between cells. The QoS seen by a VC is determined by the delay, loss, and throughput characteristics of the network path over which it is routed. These characteristics are governed by two phases: the transient disruption experienced by a connection when a handoff occurs, and the steady-state behavior when a mobile is not moving across cell boundaries. The goal of the signaling software is to maintain end-to-end connectivity while minimizing the impact of mobility on application-level QoS and also to ensure that network resources are used efficiently.

The protocols have some novel features. Since most of a mobile's movement will be within a local domain, the protocols perform *path extensions* only following a cell boundary crossing. However, when a domain boundary is crossed, building an extension will cause a more costly network path. This is used as a trigger to rebuild the routing tree. The signaling software allows an application to provide information about its QoS requirements, which is used to guide the actions taken following a handoff. A simple optimization that is used to improve network efficiency during the path extension process is to detect and eliminate loops when the same base station appears twice in the path. While the details of the above protocols for connection establishment and rerouting in the presence of mobility have been presented previously in [12], Fig. 3 summarizes the key aspects of our VC rerouting protocol.

Medium Access Control and Air Interface for Wireless ATM

Since carrying multimedia traffic to the mobiles is a major goal in SWAN, the two important drivers for the MAC and physical layer control subsystem are low-latency handoffs and support for multiple simultaneous channels in a given cell. In addition, explicit allocation of wireless resources among ATM VCs is crucial. Finally, at least initially, simplicity of implementation was considered desirable. Furthermore, as shown in the next section, we have based our wireless ATM adapter implementation on software and reconfigurable hardware so as to make algorithmic enhancements to the MAC and air interface feasible.

In the light of these considerations, the basic lower-layer strategy used in the SWAN prototype is to assign each mobile in a cell to its own radio port, or channel, on the base station. The available channels are distributed among the base stations with several channels to a cell, and base stations are accordingly equipped with multiple radio ports. The MAC implementation is thus simplified, having to deal only with the management of multiple channels, and not with the sharing of a channel by multiple mobiles. This is adequate for the use of our prototype in individual offices and small rooms with only a few (two or three) mobile endpoints each. More demanding usage patterns, such as handling conference rooms, will indeed require the ability to share a channel among multiple mobiles. This is being addressed in our ongoing work. It must be emphasized that the preceding simplified view of the lower layer is only an artifact of the current simplified implementation, not inherent to the SWAN architecture.

The implementation of the air interface and MAC does to some extent depend on the type of radio being used. Therefore, before describing the air interface and MAC in the following subsections, we briefly describe the radio transceiver in the next subsection.

Radio Transceiver — The SWAN system is largely independent of the specific radio being used. With SWAN being a pico-cellular architecture, the only architectural requirement is that the radio transceiver provide multiple software-controllable communication channels which can be spatially multiplexed and reused. Details such as the band of operation (ISM, PCS, etc.) and whether the radio uses direct-sequence or frequency-hopping spread spectrum are not factors *per se* in the choice of radio.

To meet our research needs, SWAN currently uses an off-the-shelf radio card which is a 625 kb/s half-duplex 2.4 GHz ISM band slow frequency-hopping transceiver with two power levels and two selectable antennas. Federal Communications Commission (FCC) requirements dictate that the radio must hop pseudo-randomly among at least 75 of the 83 available 1-MHz-wide frequency slots in the 2.405 to 2.4835 GHz region such that no more than 0.4 s are spent in a slot every 30 s. The radio incurs an 80 μs penalty per hop. Communicating transceivers hop according to a predetermined pseudo-random hopping sequence.

A *channel* in the current implementation of SWAN's wireless hop naturally corresponds to a hopping sequence, or a specific permutation of 75 to 83 frequency slots. Channels corresponding to two different frequency-hopping sequences interfere with each other when they occupy the same frequency slot at the same time. Therefore, geographically collocated channels use *weakly orthogonal* hopping sequences such that the chances of two different channels being in the same frequency slot at the same time are minimized. SWAN uses a family of 22 distinct hopping sequences, or channels, distributed among the base stations in various pico-cells. More than one channel can be allocated to a base station, and a base station needs to have a separate radio for each channel assigned to it. The same channel cannot be assigned to two base stations in cells that can mutually interfere. The mobiles have only one radio, and at any given time operate in a specific channel.

The radio limits transmission burst to 10 ms. At 625 kb/s, the maximum size of an air interface packet is 6250 bits. The radio provides a rated bit error rate of 1E-5. This translates into an ATM cell loss probability due to noise of less than 0.5 percent. While much larger than what is easily available on the wired backbone, this cell loss probability is overshadowed by frequency slot collision in collocated channels. For example, if two channels using 75 long frequency-hopping sequences collide even once every sequence, a loss of 4 percent occurs. This illustrates a fundamental capacity loss problem when frequency-hopping spread spectrum radios are used in a system with multiple collocated channels. Besides direct frequency slot collisions, one also has the problem of interference from adjacent frequency slots, and the adverse impact on frequency slot reuse of the high-level modulation needed to get increased bit rates. In general, techniques such as smart hopping and information spreading across hops are more crucial in frequency-hopping-based wireless links than techniques targeted at noise. For example, if two channels using 75 long frequency-hopping sequences collide even once every sequence, a loss of 4 percent occurs in the presence of just three transmitters.

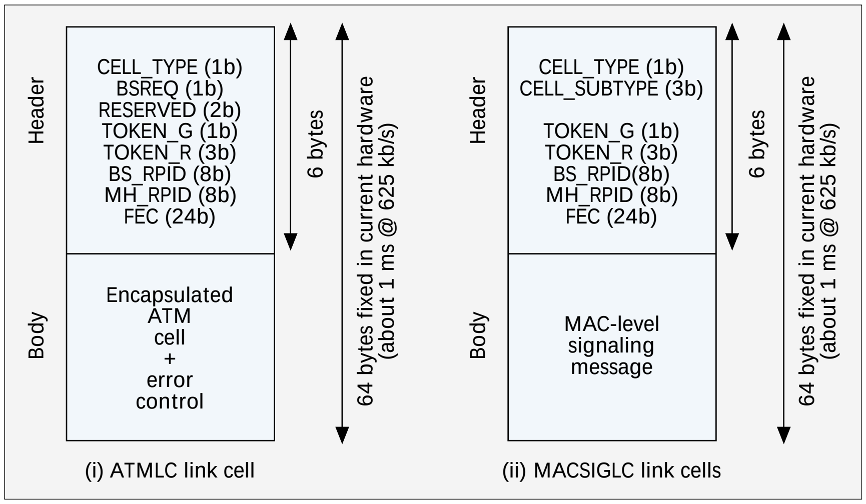


Figure 4. *Format of link cells (air interface packets).*

Mapping of ATM Cells to the Air-Interface Packets — *Air-interface packets*, or *link cells*, are the data units sent by the MAC layer to the air-interface control layer at the transmit end, and received by the MAC layer from the air-interface control layer at the receive end. The air-interface control layer directly controls the radio, and would typically be realized in hardware as an air-interface controller (or physical-layer controller). Digital radio cores in their simplest embeddable form offer serial bit-stream input and output, and the air-interface controller is expected to do parallel-to-serial conversion and channel coding at the transmitter, and clock recovery, bit framing, channel decoding, and serial-to-parallel conversion at the receiver. To take advantage of the readily available off-the-shelf serial communications controller chips that do these functions, the air-interface controller in SWAN incorporates a serial communications controller operating in a synchronous mode, resulting in the well-known synchronous data link control (SDLC) protocol being used over the air. In one transmit burst, transmitters send one or more SDLC frames separated by the SDLC SYNC bytes, with each SDLC frame consisting of one or more link cells.

There are several types of link cells. One type, called ATMLC, is defined to carry an encapsulated ATM cell. Several other link cell types are defined for MAC-level signaling. For example, the simple MAC protocol currently being used defines the following signaling link cells: CRLC, for connection request by a mobile that powers up; HRLC, for handoff request by a mobile; SYNCLC, for idle channel; and CHRCLACK1, CHRLACK2, and CHRLCACK3. for handshake during registration of a mobile at a base station. Collectively, the link cells for MAC-level signaling are referred to as MACSIGLCs. All link cells are composed of a fixed 6-byte header, and a body whose contents depend on the type of link cell. As shown in Fig. 4, the header has an 8-bit statically assigned base station radio port id (BS_RPID) field, an 8-bit dynamically assigned mobile host radio port id (MH_RPID) field, a 1-bit CELL_TYPE field indicating whether the link cell is of type ATMLC or not, 7 bits defined by the MAC protocol, and a 24-bit FEC field that uses an (8,4) linear code to forward error correct the preceding 24 bits. Obviously, in the case of a non-ATMLC link cell, the MAC protocol would use some of the seven reserved bits to disambiguate among the various signaling link cell subtypes. The current MAC protocol, as Fig. 4 shows, uses 3 bits for this purpose. The base station radio-port id BS_RPID is a logical id statically assigned at setup such that no two radio-ports in the radio vicinity have the same id. This logical radio-port id is mapped by the base station to the wired network address of the base station and the radio-port id within the base station. Similarly, the mobile host radio-port id MH_RPID is a logical id that is assigned to a mobile host by a base station radio port when the mobile registers at that base station radio port. MH_RPID is unique among the mobiles registered on the same base station radio port.

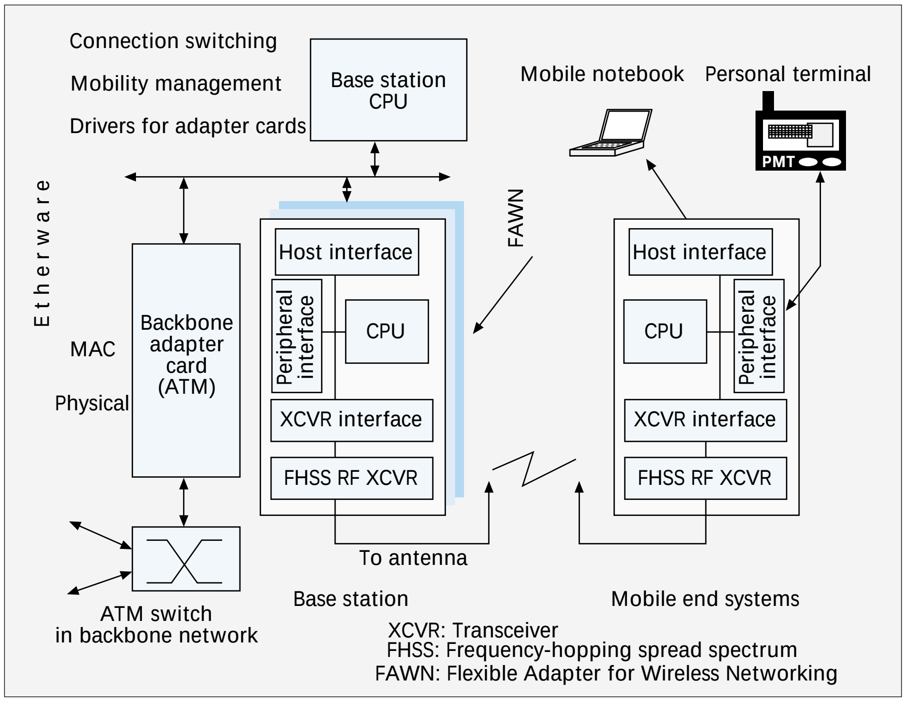
The link cell bodies in the case of MACSIGLCs are small and typically contain various address information. The ATMLC body encapsulates an ATM cell. Currently, the 53-byte ATM cell is stored as is, together with link-level error control information (2-byte CRC-16 for ATM cells in VCs using link-level retransmission). In the future, we plan to compress the ATM cell header by two mechanisms:

- Stripping the 8-bit ATM header error control (HEC) field from the transmitter and recalculating it at the receiver, in the presence of link-level error control
- Transparently mapping the 24-bit VCI/VPI fields to a smaller 8- or 16-bit VCID for the duration of the air transport. The latter mechanism has the implicit trade-off of allowing a smaller number of active VCs to a mobile host.

It must be noted that the first-phase implementation of the air-interface controller, described later, uses fixed 64-byte-sized link cells. This has the negative side effect that there are five wasted bytes (8 percent bandwidth loss) for each ATMLC when no link-level error control is used (three wasted bytes per ATMLC with link-level retransmission), and a much larger number of wasted bytes for signaling link cells with consequent higher MAC-level signaling overhead. The hardware is, however, reconfigurable, and a future version of the air-interface controller will remove this limitation.

Token-Passing MAC Protocol — The time between two frequency hops on a channel is called the *hop frame*, which is subdivided into several link cells. Access to the channel is regulated by a token-passing mechanism, with the base station acting as the master for handing out the token. The handoff is mobile-initiated. which transmits *handoff request link cells* (HRLCs) based on measurements of current base station power. On the other hand, the base station searches on its idle radio ports for mobiles that are seeking a base station.

The basic protocol for access regulation on a channel is that of token passing, with the base station acting as the central arbiter that decides who gets the token, and hence the transmission privilege. The token can be held for at most $N = 8$ link cell duration, which is slightly less than 10 ms, the length of maximum allowable continuous transmission burst. This maximum interval is used to detect lost tokens in noisy channels. The token information is a part of the link cell header, in the form of two fields: TOKEN_G (1 bit) and TOKEN_R (3 bits). The G field with value 1 is used to indicate that a token is being granted to the receiver (for $\leq N$ link

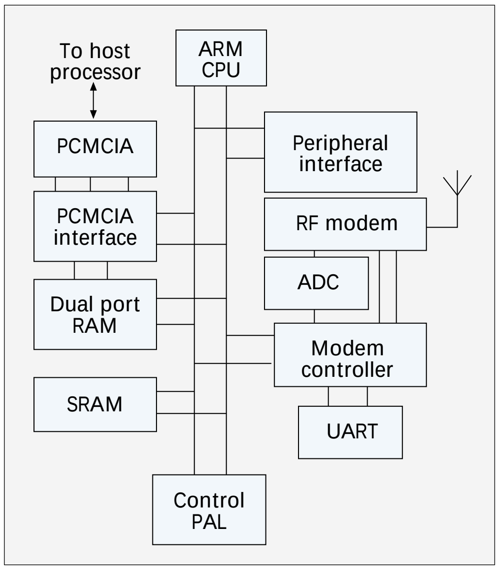


■ **Figure 5.** *Architecture of the SWAN system.*

cell duration). The R field indicates the number of ATM cells that are queued at the sender, information used by the base station in scheduling the token. In the case of an idle channel, the control will just pass back-and-forth with $G = 1$ and $R = 0$. Finally, in case the token is lost due to noise, the mobiles do nothing (and time-out) while the base station takes over the control and resets the token-passing protocol.

The token-passing approach to MAC allows the base station to act as a coordinator and scheduler of the available bandwidth within the cell. By taking into account QoS parameters of individual VCs when scheduling cells, the base station can ensure that the lower layers do not make null and void the higher ATM-level service quality assurances. Such would be the case, for example, with the carrier sense multiple access (CSMA) and CSMA with collision avoidance (CSMA/CA) based peer-to-peer MAC with inherently nondeterministic delays as found in the IEEE 802.11 standard, cellular digital packet data (CDPD), and various wireless LAN products.

Closely related to MAC is the control of frequency hopping. Instead of using a hopping scheme based on measuring hopping interval in terms of real time, the number of token passes are counted to measure the length of the hop frame. In particular, hopping is done after every $M = 8$ token grants from mobile to base station. Of course, token losses are a nuisance in this scheme, and a straightforward counting of tokens will not work. Therefore, the effective number of token passes are counted to decide when to hop. The *effective number of token passes* is the actual number of token passes plus the number of time-outs while waiting for the token.

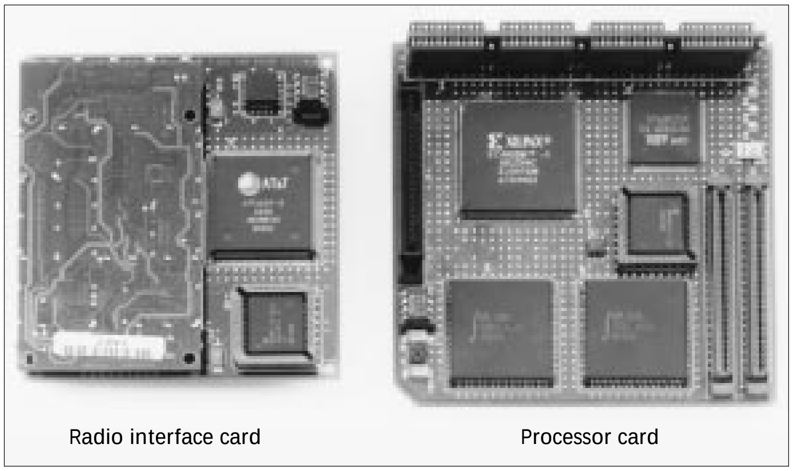


■ **Figure 6.** *Architecture of the FAWN wireless adapter.*

***Wireless Hop Scenarios* — Scenario 1: New Mobile Entering the Network —** After power-up a mobile begins to transmit a connection request link cell (CRLC). This is done using a random initial frequency-hopping sequence, and is at a fast rate whereby the mobile jumps to the next frequency slot in the sequence if no base station responds to the CRLC. The body of the CRLC consists of the globally unique mobile id and a hop sequence id. The hop sequence id defines the hopping pattern that had been assigned to the base station radio port at start-up, and the mobile uses this hop sequence id to later hop in synchrony with the base station radio port. The information in the CRLC body is protected by an FEC scheme based on an (8,4) linear code. Following CRLC is a reserved fixed time interval for an interested base station radio port to acknowledge via a CHRLCACK1 cell. Contained in CHRLCACK1 is an 8-bit logical id that the base station assigns to the mobile for the duration of the mobile's connection to the radio port. Following successful reception of CHRLCACK1 by the mobile, an exchange of CHRCLACK2 and CHRCLACK3 takes place to finish the three-phase handshake that constitutes the mobile registration process.

Scenario 2: Mobile Doing a Handoff — A mobile continually measures the RF power P_{current} of packets it receives from its base station. Two power thresholds are defined: P_{min} and P_{thresh} ($> P_{\text{min}}$). When P_{current} falls below P_{thresh} but is still above P_{min}, the mobile initiates the process of *soft handoff* by beginning to periodically transmit an HRLC with periodicity proportional to $P_{\text{thresh}} - P_{\text{current}}$. In addition, the mobile sets the "base station request" bit (BSREQ) in the header of all the link cells it transmits. This indicates to idle base stations as well as the current base station that a handoff is needed. The body of the HRLC consists of the globally unique mobile id, a hop sequence id, and the id of the current base station. As in CRLC, the body of HRLC is also protected by an (8,4) FEC linear code. The handshake that follows an HRLC is a three-phase handshake similar to that in the case of a CRLC as described above. Following the HRLC is a reserved fixed time interval for an interested base station to acknowledge via a CHRLCACK1 cell. The current base station radio port maintains radio silence during this reserved interval if it receives an HRLC from its mobile. If the power P_{current} falls below P_{min}, the mobile assumes that its connection to the current base station has been lost, and begins to continually transmit HRLCs and switches to a fast hop rate. The fast hopping rate not only reduces the effect of frequency collision with other channels, but also reduces the average time to find a new base station, thus helping in the goal of low handoff latency. Of course, soft handoff, described earlier, is the primary mechanism for low-latency handoffs.

Scenario 3: Base Station Activity at an Idle Radio Port — The base station with one or more idle ports actively hunts for mobiles that might want to connect. This is done according to the following process. First, using hints from the wired backbone, a frequency slot is chosen for the idle radio port such that none of the radio ports in the parent base station or on neighboring base stations are using that frequency slot. The idle radio port hops to the frequency slot thus chosen. Next, it measures power at that frequency and snoops for link cell headers. If no activity is detected at that frequency slot, a new frequency is chosen and the hunt is restarted. If activity is detected but link cell headers show that the BSREQ bit is not set, then the base station assumes that the mobile is not interested in a handoff, and it again restarts the hunt at a new frequency. Otherwise, the base station radio port waits for a CRLC or HRLC, or for the channel to become idle. If a CRLC or HRLC is received, the base station initiates the registration process for the powering-up mobile or for the handing-off mobile, as the case may be.



■ **Figure 7.** *Photographs of the FAWN radio interface card and FAWN processor card.*

Hardware and Software Implementation

The previous section presented a layered architectural overview of SWAN. This section describes the first-phase implementation of that architecture. Figure 5 shows the various hardware and software components in the implementation, with a particular focus on the entities in the wireless hop. The following subsections describe the various components in detail.

The FAWN Network Adapter

From a hardware perspective, the key idea behind SWAN's wireless last hop is the use of a single reusable ATM wireless adapter that interfaces to one or more digital-in digital-out radio transceivers on one side, and to one or more standard buses (Personal Computer Memory Card Industry Association —PCMCIA — or PC card bus, in our implementation) on the other side. The adapter, called FAWN for *Flexible Adapter for Wireless Networking* [7], provides reconfigurable data processing resources in the form of Field Programmable Gate Arrays (FPGAs) and a software-programmable embedded Acorn RISC Machines' ARM610 RISC processor. This flexible and reusable adapter card provides a uniform mechanism for making devices "SWAN-ready."

The FAWN card, whose architecture is shown in Fig. 6, has an embedded ARM610 processor that is intended to implement low-level ATM and MAC protocols, signaling, scheduling, and queue management software. A dual-port random access memory (RAM) provides a high-bandwidth communication link to the host. In addition, the host can also directly access memory mapped peripherals on the ARM bus at a slower speed via a cycle-stealing arbiter. A mezzanine connector on the processor card allows for one or more radio link interface cards to be connected. Each radio link interface card has an FPGA-based reconfigurable air-interface controller, a serial communication controller, an analog/digital (A/D) converter for signal power measurement, and the radio described previously. Various air-interface functions, such as packetization-depacketization and signal strength monitoring, are implemented on the FPGA. The serial controller interfaces to the air-interface controller on one side and to the serial data input and output of the radio on the other side, and provides SDLC-based communication over the radio link. A second mezzanine connector brings out a peripheral expansion bus which allows a peripheral card to be memory-mapped into the ARM address space.

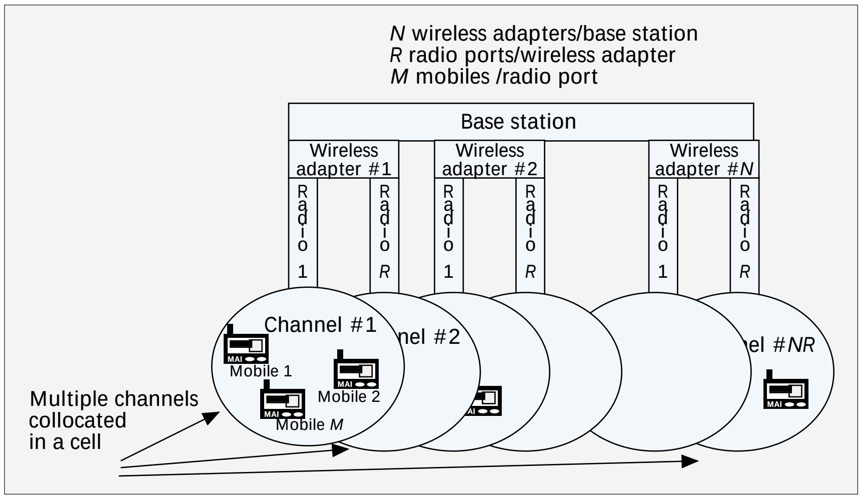
In the current implementation of the reconfigurable air-interface controller, the communication between the air-interface controller and the ARM processor is done via two sets of ping-pong link cell hardware buffers, one for transmit and another for receive. The link cells are assumed to be of 64-byte size, leading to some bandwidth wastage when transporting ATM cells, as discussed previously. A finite-state machine in the hardware does the depacketization into a byte stream necessary for the outgoing link cells in the ping-pong buffers to be sent to the serial controller, and the packetization of the incoming byte stream from the serial controller. The communication between the air-interface controller and the ARM processor is interrupt-driven, and the hardware buffers in essence serve to reduce the interrupt rate on the ARM to once per link cell (approximately 0.8 ms with the current fixed 64-byte link cells and the 625 kb/s radio).

The FAWN adapter emphasizes functional flexibility and reconfigurability, and its miniaturized physical dimensions enable portability. A FAWN card with one radio interface plugged in has dimensions of 10.8 cm (width) x 1.9 cm (height) x 11.4 cm (depth), and fits conveniently into the removable floppy drive bay of many laptop and desktop computers. The card, excluding the radio transceiver, consumes 2 W of power. The radio transceiver that is currently used (previously described) consumes another 0.6 W in the receive mode and 1.8 W in the transmit mode.

Figure 7 shows a photograph of the FAWN radio interface card and processor card.

Base Station Architecture

Figure 8 shows the high-level architecture of a typical base station in SWAN. A base station consists of multiple FAWN cards plugged into its backplane, with each card handling multiple radio transceivers. Each radio transceiver is assigned a channel (frequency-hopping sequence) that is different from channels assigned to a radio in the current or neighboring base stations. Typically in SWAN, a base station has fewer



■ **Figure 8.** *Abstract view of a base station in SWAN.*

than three to five radios per base station. The preceding base station organization results in a cellular structure where each cell is covered by multiple collocated channels. A mobile in a cell is assigned to one of the radio ports on the base station, and frequency hops in synchrony with it. While, from the hardware and software perspective, a base station can have an arbitrarily large bank of radios, there are limits imposed by the spectrum allocation and the type of radio. For example, as described previously, SWAN uses a family of 22 codes with its current radio, each of which can be viewed as a communication channel. Obviously, since the current radio is a frequency-hopping one and the codes correspond to frequency-hopping sequences, the channels are inherently mutually interfering due to frequency slot collision and adjacent frequency slot interference.

The current implementation uses PCs running Linux and Sun workstations running SunOS as the hardware platform for the base stations. In the case of PCs, the FAWN cards are interfaced to the CPU via Industry Standard Architecture (ISA)-to-PCMCIA bridge cards, while in the case of Suns SBUS-to-PCMCIA cards are used. Clearly, PCMCIA is a fairly slow 16-bit non-direct memory access (DMA) bus and is not adequate when multiple radio link interfaces are plugged into a single FAWN card. However, it is adequate for our initial experimentation where we use one radio per FAWN card, and has allowed us to rapidly construct a prototype system using the same FAWN card on the base stations and the mobiles. In future, special-version FAWN cards may be developed for the base stations by using higher-speed buses instead of the PCMCIA bus.

"Smart" and "Dumb" Mobile Hosts

SWAN has two types of mobile hosts, smart and dumb. Smart hosts, which have local general-purpose computation resources, are built by connecting FAWN cards to laptops via PCMCIA interface. More interesting perhaps are the dumb mobile hosts in SWAN. These are multimedia terminals [5] that are formed by connecting various peripherals to a FAWN card via the peripheral expansion bus on FAWN. No other processor for local computation is provided except for the ARM processor embedded on FAWN itself. These multimedia terminals, which are called personal multimedia terminals (PMTs), present a simple user interface (Fig. 9). On the front of a PMT is a 320 x 240 dot-matrix LCD display along with three control buttons. A set of earphones with an attached directional microphone plug into the side of the PMT to provide bidirectional audio capability. Another side of the PMT has a bar code scanner, controlled by a dedicated button. The bar code scanner can be used in applications such as inventory database access in a warehouse, patient record access by nurses and doctors in a hospital, and product information access by shoppers in a department store.

All the circuitry necessary for PMT resides on a peripheral card that plugs into the peripheral expansion connector on FAWN. In addition to the 2 W consumed by the embedded FAWN card, and the 0.6 W/1.8 W consumed by the radio (in receive/transmit mode), the PMT consumes 0.57 W, for a total of 3.7 W/4.9 W in the receive and transmit modes, respectively.

Unlike PDAs and laptops, and somewhat similar to Xerox PARC's Tab terminal [27], Berkeley's Infopad [6, and Zenith's Cruisepad [28], a PMT does minimal local computing. Its functionality is entirely determined by a server on SWAN's wired network. The functionality, features, and services provided by PMT are context-dependent, defined by the server with which it is communicating.

Etherware Network and System Software Implementation

The Etherware software, which implements the layered system architecture outlined previously, is shown in Fig. 10 from the point of view of a base station. A similar implementation exists at the mobile hosts. The figure also shows the correspondence between the actual software modules and the architecture functions in Fig. 2. The key software modules at a base station are:

- CM, a user-space-resident module, corresponding to the ATM Connection Management function in Fig. 2
- DT, a partially user-space- (DTuser) and partially kernel-space-driver- (DTkernel) resident module, corresponding to the ATM Data Transport and Switching function in Fig. 2
- MAC, a FAWN-adapter-resident embedded ARM software module, encompassing the cell scheduling, per-VC error control, MAC, and parts of the air-interface control functions in Fig. 2

In addition, there is an API library that is linked into the applications. Corresponding modules also exist at the mobile hosts, except that the algorithms used are different, being tailored for mobile host functions. The API library and CM modules also exist at wired ATM hosts. Mobility-aware switches in the wired backbone will need to have functionality equivalent to CM. At the present time we have not made any of the switches, except for the base station switches, in our wired ATM backbone network mobility-aware.

It is clear that much of Etherware is currently implemented via user mode processes under SunOS and Linux instead of kernel code. This was done to rapidly prototype an initial implementation so that we can experiment with the algorithms, particularly for the logically complex signaling protocols. Of course, the ease of implementation came with an often considerable performance cost due to increased process context switch overhead and copying of data buffers across the user and kernel address spaces.

The DT module primarily switches ATM cells among the various wired and wireless ATM adapters on the base stations; the CM module participates in ATM-level signaling to establish, tear down, and reroute ATM VCs; and the MAC module allocates and schedules wireless resources in response to signaling messages from the CM module, and implements low-level MAC and intercell handoffs by the mobile endpoints. PCs and workstations used as base stations act as wired hosts as well, and run application processes in addition to the three modules mentioned above. In essence, base stations in SWAN are computers equipped with banks of radios. The software view at mobile endpoints is similar, except that a mobile endpoint has only one ATM adapter (FAWN), and the CM, DT, and MAC modules at the mobile have somewhat different functionalities.

The MAC module maintains a table of per-virtual-connection information to help schedule the wireless resources among the multiple ATM VCs going over a wireless channel. When a new VC needs to be opened, the CM module sends a request to the MAC module indicating the bandwidth requirements as the channel time $T1$ needed by this VC over a period of time $T2$. The MAC module uses this information to either accept or deny admission to this new VC. This bandwidth specification is also used by the MAC module to schedule transmission of cells.

The ATM signaling among the CM modules at various nodes is done on a reserved VC, and the signaling between the CM module and the MAC module at a node is done on a separate reserved VC. The CM module, the DT module, and the applications communicate among themselves using standard SunOs and Linux interprocess communication (IPC) primitives (shared memory segments and pipes). In the future, we expect to migrate all DT functionality into the kernel code, at which time CM and applications will access DT by opening special device files.

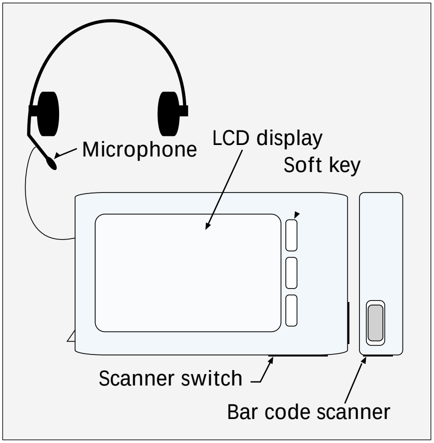


Figure 9. *PMT: dumb multimedia terminal in SWAN [5].*

Support for TCP and UDP Applications

The three modules, CM, DT, and MAC, together provide the basic wireless and mobile ATM functionality. We had also set an early milestone for SWAN to provide IP connectivity because native-mode ATM applications are still rare, and a complete implementation of mobile and wireless ATM is a long-term goal. On the other hand, applications using IP-based internet protocols such as TCP and UDP are readily available, and there already exists much previous experience with IP in a wireless and mobile context (e.g., the WaveLAN product from AT&T).

IP connectivity is provided in SWAN by using IP/ATM segmentation-and-reassembly modules at the mobiles and base stations, together with IP forwarding capabilities of the Linux operating system on the PC-based base stations, and a reserved VC over the wireless link for carrying IP traffic. The segmentation-and-reassembly module is simply a network interface driver that has been added to the IP protocol stack, and treats the reserved VC as a network link with 48-byte frames. The IP forwarding capability of Linux allows the base station to act as an IP router, forwarding IP packets between the reserved VC on the wireless link and other IP network interfaces. In effect, SWAN provides IP connectivity by using IP-over-wireless-ATM. With this approach IP-level mobility is a simple matter of using the mobile-IP implementation already available for Linux [29]. Using this IP-over-ATM connectivity, mobile hosts within SWAN have immediate access to all the IP-based applications, including those requiring connections to machines outside the SWAN network.

An alternative approach, which we have not yet explored, to provide IP connectivity in a SWAN-like network would be to provide IP over an end-to-end ATM VC that is set up to carry the IP traffic between any two SWAN nodes (wired or wireless). Only segmentation and reassembly would be needed at the two ends, and mobility would come for free because of the underlying mobile and wireless ATM. While more complex to implement, such an approach would be more elegant with only a single model of mobility.

Runtime Software Environment

A runtime environment, available under Linux and SunOS, provides support for code development and control of FAWN cards. The control interface is implemented as two special files, /dev/fawn/mem and /dev/fawn/ctl in our systems. The former allows host-resident processes to read and write memory and registers in the address space of the ARM610 processor that is embedded on FAWN. The latter allows FAWN control registers to be manipulated by writing string commands, and the FAWN status registers to be read by reading a string. For example, the commands:

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

when issued on a mobile host or a base station by a suitably privileged user cause the FAWN card to be reset, the binary

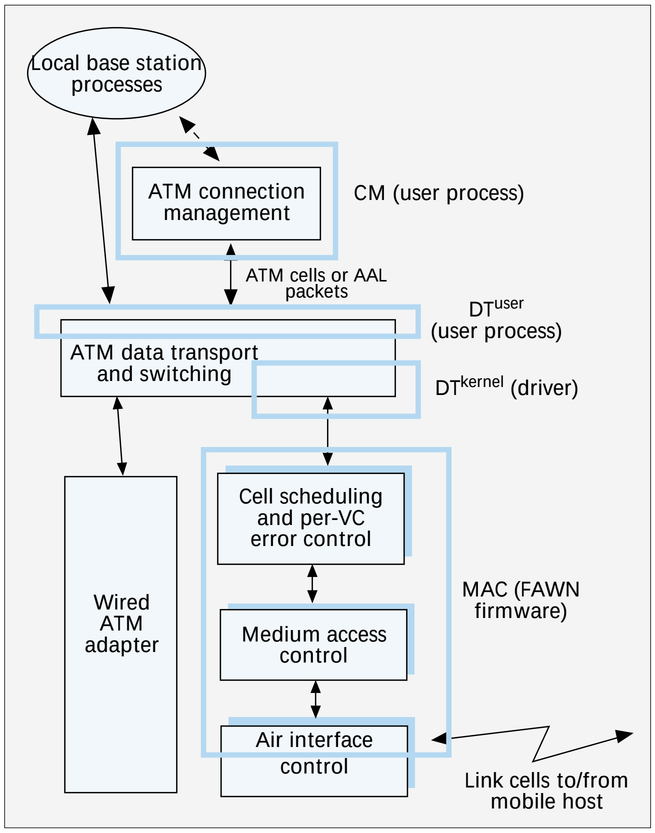


Figure 10. *Etherware software implementation at the basestation.*

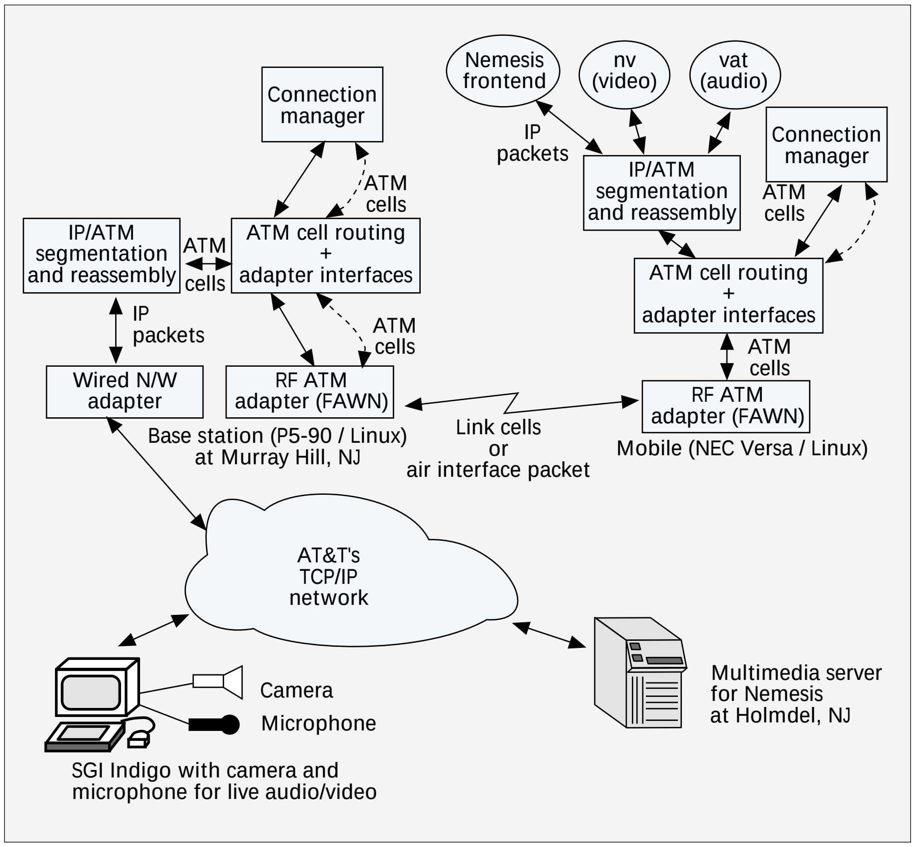

Nemesis frontend
nv (video)
vat (audio)
Connection manager
IP packets
IP/ATM segmentation and reassembly
ATM cells
ATM cell routing + adapter interfaces
IP/ATM segmentation and reassembly
ATM cells
Connection manager
ATM cells
ATM cell routing + adapter interfaces
IP packets
ATM cells
ATM cells
Wired N/W adapter
RF ATM adapter (FAWN)
RF ATM adapter (FAWN)
Base station (P5-90 / Linux) at Murray Hill, NJ
Link cells or air interface packet
Mobile (NEC Versa / Linux)
AT&T's TCP/IP network
Camera
Microphone
Multimedia server for Nemesis at Holmdel, NJ
SGI Indigo with camera and microphone for live audio/video

Figure 11. *Using SWAN for transmission of multimedia streams.*

image k.i to be loaded for execution by the embedded ARM610 processor, and the execution to begin.

Audio and Video Transmission over SWAN

The SWAN system is still under development, and native-mode ATM applications for SWAN together with signaling and MAC-layer support for ATM QoS guarantees are being developed. Experiments with ATM have so far been restricted to the performance measurement utility netperf, and a version of the X11 video conferencing tool nv that we have ported to our ATM API. Pending the availability of more native-mode ATM applications for SWAN, we have used conventional TCP/IP-based multimedia (and other) applications such as nv, vat, and xmosaic as the initial drivers of SWAN. This is accomplished by using the IP-over-ATM approach described previously.

We have experimented with several scenarios of multimedia information transmission using the IP-over-ATM mechanism. Figure 11 depicts some of the possibilities. In one such application the mobile user sees and hears a live video and audio feed transmitted from a camera- and microphone-equipped SGI Indigo. The popular Internet audio and video conferencing tools, *vat* and *nv*, are used in a host-to-host mode. The live feed can also be replaced by prerecorded video streams, such as movie and cartoon clips, transmitted by *nv_play* from a remote host. In a related experiment, a mobile user can use vat alone to conduct a voice conversation with another mobile user, or with a user on a wired host. This is a nascent step toward our eventual goal of fully subsuming a multimedia PBX-like functionality in SWAN. A final application scenario with which we have experimented is to let a mobile user play, in real time, video and audio streams being fetched using the *Nemesis* [30] multimedia service from a multimedia archives server containing audio, Motion Picture Experts Group (MPEG) video, and accompanying documents and viewgraphs of talks given at AT&T. Nemesis uses adaptive rate control so that a reasonable video quality is obtained at the mobile over the wireless link. The Joint Photographic Experts Group (JPEG) decompression is done in software at the mobile.

To conduct the above experiments with vat, nv, and Nemesis, we used 486-based PC laptops running Linux as mobile hosts. The NEC VERSA and AT&T Safari that we used have built-in audio input/output (I/O), and had FAWN adapters plugged in via their PCMCIA ports. The IP-over-ATM experiments were done with a MAC that divided the channel bandwidth (nominally 625 kb/s) equally in the receive and transmit directions, and as the following section shows, the reliable TCP throughput measured in these experiments was about 227 kb/s in each direction. With such bandwidth, and software decompression at the mobile CPU, we got 10–15 frames/s with Nemesis.

Preliminary Performance Measurements

The performance measurement experiments that we have conducted so far fall into two categories. Initially, we focused on understanding the characteristics of the wireless hop and the effect of various low-level design choices. For this we used IP connectivity with a simplified MAC which gives the token to the base station radio port and the mobile for *N* link cells each repeatedly. In effect, this makes each channel a time-division duplex (TDD) channel, with a frame length of *N*. Furthermore, since only IP traffic is being carried in these experiments, the MAC uses a nonstandard ATM cell with 60 bytes of data body. This gives us maximum performance by eliminating the wasted bandwidth discussed previously, due to the fixed 64-byte link cells that are currently supported by FAWN's air-interface controller. We refer to these experiments as using IP/ATM-60 connectivity. In a second set of ongoing experiments, we are mea-

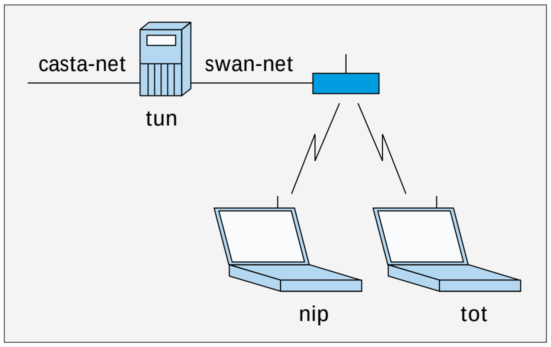
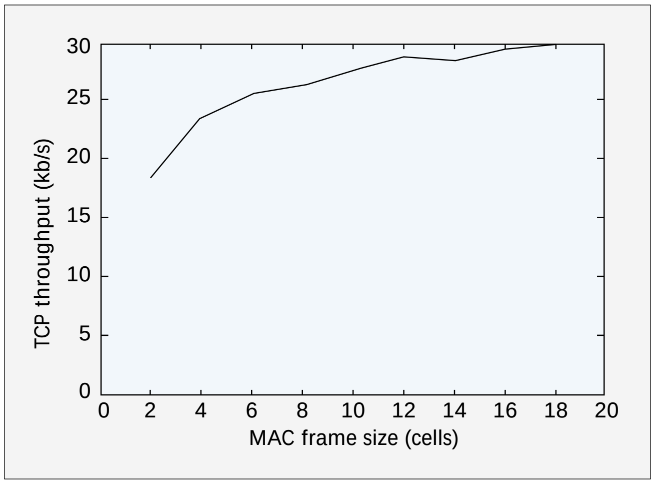
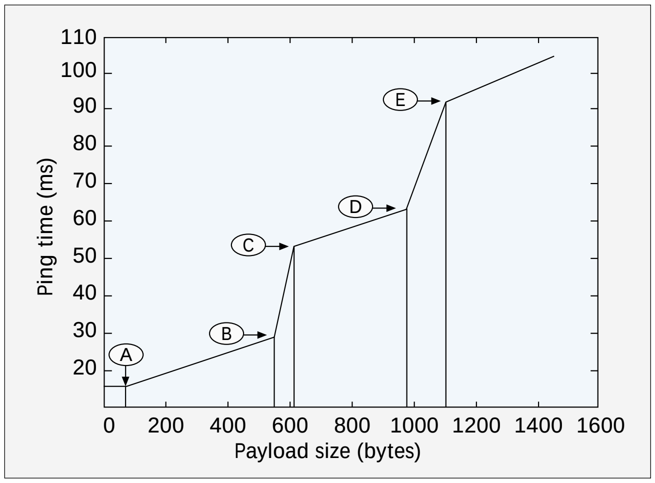


Figure 12. *Network configuration for performance measurements with IP/ATM-60.*



■ **Figure 13.** *TCP throughput vs. MAC frame size.*



■ **Figure 14.** *Ping time in milliseconds vs. payload size in bytes.*

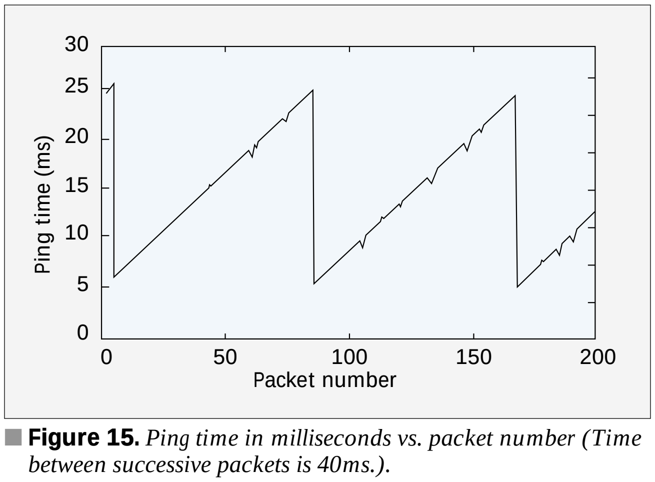
suring the performance of end-to-end ATM connectivity. This, of course, uses the standard 53-byte ATM cell with 48-byte body over the wireless link as well.

Experiments with IP/ATM-60 Connectivity

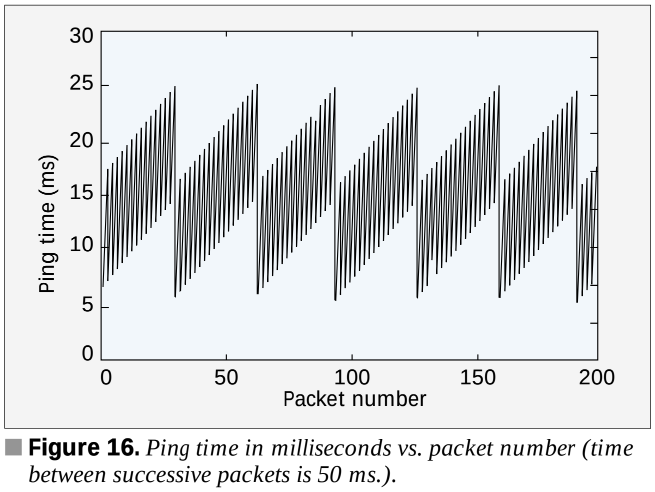
Figure 12 shows the topology of the network used during experimentation. *Tun* is a 90 MHz Pentium/PCI PC, configured as a gateway between *swan-net* and *casta-net* (the department Ethernet); *nip* and *tot* are, respectively, a NEC VERSA 100/4 and an AT&T Safari 3181 notebook. In the results that follow, TCP throughput was measured using *ttcp (*available as ftp://ftp.sgi.com/sgi/src/ttcp/) to transfer a 1 Mbyte file from tun to nip over the wireless link, with no other traffic on the link. The default buffering in ttcp was used. The antennas were placed sufficiently close to each other that the error rate on the channel was minimal. A nearby spectrum analyzer showed that no abnormal interference was present at the time of the experiments. Using the standard MAC frame size of 10 cells for both transmit and receive frames, the test file was transferred in 36.97 s, yielding a TCP transfer rate of 28.36 kbytes/s (227 kb/s).

Effect of MAC Frame Size on TCP Throughput — MAC frame size, which is the size of one transmission burst between two communicating MAC-level entities, is an important parameter over which the system designers have significant control. Figure 13 shows the effect that varying MAC frame size has on TCP transfer rate. This figure reveals that small frame sizes give rise to low transfer rates. This is to be expected because it takes some time to turn the wireless link from transmit mode to receive mode and vice versa, due to overheads such as link synchronization. Small frame sizes do this turnaround more frequently, yielding lower data rates. The graph shows that a choice of 10 cells/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.

Effect of Transfer Size on Round-Trip Delay — Figure 14 shows a plot of round-trip delays on the wireless link, measured by ping time, for a range of payload sizes. The times are the average times calculated from 10,000 individual samples for each payload size measured. Payload sizes were measured in 60-byte increments, with 60 bytes being the length of data body used in the air-interface packets for these IP/ATM-60 experiments. Payload sizes smaller than A fit into a single cell and result 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 maximum transmission unit (MTU) specified to the IP layer is 1024 bytes, payload size at this stage is increased by one or two cells due to IP fragmentation.



■ **Figure 15.** *Ping time in milliseconds vs. packet number (Time between successive packets is 40ms.).*



■ **Figure 16.** *Ping time in milliseconds vs. packet number (time between successive packets is 50 ms.).*

Raw radio link bandwidth	312 Kb/s each way
MAC level total throughput	280 Kb/s each way
MAC level user data throughput	210 Kb/s each way
End-to-end ATM user data throughput	190 Kb/s each way

■ **Table 1.** *ATM throughput in SWAN.*

Effect of Packet Arrival Times on Delay Jitter — The MAC frame structure induces noticeable effects in end-to-end delay jitter. Figures 15 and 16 show the variation 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 Fig. 15, the smallest ping time is observed when both 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 synchronized 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 Fig. 16, 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.

Discussion of IP/ATM-60 Results — Our results highlight some interesting challenges. First, a TCP transfer rate of 28.36 kbytes/s shows that we are utilizing about 227 kb/s or 73 percent of our 312 kb/s bandwidth in a single direction. Second, the wireless channel is implemented within a shared medium and we have less control over it than we would have over a wired, switch-based network, so the provision of any QoS guarantees typically associated with ATM networks is made much more difficult. The ping round-trip delay times plotted in Figs. 15 and 16 are good examples of the challenges faced. The half of round-trip 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 achievable delays and delay jitter.

Experiments with ATM Connectivity

We are currently conducting experiments with ATM connectivity in SWAN. The two applications that we have used are the *netperf* performance measurement tool and the *nv* video conferencing tool. We modified both these applications to work with our ATM API library. Using the implementation of Etherware software described previously, we have obtained the preliminary ATM throughput numbers in the presence of no handoffs listed in Table 1.

These numbers show the effect of certain features of FAWN as well as the inefficiencies of the user space implementation of the Etherware software. The discrepancy between the 280 kb/s MAC-level total throughput and the 312 kb/s raw radio link bandwidth is due to the SDLC overhead as well as the fact that the current air-interface controller implementation does not allow full pipelining of the double link cell buffers so that an interrupt handler context switch worth of delay is inserted between two successive link cells. The 210 kb/s MAC-level user data throughput is simply a result of the fact that only 48 out of 64 bytes of each ATMLC are being used for user data. The remaining 16 bytes are used for link cell header, ATM header, and wasted due to the fixed link cell size. The bandwidth of 210 kb/s corresponds to (48/64)*280 kb/s. Since we had no handoffs, there was no loss of throughput due to signaling overheads. The cell loss due to noise was also quite negligible, with typically less than 0.25 percent cell loss rate observed.

The 190 kb/s end-to-end ATM user data throughput was measured using netperf for a VC that went from a Sun workstation, across a Fore ATM switch in the wired backbone, to a SWAN base station, and then over the wireless link to the mobile host. The 190 kb/s measurement is the throughput as seen by the receiver with no transport-layer retransmission being done by the sender. As is obvious, we are currently unable to drive the wireless link to its full MAC-level ATM user data capability of around 210 kb/s. This is a result of the user space implementation of Etherware, in particular the DT^{user} module in Fig. 10. ATM cells from the wired network have to traverse from the Fore driver to DT^{user} (which uses the null ATM adaptation layer — AAL0 — to read and write ATM cells from the wired ATM adapter). DT^{user} then sends the cells to the FAWN card. A reverse sequence of action takes place on the receive path. Clearly, the user space process DT^{user} is in the ATM cell data path between the wired network and the wireless link. This results in overheads due to context switches and extra kernel-user space data copying. Furthermore, a bug in Fore's ATM driver sometimes prevents us from transporting ATM cells in blocks of multiple cells in the AAL0 mode when cells are being both sent and received. This magnifies the inefficiencies of user space implementation. As a result, the DT^{user} module is unable to send or receive data from FAWN at full bandwidth, resulting in about 10 percent bandwidth loss with just one VC. The loss is greater with more VCs because of increased overheads in DT^{user}. We are currently migrating the DT^{user} functionality into a kernel–resident driver, so it will no longer be the bottleneck.

Summary

The SWAN project is exploring the seamless delivery of multimedia information to mobile users roaming in an indoor setting. This is being accomplished by a synergistic exploitation of wireless access and QoS-specifiable ATM virtual circuits. The first-generation system is largely functional. Using IP delivery over ATM, we already have initial experience with live as well as stored video and audio transmission using nv and vat, and also with access to a multimedia archives server via Nemesis. We are currently implementing native-mode ATM applications for SWAN, and characterizing the end-to-end ATM performance in the presence of handoffs.

Acknowledgments

We would like to acknowledge the help and contributions of Abhaya Asthana, Mark Cravatts, and B. Narendran in the SWAN project. Abhaya is the person behind the PMT multimedia terminal in SWAN. Mark helped in the hardware implementation of PMT as well as FAWN. Narendran is working on the directory and location servers for SWAN.

References

[1] B. Tuch, "Development of WaveLAN, an ISM Band Wireless LAN," *AT&T Tech. J.*, July/Aug. 1993, pp. 27–37.
[2] D. C. Cox, "Wireless Personal Communications: What Is It?" *IEEE Pers. Commun.*, April 1995, pp. 20–35.
[3] R. A. Schaffer, "High-Level Computing (Mobile Technology)," *Forbes*, vol. 156, no. 8, Oct. 9, 1995, p. 116.
[4] A. P. Chandrakasan and R. W. Brodersen, *Low Power Digital CMOS Design*, Kluwer Academic, 1995.
[5] A. Asthana, M. Cravatts, and P. Krzyzanowski, "An Indoor Wireless Sys-

tem for Personalized Shopping Assistance," *Proc. IEEE Workshop on Mobile Comp. Sys. and Applications*, Dec. 1994.
[6] B. Barringer *et. al.*, "Infopad: A System Design for Portable Multimedia Access," *Wireless 1994*, Calgary, Canada, July 1994, pp. 382–96.
[7] J. Trotter and M. Cravatts, "A Wireless Adapter Architecture for Mobile Computing," *Proc. 2nd USENIX Symp. on Mobile and Location-Independent Comp.*, Apr. 1994, pp. 25–31.
[8] Condon *et al.*, "Rednet: A Wireless ATM Local Area Network Using Infrared Links," *Proc. First Int'l. Conf. on Mobile Comp. and Networking*, Nov. 1995.
[9] K. Y. Eng *et al.*, "BAHAMA: A Broadband Ad-Hoc Wireless ATM Local Area Network," *Proc. 1995 IEEE Int'l. Conf. on Commun. (ICC '95)*, June 1995, pp. 1216–23.
[10] L. French and D. Raychaudhuri, "The WATMnet System: Rationale, Architecture, and Implementation," *Proc. IEEE Comp. Commun. Workshop*, Sept. 18–20, 1995.
[11] J. Porter and A. Hopper, "An Overview of the ORL Wireless ATM System," *IEEE ATM Workshop*, Washington, DC, Sept. 1995.
[12] P. P. Mishra and M. B. Srivastava, "Call Establishment and Rerouting in Mobile Computing Networks," private communication, Sept. 1994.
[13] M. B. Srivastava, "Medium Access Control and Air-Interface Subsystem for an Indoor Wireless ATM Network," *Proc. Ninth Int'l. Conf. on VLSI Design*, Bangalore, India, Jan. 1996.
[14] P. Agrawal *et al.*, "A Testbed for Mobile Networked Computing," *Proc. ICC '95*, June 1995, pp. 410–16.
[15] J. Ioannidis *et al.*, "IP-Based Protocols for Mobile Internetworking," *Proc. ACM SIGCOMM '91 Conf.*, Sept. 1991.
[16] F. Teraoka et al., "A Network Architecture Providing Host Migration Transparency," *Proc. ACM SIGCOMM '91 Conf.*, Sept. 1991.
[17] D. Raychaudhuri and N. D. Wilson, "ATM-Based Transport Architecture for Multiservices Wireless Personal Communication Networks," *IEEE JSAC*, vol. 12, no. 8, Oct. 1994, pp. 1401–14.
[18] B. Rajagopalan, "Mobility Management in Integrated Wireless-ATM Networks," *Proc. First Int'l. Conf. on Mobile Computing and Networking (MobiCom '95)*, Nov. 1995, pp. 127–35.
[19] K. Keeton *et al.*, "Providing Connection-Oriented Services to Mobile Hosts," *Proc. USENIX Symp. on Mobile and Location-Independent Comp.*, Cambridge, MA, Aug. 1993, pp. 83–102.
[20] A. Acampora and M. Naghshineh, "An Architecture and Methodology for Mobile-Executed Handoff in Cellular ATM networks," *IEEE JSAC*, vol. 12, no. 8, Oct. 1994, pp. 1365–75.
[21] S. Biswas and A. Hopper, "A Connection Management Scheme for a Mobile Radio LAN," *Proc. IEEE Int'l. Conf. on Personal Wireless Commun.*, Bangalore, India, Aug. 1994.
[22] C-K Toh, "The Design and Implementation of a Hybrid Handover Protocol for Multi-Media Wireless LANs," *Proc. MobiCom '95*, Nov. 1995, pp. 49–61.
[23] D. P. Chandler *et al.*, "An ATM-CDMA Air Interface for Mobile Personal Communications," *Proc. PIMRC '94*, 1994.
[24] J. Crowcroft *et al.*, "A Rough Comparison of the IETF and ATM Service Models." *IEEE Network*, Nov./Dec. 1995, pp. 12–16.
[25] L. Zhang *et al.*, "Resource ReSerVation Protocol (RSVP)," *IEEE Network*, Sept. 1993.
[26] R. Caceres and L. Iftode, "The Effects of Mobility on Reliable Transport Protocols," *Int'l. Conf. on Distributed Comp. Sys.*, Cracow, Poland, June 1994.
[27] M. Weiser, "Some Computer Science Issues in Ubiquitous Computing," *Commun. ACM*, vol. 36, no. 7, July 1993, pp. 75–85.
[28] Zenith Data Systems Cruisepad Product Introduction. "Cruisepad Is Not a PDA," *LAN Mag.*, vol. 10, no. 2, Feb. 1995, p. 20.
[29] V. Gupta and B. Lancki, "Mobile-IP for Linux," available from the CS Department, State University of New York, Binghamton, NY, at ftp://anchor.cs.binghamton.edu/pub/Linux-MobileIP/Linux-MobileIP.tar.gz.
[30] H. P. Katseff and B. S. Robinson, "Predictive Prefetch in the Nemesis Multimedia Information Service," *Proc. ACM Multimedia 1994*, San Francisco, CA, Oct. 1994.

The SWAN project is exploring the seamless delivery of multimedia information to mobile users roaming in an indoor setting through a synergistic exploitation of wireless access and QoS-specifiable ATM virtual circuits. The first-generation system is largely functional.

Biographies

PRATHIMA AGRAWAL [S '74, M '77, SM '86, F '89] heads the Networked Computing Research Department at Bell Laboratories in Murray Hill, New Jersey. She presently leads the Seamless Wireless ATM Network (SWAN) research team. Her previous positions include Distinguished Member of Technical Staff and supervisor of AT&T's Microprocessor Design Methodology Group. She led the the MARS and PACE multiprocessor architecture and applications projects. Her research interests are computer networks, parallel architectures and algorithms, and VLSI CAD (simulation and test.) She currently serves on the editorial boards of the *Journal of Parallel and Distributed Computing* and the *International Journal of Wireless Personal Communications*. She was the Program Chair for the 1987 IEEE International Conference on Computer Design (ICCD '87) and the General Chair for ICCD '88. Dr. Agrawal holds B.E. and M.E. degrees in electrical communication engineering from the Indian Institute of Science, Bangalore, India, and a Ph.D. degree in electrical engineering from the University of Southern California.

EOIN HYDEN [S '81, M '82] received the B.Sc., B.E., and M.Eng.Sc. degrees from the University of Queensland, Australia, and the Ph.D. from the University of Cambridge Computer Laboratory in England. While with the Systems Research Group in the Computer Laboratory, he worked on operating systems, high-speed networks, and multimedia systems. Currently, he is a member of technical staff in the Networked Computing Research Department at AT&T Bell Laboratories, Murray Hill, New Jersey. His interests include operating systems, mobile computing, and multimedia systems.

PAUL KRZYZANOWSKI received the B.E. degree in electrical engineering in computer science from the Cooper Union in 1985, the B.E. degree in computer science from New York University in 1985, and the M.S. degree from Columbia University in 1987. He joined the UNIX System laboratory at AT&T Bell Laboratories in 1985 where he worked on distributed file systems. In recent years, he has been working on mobile ATM in the Networked Computing Research Department at Bell Labs. His research interests are in computer architecture, operating systems, and mobility.

PARTHO MISHRA [M '95] received the B.Tech. degree from the Indian Institute of Technology, Kharagpur, in 1988, and the M.S. and Ph.D. degrees from the University of Maryland, in 1991 and 1993, all in computer science. He is currently a vember of technical staff in the Networked Computing Research Department at AT&T Bell Laboratories in Murray Hill. His research interests include ATM traffic management, mobile networking, and packet video services. His e-mail address is partho@research.att.com

MANI SRIVASTAVA [S '85, M '92] is a member of technical staff in the Networked Computing Research Department at AT&T Bell Laboratories, Murray Hill. Prior to joining Bell Labs in 1992, Mani received his Ph.D. and M.S. from U.C. Berkeley and B. Tech. from the Indian Institute of Technology, Kanpur. His research interests are in various aspects of mobile and multimedia information systems, including networking technology, endpoint architectures, digital signal processing, power management and optimization, and computer-aided design techniques.

JOHN TROTTER [M '90] received his B.Sc. degree from Brunel University, U.K., in 1986 and his D.Phil. from Oxford University, U.K., in 1990, both in electronic engineering. He then joined the Networked Computing Research Department at AT&T Bell Laboratories, Murray Hill, as a member of technical staff. His research interests are in ubiquitous information access and include wireless networked computing, mobile computing, and mobile information access. He is also interested in fault tolerance in distributed computer systems.