Open Programmable Mobile Networks

Oguz Angin, Andrew T. Campbell, Michael E. Kounavis, and Raymond R.-F. Liao

COMET Group, Center for Telecommunications Research,
Columbia University, New York NY 10027
`http://comet.columbia.edu/wireless`

Abstract

We present the design, implementation and evaluation of an open programmable mobile network based on distributed object technology called mobiware that dynamically exploits the intrinsic scalable properties of adaptive mobile applications. While a number of adaptive mobile systems have been proposed in the literature few experimental systems exist today to assess the viability of the adaptive approach. We argue that existing mobile systems (e.g., mobile IP, mobile ATM and third generation cellular systems) lack the intrinsic architectural flexibility to deal with the complexity of supporting adaptive mobile applications in wireless and mobile environments. We believe that there is a need to develop alternative network architectures from the existing ones to deal with the tremendous demands placed on underlying mobile signaling, adaptation management and wireless transport middleware. Mobiware represents one example of such an alternative approach. Software-intensive (comet.columbia.edu/wireless/software/mobiware), mobiware provides an open programmable approach to dealing with the complexity of supporting adaptive mobile applications in time-varying mobile and wireless networks.

1. Introduction

The phenomenal growth in cellular telephony over the past several years has demonstrated the value people place on mobile voice communications. The goal of next-generation wireless systems is to enable mobile users to access, manipulate and distribute voice, video and data anywhere anytime. As the demand for mobile multimedia services grows, high-speed wireless extensions to existing broadband and Internet technologies will be required to support the seamless delivery of voice, video and data to mobile terminals with sustained high quality. New wireless services will include Internet access to interactive multimedia, video conferencing and real-time data as well as traditional services such as voice, email and web access. The wireless and mobile environment presents a number of technical challenges to this vision. First, physical layer impairments contribute toward time-varying error characteristics and time-varying channel capacity as observed by mobile devices. We describe the quality index maintained across the wireless channel as *wireless-QOS* as illustrated in Figure 3. Second, user mobility can trigger rapid degradation in the quality of the delivered signal. This can lead to handoff dropping in broadband cellular networks and loss of service in mobile ad hoc networks. As a result, mobile applications can experience unwarranted delays, packet losses or loss of service. We describe the quality index maintained during mobility as *mobile-QOS* as illustrated in Figure 3. There is growing consensus that adaptive techniques [Katz,94], [Lee,95] [Lu,97] and [Naghshineh,97] present the only viable approach to countering time varying quality of service impairments in wireless and mobile network environments. However, providing system-wide (i.e., end-system and network) adaptive quality of service support is complex to realize and not well understood [Campbell,97b].

To address these challenges, we have built an open [Lazar,97] and active [Tennenhouse,97] programmable mobile network [Campbell,96] that is controlled by a software middleware platform called *mobiware* [Mobiware,98]. Mobiware extends earlier work by the COMET group on the xbind broadband kernel [xbind,96] to the mobile and wireless domain. By open, we mean that there is a need to "open up" hardware devices (e.g., mobile devices, access points and mobile-capable switches and routers) for implementation of new mobile signaling, transport and adaptive quality of service management algorithms. At the lowest level of programmability, mobiware abstracts hardware devices and represents them as distributed computing objects. These objects (e.g., an access point object) can be programmed via a set of open programmable network interfaces to support adaptive quality of service assurances. By programmable, we mean that these programmable network interfaces are high-level enough to allow new services to be built using distributed object computing technology, e.g., CORBA, DCOM and Java. By active, we mean that adaptive quality of service algorithms can be represented as active transport objects and injected on-the-fly into mobile devices, access points and mobile capable network switches/routers to provide value-added quality of service support when and where needed.

In this paper, we present an overview of mobiware followed by the design, implementation and evaluation of a programmable mobile network. The structure of the paper is as follows. Section 2 describes an adaptive-QOS API and service model, the mobiware architecture and the network model. Following this, Section 3 presents the design and implementation details of mobiware's programmable mobile network. This is followed in Section 4 by an evaluation of the system in an experimental setting. Finally, in Section 5 we present some concluding remarks.

2. Mobiware

Mobile applications need to be capable of responding to time-varying wireless-QOS and mobile-QOS conditions. Wireless transport systems should be capable of manipulating content in response to fluctuating quality of service conditions. Medium access controllers must be capable of supporting adaptive quality of service assurances to the mobile devices. Mobiware is well suited toward managing the evolving service demands of adaptive mobile applications and dealing with the inherent complexity of delivering audio, video and real-time services to mobile devices. Based on an adaptive quality of service API, mobiware consists of a set of controllers that interact with transport, network and medium access controller distributed objects that maintain application-specific adaptive quality of service needs.

2.1 The Adaptive-QOS API and Service Model

By trading off temporal and spatial quality with available bandwidth, mobile applications can be made to adapt to time varying conditions with minimal perceptual distortion. In [Binachi,98a], we introduced an *adaptive-QOS API* and service model specifically designed to quantitatively address the wireless-QOS and mobile-QOS needs of adaptive mobile applications. Mobile applications use an adaptive-QOS API at the transport layer specifying:

- *a utility function*, which captures the adaptive nature over which an application can successfully adapt to available bandwidth in terms of a utility curve that represents the range of observed quality to bandwidth. The observed quality index refers to the level of satisfaction perceived by an application at any moment as illustrated in Figure 1; and
- *an adaptation policy*, which captures the adaptive nature of mobile applications in terms of a set of adaptation policies (viz. fast, slow, after handoff, never). These policies allow the application to control how it moves along its utility curve as resource availability fluctuates, e.g., scale-up only after handoff, fast adaptation and slow adaptation.

The utility function allows *utility-fair* bandwidth allocation algorithms [Bianch,98a] to derive explicit optimization rules under heterogeneous application adaptation behavior. Here bandwidth is allocated fairly to all the flows so that the same utility value is achieved at an access point. For full details of the utility-fair bandwidth allocation algorithm see [Bianchi,98a]. The utility function alone, however, is not capable of capturing application specific adaptation dynamics. Rather, a simple set of adaptation policies is used to capture how an application wishes to respond to instantaneous bandwidth availability.

A mobile multimedia application's range of perceptible quality is strongly related to how and when it responds to resource changes. Frequent oscillation between what may be considered optimal and minimum utility or even the frequent small change around an average application quality may be annoying to many applications. Some applications may wish to limit the frequency of adaptation to change, e.g., multi-resolution application. In contrast, others may wish to exploit any opportunity for adaptation, e.g., real-time data application. By limiting or dampening the response to change an application attempts to follow trends in resource availability rather than fluctuations to instantaneous changes. Such a conservative adaptation policy may lead to a more stable operating point on an application's utility curve. This is in contrast to a policy that responds to instantaneous fast moving points that may suit other styles of mobile application. The API allows the application to specify temporal or event-based dimension to utility.

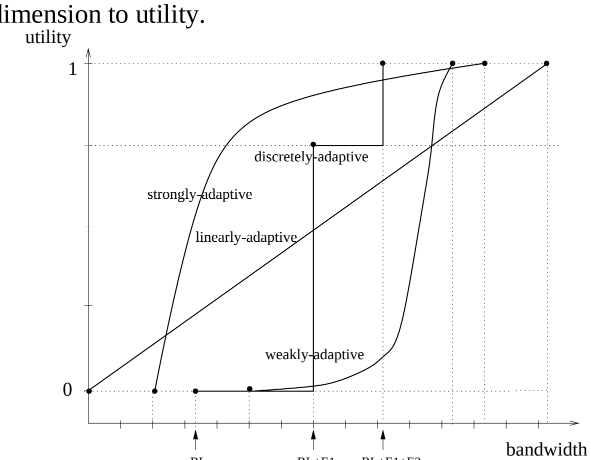

Figure 1: Utility functions

The service model supports the following adaptation 'menu' policy options[1]:

- *fast*, to instantly move up or down the utility curve responding instantly to any resources changes;
- *slow*, to move up or down the utility curve only after a suitable damping period has passed;
- *handoff*, to move up or down the utility curve only after handoff; and
- *never*, to never move up the utility curve after the initial bandwidth allocation.

The adaptive-QOS API is supported by mobiware at the transport and realized at the mobile device and in the network. This allows the mobile application to specify a flow's utility function and adaptation policy, which is supported by mobiware.

2.2 The Architecture

Mobiware is a software-intensive mobile networking environment based on distributed object technology. As illustrated in Figure 2 mobiware promotes the separation between mobile signaling and adaptation management, and the transport of media. At the transport layer, an *active wireless transport* supports the end-to-end transmission of audio, video and real-time data services based on an adaptive-QOS paradigm. The active wireless transport is an object-based transport that blurs the region over which traditional transports (e.g., TCP and RTP) typically operate to include access points and end systems.. Built on a set of Java classes, the transport system binds active and static transport objects at mobile devices and access points to provide end-to-end transport adaptation services. Static transport objects include segmentation and reassemble, rate control, flow control, playout control, resource control and buffer management objects. These objects are loaded into the mobile device as part of the transport service creation process. Active transport objects are dynamically dispatched to mobile devices and access points to support valued-added QOS. Currently, two styles of active transport objects have been implemented: active media filters [Balachandran,97], which perform temporal and spatial scaling for multi-resolution video and audio flows and adaptive FEC filters [Campbell,97a] which protect content against physical radio link impairments by matching the level of Reed Solomon channel coding to match time-varying error characteristics.

1 A generalization of this approach is detailed in [Bianchi,98a]. Adaptive mobile applications supply adaptation handlers, which implement application-specific adaptation policies supporting more sophisticated levels of adaptation than the current menu options (e.g., fast, handoff) offered in the existing system. Adaptation handlers can program simple or sophisticated adaptation strategies. Mobiware exposes a set of low level programming APIs to allow the application to control its adaptation strategy.

At the network layer, a *programmable mobile network* supports the introduction of new mobile adaptive-QOS services based on the xbind broadband kernel [xbind,96]. The network layer supports switching IP flows over ATM native transport services. Architecturally, the network comprises a set of network objects and adaptation proxies that operate at the mobile device, access points and at mobile capable switch/routers. Currently, an adaptive-QOS network service supports the delivery of multi-resolution flows having a base layer and one or more enhancement layers. The base layer provides a foundation for better resolutions to be delivered through the reception of enhancement layers based on the availability of resources in the wireless environment. Three key network algorithms include: (i) *QOS controlled handoff*, which gracefully scales flows (up and down) based on the semantics of the adaptive-QOS service during handoff when bandwidth availability varies; (ii) *mobile soft-state*, which provides mobile devices with the capability to respond to changes in wireless-QOS and mobile-QOS; and (iii) *flow bundling*, which exploits a common routing representation for all the flows to and from a mobile device to speed up handoff. *The focus of this paper is the design and evaluation of the programmable mobile network described in Sections 3 and 4, respectively.*

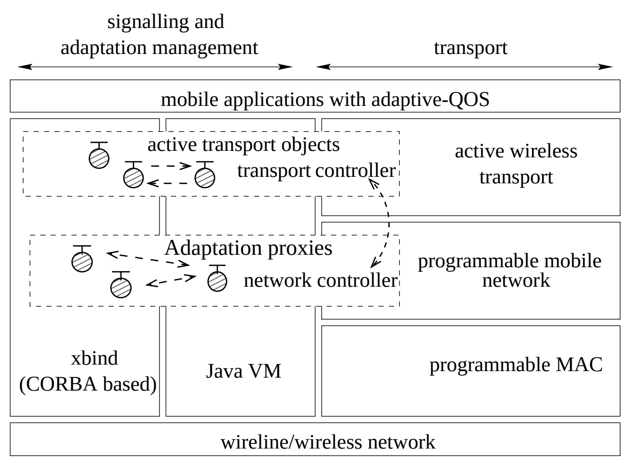

Figure 2: The mobiware architecture

At the data link layer, a *programmable MAC* [Bianchi,98b] combines a set of foundation services to support more sophisticated adaptive wireless-QOS services. Foundation services provide sustained rate services used to support minimum wireless-QOS assurances, and active and passive adaptive services to support application specific adaptation policy. The 'programmable' nature of the data link service provisioning over wireless networks provides an alternative approach to that found in the literature. Rather than supporting a specified set of 'hard wired' MAC services (e.g., VBR) by means of centralized control schemes, it provides a programmable air-interface that allows new services to be dynamically created and installed on the fly. This programmable MAC service support relies on a simple core architecture that pushes complexity and application specific adaptation decision making to the mobile device. For full details on the programmable MAC see [Bianchi,98b]

2.3 The Network Model

Mobiware is a novel software middleware platform that controls an experimental broadband wireless access network to a next generation mobile Internet [Mobicom,97] called *mobinet*. The network model comprises a set of mobile devices, wireless access points and mobile-capable switches/routers providing broadband cellular and ad hoc communication services to mobile users. Mobinet is based on ATM switching technology that supports IP switched flows in the access network. Mobile devices can be connected to mobinet via broadband cellular or ad hoc wireless access modes. In broadband cellular mode, mobile devices receive core network services via a set of wireless access points. Ad hoc devices may operate autonomously without the aid of any fixed infrastructure and core network services or can connect to the broadband cellular network via multiple ad hoc hops as illustrated in Figure 3.

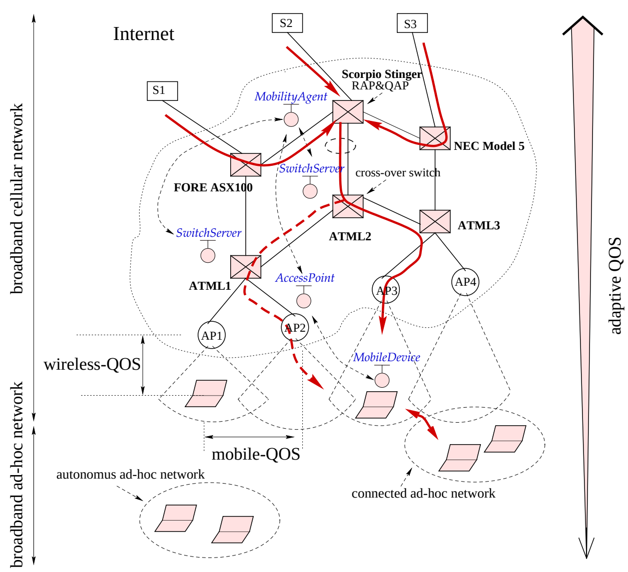

Figure 3: mobinet

Providing quality of service assurances in the broadband cellular networks is difficult. However, providing quality of service assurances without the aid of any fixed infrastructure as in the case of ad hoc is more challenging [Corson,98]. We believe there is a need understand the level of quality of service that can be supported at different points of attachment in mobinet, e.g., at the access point or multiple hops away from the access point. We observe that quality of service assurances are likely to diminish as a mobile device moves away from the core network as illustrated in Figure 3. Providing seamless quality of service support to mobile devices on the move (e.g., switching between broadband cellular and ad hoc modes) underpins mobiware's adaptive-QOS design approach[2].

3. Programmable Mobile Network

3.1 Programmable Objects

The mobile network comprises a set of programmable distributed CORBA objects[3] that support the delivery of adaptive-QOS flows to mobile devices over mobinet. The use of distributed object technology also provides support for interoperability between mobile devices utilizing different operating systems and protocol support. Mobiware objects execute on mobile devices, access points and mobile capable switch/routers supporting a set of mobile signaling and quality of service adaptation algorithms (viz. QOS controlled handoff, flow bundling and mobile soft state) as illustrated in Figure 3. Objects combine data structure (defining the object's state) with a set of methods (defining the object's behavior). Methods are executable programs associated with objects that operate on information in an object's data structure.

Per-mobile adaptation proxies support the communications of flows to/from mobile devices and are used to support fast and scalable handoff. These include *Routing Anchor Proxies (RAP)* objects, which 'bundle' (i.e., aggregate) flows to and from mobile devices for fast handoff [Campbell,97b] and *QOS Adaptation Proxies (QAP)* objects, which are used as an integral part of the mobile soft-state probing mechanism for distributed resource allocation over the wireless network.

To manage the network states introduced by flow-oriented communications and, more importantly, to gain efficiency across a wireless link, mobiware deploys a number of network objects that can execute on network nodes or on servers at the edge of the network. The mobile device object abstracts the operation of mobile stations and provides APIs for querying beaconing information, registration with an access point, flow establishment and handoff. It also includes the functionality to dynamically control the transport system at the access points, e.g., to set the media scaling or error

2 In this paper we focus on adaptive quality of service support for broadband cellular communications leaving ad hoc support for future work [Coroson,98].

3 Mobiware includes active transport objects based on Java classes that 'plugin' to the data path at wireless access points to provide value-added adaptation support. Currently, the active wireless transport supports active media scaling and adaptive FEC support.

control level for video flows. The mobile device object states mainly consist of quality of service specification (viz. utility function and adaptation policy) for all the flows directed towards/from the mobile station and end-point information for all the network nodes between the source-destination pairs.

```
interface AccessPoint : NodeServer {

  // flow setup from the current access point to
  // the network, called by the mobile device object
  void setupFlow(in long fbi,
      inout QOSSpecification qosSpec,
      inout FlowInfo flowinfo, in string<40> srcname,
      inout EndPoint peerEp,
      inout EndPoint RAP_fix, inout EndPoint RAP_mobile,
      inout EndPoint AP_fix, inout EndPoint AP_mobile,
      inout EndPointId airIP, inout double msr_time)
    raises(Reject);

  // handoff flow bundle for a specific mobile device
  void handoffFlowBundle(in long fbi,  inout QOSSpecList
      qosSpec[2],  inout FlowInfoList flowinfo[2],
      inout SourceList srcname[2], inout EndPointList
      RAP_fix[2], inout EndPointList RAP_mobile[2],
      inout EndPointList AP_fix[2], inout EndPointList
      AP_mobile[2], inout EndPointIDList airIP[2],
      inout double msr_time)
    raises(Reject);

  // refresh mobile soft-state for flow bundle
  // through the current access point to the network
  void refreshFlowBundle(in long fbi,
    inout EndPointList RAP_mobile[2],
    inout EndPointList AP_fix[2],
    inout EndPointList AP_mobile[2],
    inout double ntw_msr_time)
   raises(Reject);
};
```

Figure 4: CORBA IDL for access point object

An access point object supports APIs for binding to wireline network objects on behalf of mobile stations, propagating CORBA calls and for the establishment and periodic refreshing of local wireless connections for flows as illustrated in Figure 4. This object plays an important role in QOS controlled handoff and interacts with the transport system for the injection of active transport objects when and where needed. The mobility agent object provides flow-management, handoff signaling and mobility management services. It interacts with per-mobile RAP or QAP state in the switch servers and supports APIs for retrieving network topology information from a router object (e.g., location of the cross over switch) and communicating with each network node separately. A router object stores end-to-end route information.

Switch server objects [xbind,96] abstract and represent physical ATM switch/routers and are fully quality of service programmable. These objects support APIs for the reservation and release of namespace (viz. VCI/VPI pairs) and the allocation of network resources (viz. bandwidth). State mainly consists of per-flow connection information, stored in local hash-tables called switch caches. Switch server objects have been extended to be mobile capable, i.e., support RAP and QAP functionality. The General Switch Management Protocol (GSMP) protocol is used at the access points and switch/routers for accessing the switch tables.

3.2 QOS-controlled handoff

QOS controlled handoff gracefully scales flows (up and down) during handoff based on the semantics of the adaptive-QOS API described in Section 2.1. By scaling flows during periods of resource contention (e.g., during handoff), mobiware improves the wireless resource utilization and reduces the handoff dropping probability. While the style of handoff is entirely programmable, the current implementation style is mobile-initiated, forward handoff with soft-handoff on the down-link and hard-handoff on the uplink. By mobile-initiated, we mean that after a suitable dwell time the mobile device initiates a handoff by first registering with the forward/new access point. By soft-handoff, we mean that during handoff the mobile device simultaneously receives flows from the old and new access point on the down-link. In contrast, uplink flows use a 'break and make' approach between the old and new access points. During handoff, registration to the new access point, rerouting of flows and quality of service adaptation is accomplished by signaling objects and associated APIs outlined in Section 3.1. Signaling APIs are programmable[4] allowing various styles of handoff to be tailored toward particular radio environments.

QOS controlled handoff object-interaction is illustrated in Figure 5. Mobile device objects periodically 'hunt' for beacon signals from neighboring access points. Beacons are fully programmable in mobiware and carry low-level signal information in addition to the current bandwidth availability at the sending access point. The mobile devices' hunting algorithm periodically compares all beacons received over the current hunt period and cumulatively over multiple hunt periods. If the wireless-QOS[5] indicated in the beacon from the current access point falls below a pre-determined threshold the hunt algorithm selects a new access point for handoff. Handoff is initiated after a suitable dwell period after which the mobile device registers with the new access point starting the handoff process as indicated by (2) (3) in Figure 5.

[4] Note that the object oriented aspect of mobiware signaling makes it easy to rapidly 'program' different styles of handoff algorithm, e.g., the object APIs can support network initiated handoff that are hard in nature.

[5] The beacon informs the mobile device of the channel conditions upon the reception of each packet arrival reporting the signal level, silence level signal quality and antenna selected. The signal and silence level area derived from the receiver's automatic gain control settings. Beacon messages are augmented with a 16-bit field that indicates the available resources at the access point. The mobile device can use this to scale down request for bandwidth resources during handoff given that the bottleneck node is typically the access point. Our radios are based on WaveLAN operating in the 2.4-2.8 GHz ISM band, which we have low level access to for programming the beacon.

The device registration procedure triggers the new access point object to bind to a mobility agent object (4). The mobility agent caches bindings to the per-mobile adaptation proxies that are implemented as part of switch servers. Mobility agents and proxies can run anywhere in the mobinet, i.e., mobility agents can operate at fixed edge device, mobile device or on the switches. Mobility agents are the main controllers for managing handoff in mobiware. Currently, mobiware uses a single mobility agent to manage handoff for the complete testbed. Mobility management is a fully distributed algorithm that includes one or more mobility agents for scalability. When the mobile device initiates a handoff (5) it passes a unique mobile device identifier called the flow bundle identifier (FBI) to the access point that allows mobiware to identify the mobile device's flow bundle in the wireless access network.

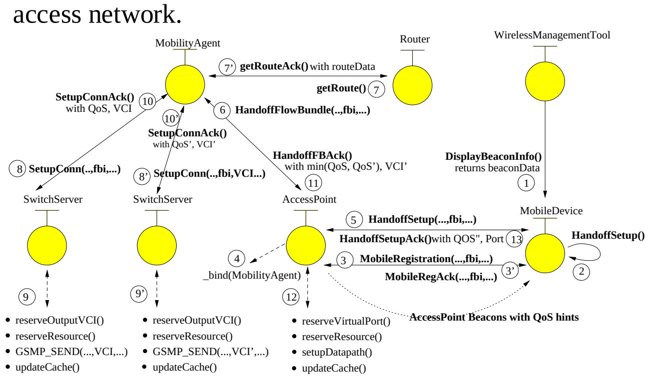

Figure 5: QOS-controlled handoff object interaction

Mobility agents are responsible for re-routing a mobile device's flow bundle from an old access point to a new one as illustrated in Figure 3. This entails the mobility agent invoking the route object to determine the location of the cross over switch as illustrated in Figure 3. Switch server objects are used to re-establish new flow state at all switches between the cross over switch and the new access point. The re-routing phase includes name space reservation (viz. outgoing VCI/VPI) and bandwidth value at each network switch and the new access point. The final process of re-routing a flow bundle through a switch includes the use of GSMP (9) (9') (12) to set up the switch table and reserve resources. However, GSMP does not support the concept of flow bundling. While the mobile agent informs the switch server objects to establish state for a complete flow bundle, the switch server interacts with GSMP on a flow by flow basis. In the evaluation section we describe enhancements to GSMP to support flow bundling. The mobility agents interact with mobile capable switch servers and the new access point in parallel resulting in a speed up of the re-routing phase of the handoff algorithm over conventional hop-by-hop signaling. After the re-routing of the flow bundle is complete the mobile agent informs the new access point of the negotiated quality of service and flow bundle VCI/VPI mappings. The new access point interacts with the active wireless transport to provide active media filters based on the available bandwidth at the interface.

To keep the name space binding between the mobile device and access points constant with mobility we have implemented the notion of virtual wireless ports. As mobile devices connect to different access points their VPI/VCIs mapping remain constant. The flow bundle to VPI/VCI bundle is resolved by a virtual wireless port, which is dynamically allocated by the new access point during handoff. This approach minimizes the impact of re-negotiation in comparison to a full name space re-negotiation, which would be disruptive during handoff.

3.3 Flow bundling

QOS controlled handoff and mobile soft state exploit flow bundling to speed up handoff and minimize the signaling overhead associated with maintaining the network state. Flow bundling [Porter,95] provides a common routing representation for all the flows to and from a mobile device as illustrated in Figure 3. This is similar to the virtual path concept in ATM networks or tunneling in IP networks. Flow bundling is a general method for encapsulating and routing. For example, flow bundles can carry both VPs and VCs in mobiware. Flow bundling is motivated by the need to reduce the complexity of re-routing multiple independent flows to and from mobile devices during handoff. By aggregating flows in this manner we can speed up handoff, simplify mobile soft-state probing and minimize signaling overhead. Using flow bundling, QOS controlled handoff simply discovers a single crossover switch then re-routes all flows to the new access point in a single object-level operation.

During handoff the flow bundle object interaction is as follows. The mobile device object invokes the access point's handoffSetup() method once for the flow bundle minimizing the signaling overhead at the air-interface. Mobiware supports the option of enabling or disabling bundling when flows are established. Note that when flow bundling is disabled the mobile device, access point, mobility agent objects treat each flow independently during handoff. As discussed in the evaluation this increases the signaling overhead and the handoff latency. During each invocation a separate cross over switch needs to be located, using a shortest path algorithm and individual flows need to be re-routed and signaled independently. The mobile agent interacts with the switch servers to re-establish flows and update switch tables for all switches between the cross over switch and the mobile devices. GSMP is used to update the switch table at each traversed switch. In order to support the atomic re-routing of flow bundles we have enhanced the GSMP protocol.

Flow aggregation at the switch control level has been implemented as a modification to the GSMP invocation mechanism. Two enhancements have been considered in Section 4. The first has no impact the current GSMP client server interaction. The mobile agent simply invokes the GSMP client for each flow without waiting for acknowledgement from each GSMP command. This results in the switch server sending a burst of GSMP setup messages to the switch then waiting for acknowledgements. The second enhancement augments the GSMP setup message to allow up to 256 VCI pairs to be passed across the interface in one client-server interaction. This allows the switch server to setup the switch table for a flow bundle in one operation. We discuss the performance benefits of flow bundling in Section 4.

3.4 Mobile soft-state

During handoff a flow bundle must be re-routed to a new access point, resources need to be reserved, and the old flow bundle state between the old access point and the cross over switch removed. Mobile devices resident in cells also need to scale flows in accordance with channel conditions, whether new flows are established or released, and when new mobile devices enter and leave cells. Mobile soft-state provides quality of service adaptation support to cater to a number of these conditions. Mobile soft-state results in the periodic establishment of bandwidth and name space resources for flow bundles between a mobile device and a per-mobile QOS adaptation proxy (QAP). Mobiware supports the idea of soft-state [Campbell,97] in the mobinet to refresh the network state. Periodically refresh messages sent by mobile devices are on a flow bundle basis not a per-flow basis. During the refresh phase mobile devices respond to any changes in allocated bandwidth (based on utility functions) to the flow bundle.

In [Campbell,97] we argue that a soft-state approach is well suited to supporting QOS adaptation in mobile networks. Mobile soft state supports a distributed probing mechanism based on flow bundles allowing mobile devices to compete fairly for bandwidth in a completely decentralized and scalable manner. During handoff mobile devices do not explicitly remove the old flow bundle-state between the old access point and the cross over switch. In this case, mobile soft-state timers located at the switches and old access point timeout and release resources automatically. Mobile devices resident at the old access point compete for these available resources thereby improving their utility.

Mobile devices periodically probe the path between the mobile device and the per-mobile QAP and contend for resources. Note, that per-mobile QAPs can be located at an access point or any mobile switch/router between the mobile and its corresponding routing anchor proxy (RAP). If the QAP is located at the access point then mobile soft state is only active over the air-interface; that is, between the mobile device and access point. The position and configuration of where these proxies reside is programmable. In many cases, the access point is the most suitable location because radio resources are generally the bottleneck in broadband wireless or wireless LAN systems.

Mobile devices independently probe the wireless access network. The probe includes a list of flow requirements for the complete flow bundle, which includes the utility function for each flow. In the current system, we have implemented the discretely-adaptive flows (see Figure 1) where a base layer (BL) and one or more enhancement layers (viz. E1, E2) would be signaled in the probe message. The probe interacts with the access point and all mobile capable switches/routers between the mobile device and its QAP. The response is returned to the mobile device indicating the available bandwidth reserved for each flow in the bundle over the next time interval. This time interval is called the refresh interval. If a probe message is received along the path before the refresh interval expires then the reservation is ‘refreshed’ based on available resources. In contrast, if no refresh message is received then the timer expires and the resources deallocated and state removed. This reservation-adaptation style probing and response is implemented at the object level as a set of refreshFlowBundle() method invocations (2) (3) (3’). The mobile device issues a refresh to the mobility agent object which then refreshes all switches and the current access point on the path between the mobile and QAP (see Figure 6).

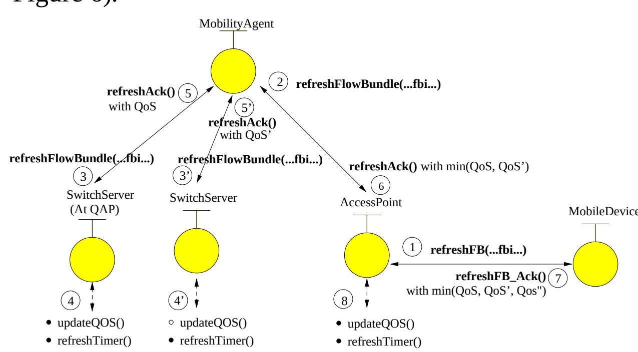

Figure 6: Mobile soft-state object interaction

4 Evaluation

To evaluate the programmable mobile network performance, we built the mobinet testbed, developed a wireless network management tool and designed a set of experiments to analyze QOS controlled handoff, flow bundling, mobile soft-state and QOS adaptation algorithms. Through the course of measurement, evaluation and re-design we managed to substantially improve upon our initial baseline design.

4.1 The Experimental Environment

At the network level, mobinet provides wireless access to the Internet and comprises four ATM switches[6] (viz. ATML Virata, Fore ASX/100, NEC model 5, Scorpio Stinger switches) and four wireless access points as illustrated in Figure 3. The access points run the mobiware code and are based on a set of multi-homed 200 MHz pentium PCs that provide radio access to a wireline switched IP/ATM sub-network. High performance notebooks (viz. IBM, Gateway and NEC) provide support to mobile applications and mobile access to the access network. Wireline ATM links operate OC-3 rates between the switches and fixed end-systems, and at 25 Mbps between the switches and access points. Currently, the air-interface is based on WaveLAN operating at 2 Mbps in the 2.5 GHz band. A future version of the mobinet will operate at 25 Mbps spectrum in the 5GHz NII/SUPERNet unlicensed band. Mobile capable switches, access points and mobile devices are abstracted as programmable CORBA objects. Mobiware requires IIOP CORBA for mobile signaling and adaptation management. For the results provided in this paper the mobile devices and access points use IONA's Orbix CORBA running under Windows NT, and the mobile capable switches/routers use IONA's Orbix running under UNIX or proprietary operating systems.

We have designed and implemented *mobiman* a Java-based management tool for wireless networks. Mobiman drives experiments, displays recorded statistics, and provides network management capability to view the network and inspect the state of any mobile device. To measure the performance of the programmable mobile network we inserted measurement checkpoints throughout the code and recorded performance statistics during the experiments. Mobiman, which runs on any fixed or mobile device, can remotely target any mobile device operating in mobinet displaying mobile device object statistics. It can setup flows, turn on/off flow bundling and mobile soft state, interact with media scaling and adaptive FEC control at the transport level, and force handoff operations to occur. Measured information displayed by mobiman comprises flow information, quality of services measurements (e.g., signal level, silence level signal quality, and access point bandwidth availability) and experimental checkpoint measurements. Mobiman displays measured information, wireless network topology and mobile device location in a control window as illustrated in Figure 7. When a mobile device is selected by mobiman a control window indicates the state of the mobile, e.g., three flows are delivered to 'mobile-air1'. In this example, the mobile-air 1 is running mobiman and the three flows correspond to the playout of the "True Lies" video clip and two low-resolution text windows. A flow setup panel appears in the top-left corner of Figure 7.

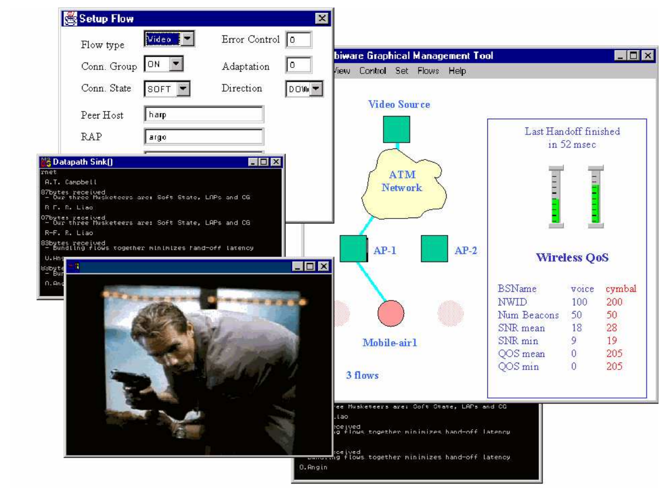

Figure 7: mobiman: a Java-based management tool for wireless networks

Microsoft's Active Movie is used for the reception, decoding and rendering of digital video. It provides a software tool capable of controlling and processing streams of multimedia data. Active movie uses modular components called filters and filter graphs. Typically, a filter graph consists of a source filter that provides the system with multimedia data, a transform filter that performs data decompression and a rendering filter. Active Movie's filter graph has been enhanced with an appropriate mobiware static transport object to perform synchronization of flows during handoff, controls delay-jitter control and rate control.

4.2 The Experiments

Our evaluation methodology is based on a set of experiments designed to investigate the performance of the programmable mobile network in supporting mobile multimedia communications. The use of CORBA for mobile signaling, wireless adaptation and mobile network programmability is a novel aspect of our work. CORBA objects run at the edges and in the mobile network to

6 We modify the concept of switchlets [Merwe,97] to provide an extended network for mobiware evaluation. Switchlets allows multiple virtual network elements to be operational within the same physical nodes. Three ATM switches (ATML 1, 2, and 3 as shown in Figure 3) in our network are switchlets physically co-located at the same physical switch. For example, packets traversing three switchlets located at the one physical switch travel across the physical switch three times via cables that directly connect one port to another in the same switch. Each switchlet corresponds to a different CORBA switch server object with different name space, and manages its own resources and controls connections independently of others.

support wireless-QOS and mobile-QOS. An important aspect of our evaluation was to determine if such distributed object technology was viable in supporting mobile signaling and adaptation management.

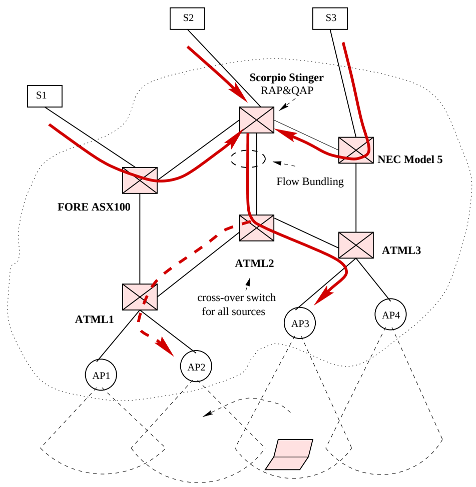

Figure 8a: Handoff with flow bundling ON

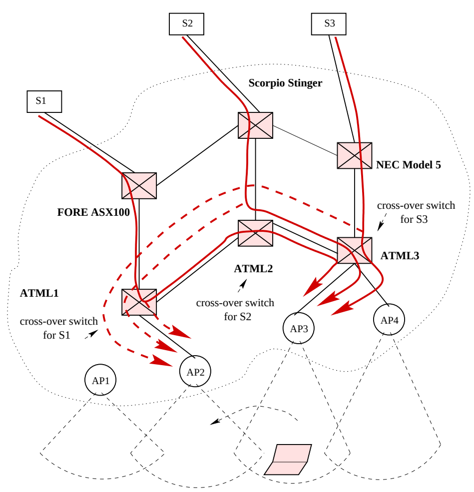

Figure 8b: Handoff with flow bundling OFF

4.2.1 *Handoff Analysis*

An important objective of this experiment is to measure the handoff latency and understand how the signaling system delays breakdown. In this experiment we investigated the handoff of a single flow. Handoff with flow bundling is described in the next section. For this experiment we streamed a single video flow from a fixed network server (S1) to a mobile device (M1). The mobile device moved repeatedly between access points AP2 and AP3 with the crossover switch located at the ATML2 switch. The average handoff latency for the baseline code is 171 msec. This figure broke down into 102 msec for mobile registration and object binding, 30 msec for wireless ATM connection setup and 35 msec for wireline connection setup. The greatest portion of the total latency time being absorbed by the binding process between objects during handoff. As described in Section 3.2, the mobile device object remotely binds to the access point object at the forward access point. Following this, the access point locally binds to a QOS mapper object and remotely binds to a mobility agent object for handoff control.

The following enhancements were made to the baseline code to reduce binding and Remote Procedure Call (RPC) overhead. First, by collapsing unnecessarily independent CORBA objects into a single object the binding overhead was reduced. To reduce binding over the air-interface the mobile proxy and QOS mapper in the access point object was collapsed. This reduced the number of CORBA requests across the air-interface reducing the binding time to from 102 msec to 42 msec. Collapsing objects in this manner reduced the handoff latency to 111 msec as illustrated in Figure 9. Next, by reducing the number of CORBA RPCs the overhead between objects during handoff we reduced. An RPC across the air-interface between the mobile and access point took an average of 15 msec to complete. The number of RPCs between the mobile and access point was reduced from four to two (viz. registration, handoff request). Reducing the number of CORBA RPCs during handoff reduced the handoff latency by 28 msec to 93 msec as illustrated in Figure 9.

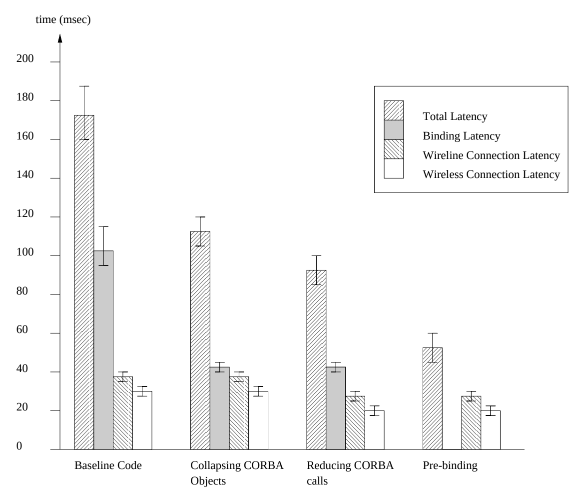

Figure 9: Handoff latency measurement results

The final enhancement to the baseline handoff code exploited the concept of caching object bindings. In order to eliminate the binding latency, we setup and cached bindings between remote CORBA objects prior to handoff being initiated which we call this *pre-binding.* All access points periodically broadcast their beacon including address information, signal strength and available resources. When mobile devices receive these beacons in promiscuous mode they register the signal quality in lieu of a possible handoff to a new access point. A pre-bind capability was to be added to the programmable mobile network to allow mobile devices to pre-bind to neighboring access points in advance of handoff. The pre-binding criterion is based on the signal strength and available resources. The pre-binding algorithm issues a pre-bind to an access point object on-the-fly establishing TCP connections for the CORBA IIOP between the mobile device and the access points. Another enhancement establishes binding between all access point objects in a domain and its associated mobile agent object. This final enhancement reduced the average handoff latency from 171 msec to 51 msec.

4.2.2 *Flow Bundling Analysis*

This experiment evaluates the performance gains using flow bundling during handoff. We observe the performance of handing off multiple flows with flow bundling disabled and then enabled. In the experiment video is streamed from three independent sources (viz. S1, S2, S3) across the network to a single mobile device, which is repeatedly moving between access points AP2 and AP3. When flow bundling is disabled each flow S1, S2 and S3 is independently re-routed during handoff via the ATML1, ATML2 and ATML3 crossover switches, respectively. When flow bundling is enabled all flows are bundled at the per-mobile QAP located at the Scorpio switch and re-routed during handoff via a single cross over switch located at the ATML2 switch as illustrated in Figure 8a.

In this experiment, we vary the number of video streams from one to ten flows with bundling enabled and disabled and measure the handoff latency. The results from the baseline measurement highlight the speed up achieved using flow bundling in the access network as illustrated in Figure 10. For two flows with and without flow bundling enabled the handoff latency is 200 msec and 250 msec, respectively. As the number of flows increase the benefit of bundling becomes more pronounced. For ten flows with and without flow bundling enabled the handoff latency is 320 msec and 780 msec, respectively. The adoption of flow bundling provides an improvement of 20% for two flows and 59% for ten flows. With flow bundling enabled (as illustrated in Figure 8a) the handoff latency converges whereas with flow bundling disabled, the latency increases almost linearly as the number of flows increases. These results indicate the benefit of using flow bundling to reduce handoff latency and signaling overhead. This is mainly due to the fact that all interactions between objects during handoff deals with aggregated signaling rather than per-flow signaling.

The baseline code was enhanced to support the optimization described in section 4.2. In addition, the GSMP messaging between the switch server and GSMP provided some incremental improvements. The handoff latency was reduced to 56 msec with bundling and to 67 msec without bundling for two flows showing a 16% improvement. In contrast, the handoff latency for ten flows was 155 msec and 420 msec with and without bundling, respectively showing a 63% improvement.

The baseline code only provides flow bundling support between the mobile device, access point and mobility agent objects. The interface between the mobility agent and the switch server are per-flow. Another observation is that the GSMP interface between the switch server object and switch does not provide any support for aggregation, i.e., GSMP client cannot update the switch table for more that one VCI pair. To address this we enhanced the GSMP interfaces used by the switch server object. This resulted in seamless support for flow bundling aggregation from the mobile device to the switch tables providing some level of speed up as illustrated in Figure 10. The GSMP enhancements described in Section 3.3 include a "parallel" enhancement, which did not require any changes to the GSMP code. In this case, for two flows the latency for total GSMP messages is 849 μsec without aggregation, and 511 μsec with aggregation showing a 40% improvement. With increasing number of flows, the total gain obtained by aggregation increases up to 70% for ten flows (3907 μsec vs. 1184 μsec).

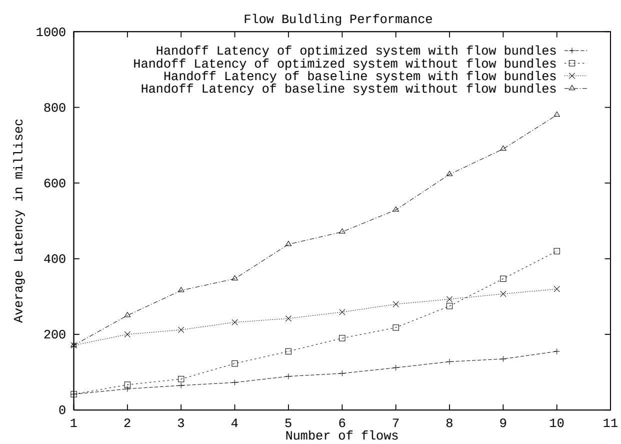

Figure 10: Performance gain with flow bundling

4.2.3 *Mobile Soft-State Analysis*

This experiment demonstrates the ability of mobile devices to adapt their bandwidth needs to changes in wireless-QOS and mobile-QOS based on mobile soft state. Mobile devices periodically probe and adapt to changes in available resources in wireless access networks. Users characterize flows using an adaptive-QOS API (described in Section 2.1) that includes a utility function and adaptation policy. In this experiment we present two scenarios that illustrate the benefit of mobile soft state in wireless and mobile environments.

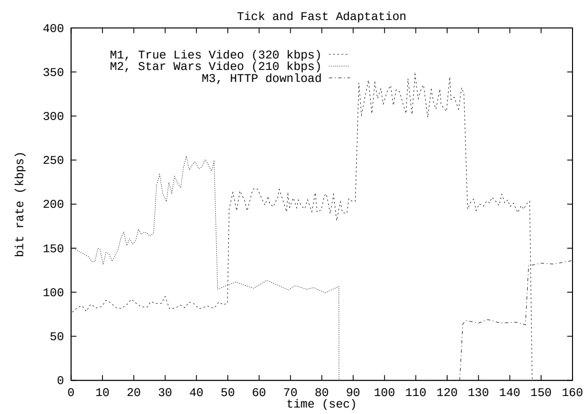

Figure 11a: QOS adaptation within a single cell

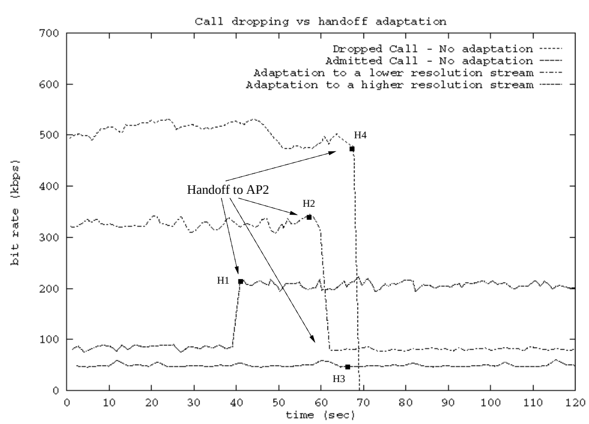

Figure 11b: Handoff driven QOS adaptation

The first scenario illustrated in Figure 11a shows the QOS adaptive behavior of two mobile devices M1 and M2 operating within a single wireless cell. Mobile devices M1 and M2 receive the “True Lies” and “Star Wars” video streams, respectively. Both video flows are based on discretely-adaptive utility functions, i.e., multi-resolution flows. Initially, M1 and M2 receive a base layer (BL) at 80 Kbps, and a base and enhancement layer (E1) at 150 Kbps, respectively. Currently, the adaptive-QOS service gives priority to support the minimum bandwidth requirements of multi-resolution flows [Campbell,95]. During the scenario, M2 registers an increase in bit error rate as it moves away from its current access point. Adaptive FEC is applied to the video between the access point and M1 based on the observed SNR and the measured bit error rate. An adaptive FEC object selectively codes the base and enhancement layers of the “star wars” video increasing the bandwidth consumed by M1 from 150 Kbps to 250 Kbps. For the experiment, the maximum capacity of the air-interface is set to 330 Kbps and around 50 seconds into the scenario the M1 video is adapted back to the base layer with FEC only. Resources released by M1 are consumed by M2 increasing its utility at 50 seconds[7] into the scenario. This situation remains constant until M1 handoffs to a new access point after 80 sec allowing the access point to deliver another enhancement layer to M2. Note that mobile initiated adaptation to released resources (i.e., scaling up) is somewhat dependent on the refresh/probe interval. When a new mobile devices M3 enters the cell around 120 sec access point M1 is explicitly scales back by dropping an enhancement layer. Toward the end of the scenario M3 probes and scale up to a better perceptible quality as M2 hands off to a new access point. At 140 sec into the scenario, mobile device M3 sets up a new flow to access web services initially downloading a GIF file at a rate of 70 Kbps scaling up to 135 Kbps. In related work [Bianchi,98a] we are investigating a generalized adaptation policy mechanism where applications can specify application specific adaptation semantics. For example, some applications would not wish to experience the adaptation observed by M2 while others may be as aggressive as M3 in exploiting any available resources.

The second experiment highlights a number of different QOS adaptation scenarios that can take place during hand off. In this experiment, mobile devices hand off to the access point AP2 from AP1 and AP3. In this experiment, QOS adaptation is not, however, based on the mobile soft-state refresh mechanism described and evaluated in the previous section. Rather, as part of the QOS re-negotiation phase during handoff, mobile devices scale their quality of service needs to match the available resources. The handoff point at which each of the four mobile devices (viz., M1, M2, M3, M4) enter the new access point AP2 is illustrated in the trace shown in Figure 11b. The type of adaptation that takes place after handoff points, which is marked as H1 through H4, is illustrated. During handoff a number of adaptation scenarios may occur depending on the available resources and the ability of existing mobile devices to adapt. For example, the new access point may force existing mobile devices to drop enhancement layers to allow a new mobile to enter the cell with minimum QOS assurances. In this experiment, mobile device M1 enters the new cell at H1 and scales up its utility to take advantage of available resources. M1 adaptation policy is to only scale

[7] Mobile device M1 probes and adapts to the additional bandwidth within a single refresh interval that currently set to 10 sec. There are a number of tradeoffs in setting the probe interval. A Smaller duration allows mobile devices to aggressively grab resources on a fast time scale. However, this increases the signaling load overhead. We are investigating coupling the probing and adaptation time scales to the application level adaptation policy [Bianchi,98a]

after handoff. At point H2 mobile device M2 hands off to the access point AP2 and is forced to scales down to its base layer. Mobile device M3 has an adaptation policy of never adapting. At H3 the mobile hands off to AP2 and maintains its current utility. In the final part of the experiment, M4 hands off to AP2 at point H4. In this instance, insufficient resources are available to support the base layers of M1, M2, M3 and M4 forcing the access point to deny the handoff.

5. Conclusion

In this paper we presented the design, implementation and evaluation of an open programmable mobile network based on distributed object technology called mobiware that dynamically exploits the intrinsic scalable properties of adaptive mobile applications. We have analyzed the performance of mobiware's QOS controlled handoff, flow bundling and mobile soft state algorithms. While the baseline code raised some initial performance concerns about the viability of using distributed object technology for controlling mobile networks the enhanced software is extremely competitive in relation to existing work. The latency measured for QOS controlled handoff was reduced from 171 msec to 51 msec for the handoff of a single flow through two ATM switches making handoff through a single switch in the order of 25 msec. In [Porter,95, Naylon,97] and [Mishra,97] handoff latencies were measured to be 10 and 30 msec for a single flow through a single cross over switch. The use of flow bundling techniques in mobile networks shows great performance increases as the number of flows increase during handoff. The handoff latency for ten flows is 155 msec when flow bundling was enabled and 420 msec when disabled. This clearly shows the advantage of such aggregation techniques. Mobiware's flow bundling compares favorably to the literature. In [Mishra,97] Partho reported a handoff latency of 125 msec for ten flows using native ATM signaling code. Mobile soft-state also exploits aggregation techniques provided by flow bundling. This allows resource probing to be based on flow bundles rather than per-flow. In the paper QOS adaptation techniques clearly demonstrates the benefit of mobile soft-state in sharing resources among competing mobiles in a cell.

A number of researchers have applied distributed object technology to mobile systems. Our work, however, differs from these efforts. First as part of the open signaling community [OPENSIG,96] we are deeply interested in identifying open programming interfaces for mobile and wireless networking. In this work we have identified a number of objects, APIs and algorithms that provide QOS support for adaptive mobile systems. The mobiware technology we have developed over the last two years marks a considerable software effort. To our knowledge we are one of the first group to apply distributed object technology as a mobile middleware solution. Mobiware objects execute on the mobile devices, at the access points and on switch/routers exposing open APIs that can be programmed to support mobile signaling, QOS adaptation management and wireless transport. We observe that once the wireless and mobile APIs have been designed the programming of new network algorithms, e.g., QOS controlled handoff, flow bundling and mobile soft-state was straightforward engineering.

6. Acknowledgements

We would like to thank Aurel A. Lazar and the xbind team for providing the xbind broadband kernel which mobiware is built on. Also, we would like to thank the following colleagues for their major contributing toward the implementation of mobiware: Oguz Angin implemented flow bundles, Anand Balachandran implemented the active media filters, Javier G. Castellanos, retooled the beacon to include QOS hints, Michael E. Kounavis implemented the transport system, Raymond Liao implemented the signaling system, Chien-Ming Yu implemented the active error control and finally Yasuro Shobatake (Toshiba Corp. Japan) implemented the wireless transport management. We would also like to thank Laura Zhou for implementing mobiman and Mun Choon Chan (Lucent Technologies) who implemented the xbind connection management system from where the mobiware mobility agent grew. Finally, we would like to thank Lucent Technologies for providing a beacon API.

7. References

[Balachandran,97] A. Balachandran, A.T. Campbell, M.E. Kounavis, "Active Filters: Delivering Scalable Media to Mobile Devices", *Proc. Seventh Intl. Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV)*, St Louis, May, 1997.

[Bianchi,98a] G. Bianchi, A.T. Campbell and R. R.-F. Liao, "Supporting Utility-Fair Adaptive Services in Wireless Networks", *Proc. of 6th International Workshop on Quality of Service (IWQOS'98)*, Napa Valley, May, 1998.

[Bianchi,98b] G. Bianchi and A. T. Campbell, "A Programmable MAC", to appear: ICUPC'98, Florence, October, 1998.

[Campbell,95] A.T. Campbell, D. Hutchison, and C. Aurrecoechea, "Dynamic QOS Management for Scalable Video Flows" , *Proc. Fifth Intl Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV)*, Durham, New Hampshire, 1995.

[Campbell,97a] A.T. Campbell and G. Coulson, "QOS Adaptive Transports: Delivering Scalable Media to the Desk Top", *IEEE Network.*, March 1997.

[Campbell,97b] A.T. Campbell, "Mobiware: QOS-Aware Middleware for Mobile Multimedia Communications", 7th IFIP

International Conference on High Performance Networking, White Plains, NY, April, 1997.

[Corson,98] M.S. Corson and A.T. Campbell, "Toward Supporting Quality of Service in Mobile Ad hoc Networks, work in progress session, *First IEEE OPENARCH'98*, San Francisco, CA, April 1998.

[Katz,94] R. H. Katz, "Adaptation and Mobility in Wireless Information Systems", *IEEE Personal Communications Magazine, Vol. 1, No.1,* First Quarter 1994.

[Lazar,97] A. A. Lazar, "Programming Telecommunication Networks", *IEEE Network*, October 1997.

[Lee,95] K. Lee, "Adaptive Network Support for Mobile Multimedia", *Proc. of Mobicom'95*, Berkeley, CA, Nov. 1995.

[Lu,97] S. Lu, K.-W. Lee, V. Bharghavan, "Adaptive Service in Mobile Computing Environment", *Proc. of 5th IFIP Intl. Workshop on Quality of Service (IWQOS'97)*, New York, May 1997, pp25-36.

[Merwe,97] Van der J.E. Merwe and I. Leslie, "Switchlets and Dynamic Virtual ATM Networks", Integrated Network Management V, A.A. Lazar, R. Saracco and R. Stadler, eds., Chapman and Hall, New York, 1997.

[Mishra,97] P. Mishra, "Implementation and Experimental Evaluation of Mobility-enhanced ATM Signaling", OPENSIG FALL '97 Workshop Open Signaling for ATM, Internet and Mobile Networks, New York, October, 1997.

[OPENSIG,96] http://comet.columbia.edu/opensig/

[Mobicom,97] Panel on QoS in the Next Generation Mobile Internet: What is Feasible?, Chaired by A.T. Campbell, The Third Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom'97), Budapest, October, 1997.

[Mobiware,98] Mobiware source code: http://comet.columbia.edu/wireless/software/mobiware

[Naghshineh,97] Naghshineh M., and M. Willebeek-LeMair, "End-to-End QOS Provisioning in Multimedia Wireless/Mobile Networks" IEEE Network, March 1997.

[Naylon,97] Naylon, J., "Radio Handover Measurements", OPENSIG SPRING '97 Workshop Open Signaling for ATM, Internet and Mobile Networks, Cambridge, England, April 1997.

[Porter,95] J. Porter, A. Hopper, D. Gilmurray, O. Mason, J. Naylon, and A. Jones, "The ORL Radio ATM System, Architecture and Implementation", ORL Technical Report, 1995

[Tennenhouse,97] D. L. Tennenhouse, J. M. Smith, W. D. Sincoskie, D. J. Wetherall, G. J. Minden, "A Survey of Active Network Research", *IEEE Communications Mag.*, Vol. 35, No. 1, pp 80-86. January 1997.

[xbind,96] xbind Broadband Kernel source code: http://comet.columbia.edu/xbind