ELSEVIER

Computer Networks 31 (1999) 741–765

COMPUTER NETWORKS

# Programmable mobile networks

Andrew T. Campbell [*], Michael E. Kounavis, Raymond R.-F. Liao

*COMET Group, Center for Telecommunications Research, Columbia University, New York, NY 10027, USA*

**Abstract**

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 demands placed on underlying mobile signalling, adaptation management and wireless transport systems in support of new mobile services, e.g. interactive multimedia and web access. In this paper we present the design, implementation and evaluation of mobiware, a middleware technology that enables the introduction of new services in mobile networks. Mobiware provides a toolkit that service providers can utilize to build services that can dynamically exploit the intrinsic scalable properties of mobile multimedia applications in response to time-varying mobile network conditions. Based on an open programmable networking paradigm, mobiware runs on mobile devices, wireless access points and mobile-capable switch/routers providing a set of open programmable interfaces and distributed objects for adaptive mobile networking. Mobiware is software-intensive and is built on CORBA and Java distributed object technologies. The source code for mobiware v1.0 is freely available (comet.columbia.edu/mobiware) for experimentation. © 1999 Elsevier Science B.V. All rights reserved.

*Keywords:* Mobiware toolkit; QoS controlled handoff; Flow bundling; Mobile soft-state; Active filtering

## 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 devices 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 quality of service (QOS). Second, user mobility can trigger rapid degradation

[*] Corresponding author. Tel.: +1-212-854-3109; Fax: +1-212-316-9068; E-mail: campbell@comet.columbia.edu.

in the quality of the delivered signal. This can lead to transient service outages resulting in handoff dropping in broadband cellular networks when a new access point is unable to accommodate a new mobile device at its current level of service. As a result, mobile applications can experience unwarranted delays, packet losses or loss of service. We describe the quality index maintained during handoff between access points as mobile-QOS.

There is a growing consensus that adaptive techniques [14] present a viable approach to countering time varying quality of service impairments found in wireless and mobile networking environments. However, providing system-wide (i.e., end-system and network) adaptive quality of service support for mobile multimedia communications is complex to realize in practice and not well understood by the community [21]. Recently, a number of adaptive mobile systems [16,12,28,32,18,23] have been proposed in the literature; however, few experimental systems exist today to assess the viability of the adaptive approach. We believe that there is a need to build adaptive mobile networking testbeds that support the introduction of new adaptive services, study their behavior, and learn from these experiments in building more scalable adaptive mobile systems. We believe that there is a need to take a ‘‘hands-on’’ system approach coupled with the analysis of well-founded adaptive quality of service models to investigate the viability of the approach and the utility of adaptive mobile networking to mobile users.

To address these challenges, we have built an open [15] and active [29] programmable mobile network [9,1] that is controlled by a software middleware toolkit called *mobiware* [22]. Mobiware extends earlier work by the COMET group on programmable broadband networks [30] 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 signalling, 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 based on CORBA technology [25]. 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 adaptive services to be built using distributed object computing technology. By active, we mean that adaptive quality of service algorithms can be represented as active transport objects based on Java classes 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 a detailed discussion of the design, implementation and evaluation of the mobiware programmable mobile network layer. 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 the mobiware programmable mobile network layer that includes algorithms for mobile signalling, adaptation management and wireless transport. This is followed by the evaluation of mobiware in an experimental networking setting and a discussion of our results in Sections 4 and 5, respectively. Finally, in Section 6 we present some concluding remarks.

## 2. Mobiware

Mobile applications need to be capable of responding to time-varying wireless-QOS and mobile-QOS conditions. To address this, wireless transport and adaptation management systems should be capable of transporting and manipulating content in response to changing mobile network quality of service conditions. Mobile signalling should be capable of establishing suitable network support for adaptive mobile services, e.g., the delivery of scalable flows or packet services with drop preferences. Medium access controllers must be capable of sharing the wireless link capacity among mobile devices supporting adaptive quality of service assurances when possible.

Mobiware is based on a methodology of open programmability [15] for the introduction, control and management of new adaptive mobile services. It provides a set of open programmable CORBA interfaces and objects that abstract and represent network devices and resources providing a toolkit for programmable signalling, adaptation management and wireless transport services. Mobiware aims to provide a foundation for open programmable mobile networking that is suited to managing the evolving service demands of adaptive mobile applications dealing with the inherent complexity of delivering scalable audio and video and real-time services to mobile devices. Built on an adaptive quality of service API, mobiware consists of a set of controllers that interact with transport, network and medium access control distributed objects that maintain application-specific adaptive quality of service needs.

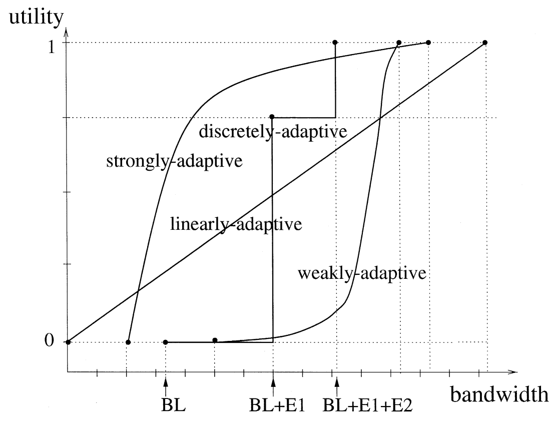


Fig. 1. Utility functions.

## 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 Ref. [4], 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 this API at the transport layer specifying:

- *A bandwidth 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 Fig. 1.
- *An adaptation policy*, which captures the adaptive nature of mobile applications in terms of a set of adaptation policies (viz. fast, smooth, 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 smooth adaptation.
- *Session preferences*, which allow the mobile user to classify all sessions (e.g., audio flows, video flows, data sessions, etc.) to and from a mobile device. This preference list is used during the QOS controlled handoff of flow bundles (see Section 3.2) to degrade lower priority sessions first, e.g., degrade a session to best effort support rather than drop it during handoff if insufficient resources are available or, in contrast, drop lower priority sessions if their minimum bandwidth requirements cannot be met during handoff.

The utility function allows *utility-fair* bandwidth allocation algorithms 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 Ref. [4]. 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 video applications. In contrast, others may wish to exploit any opportunity for adaptation, e.g., real-time data applications. 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 adaptive-QOS API is supported by mobiware at the transport-level and realized at the mobile devices and in the network. Mobile applications use this API to specify flow utility functions, adaptation policies and session preferences. The adaptive-QOS API allows applications to associate temporal or event-based dimensions to their utility functions.

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

- *fast*, to instantly move up the utility curve responding instantly to any resources changes;
- *smooth*, to move up 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.

## 2.2. The architecture

Mobiware is a software-intensive adaptive mobile networking environment based on distributed object technology. As illustrated in Fig. 2, mobiware promotes the separation between mobile signalling 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 mobile devices. 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 can be 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 [3], which perform temporal and spatial scaling for multi-resolution video and audio flows and adaptive FEC filters [22], which protect content against physical radio link impairments by matching the level of Reed Solomon channel coding to time-varying error characteristics. Active filters exploit the intrinsic scalable properties of multi-resolution audio and video to actively filter flows at access points in order to best utilize the available wireless bandwidth. Active filtering of sessions (i.e., audio flows, video flows, real-time data sessions) at access points can help reduce handoff dropping by scaling and dropping packets in an intelligent manner. The network layer support for active filtering is described in Section 3.

At the network layer, a *programmable mobile network* supports the introduction of new mobile adaptive-QOS services based on the xbind broadband kernel [30]. The network layer supports switching IP flows over ATM native transport services. Architecturally, the network comprises a set of CORBA network objects and adaptation proxies that operate at the mobile device, access points and at mobile capable switches/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 mobiware 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-*

[1] A generalization of this approach is detailed in Ref. [4]. 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.

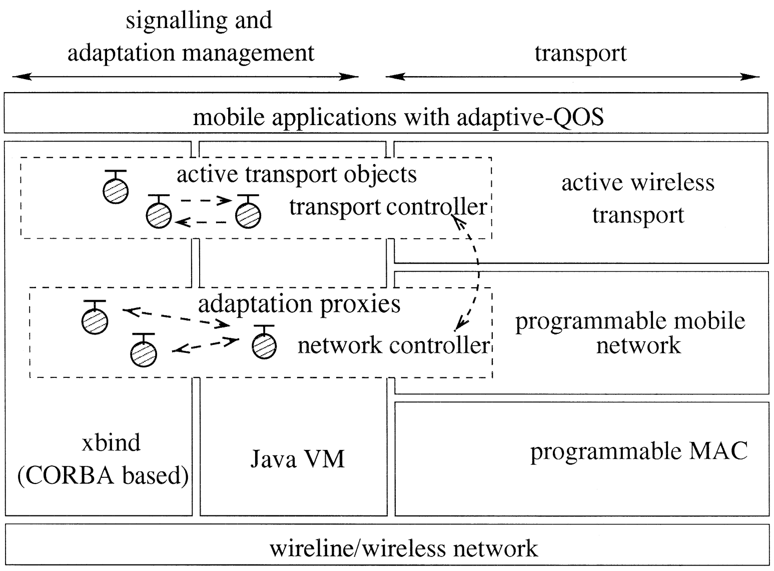

signalling and adaptation management
transport
mobile applications with adaptive-QOS
active transport objects
transport controller
active wireless transport
adaptation proxies
network controller
programmable mobile network
xbind (CORBA based)
Java VM
programmable MAC
wireline/wireless network

Fig. 2. The mobiware architecture.

*state*, which provides mobile devices with the capability to respond to changes in wireless-QOS and mobile-QOS; (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 layer described in Sections 3 and 4, respectively.

At the data link layer, a *programmable MAC* [5] 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 policies as discussed in Section 2.1. 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 of the programmable MAC see Ref. [5].

## 2.3. The network model

Mobiware provides a middleware toolkit that controls *mobinet*, an experimental programmable broadband cellular access network. The network model [2] 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

[2] It is envisioned that the Internet will provide interconnectivity between a set of these access networks providing a programmable cellular internetworking environment. In this case the Internet will support macro-level mobility using Mobile IP and the programmable access networks will support QOS controlled handoff, flow bundling, mobile soft-state and active wireless transport services to mobile devices.

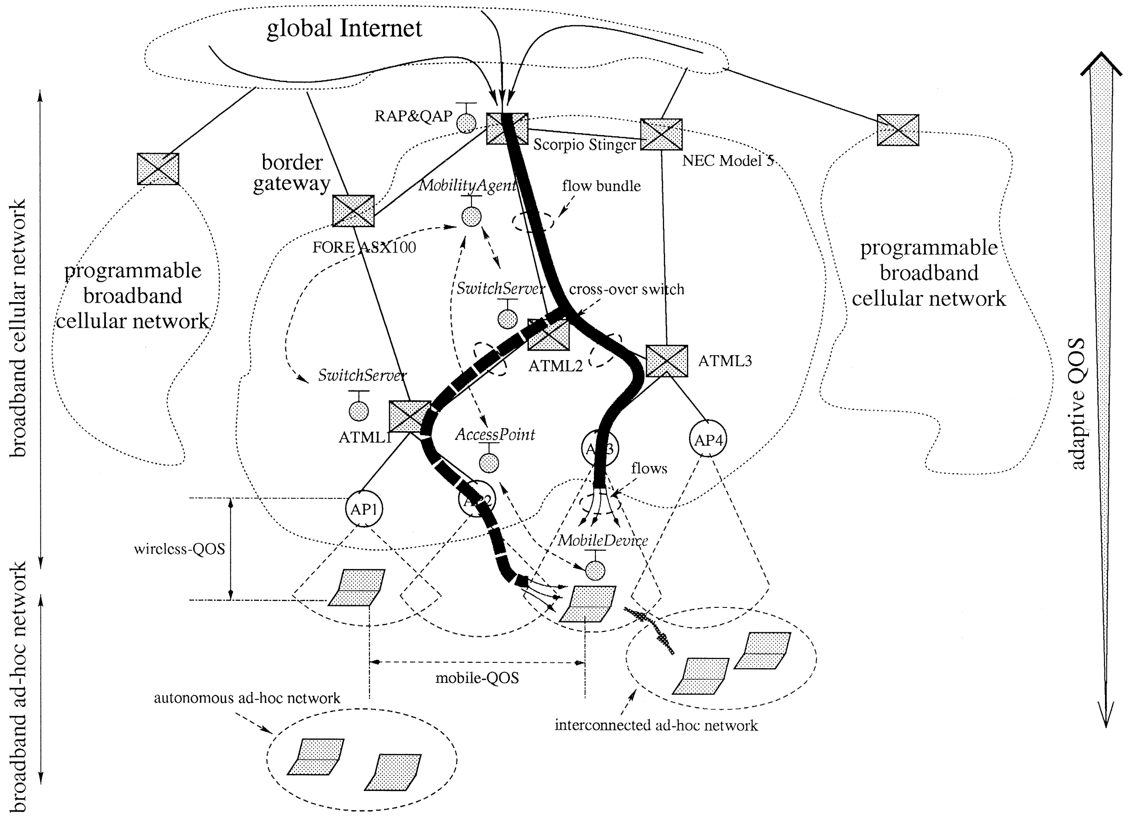


Fig. 3. Mobinet.

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 Fig. 3.

Providing quality of service assurances in broadband cellular networks is difficult. However, providing quality of service assurances without the aid of any fixed infrastructure as in the case of mobile ad hoc networking presents a greater challenge [11]. We believe there is a need to 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. 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. [3]

## 3. A programmable mobile network

### 3.1. Programmable objects

The mobile network comprises a set of programmable distributed CORBA objects [4] that support the delivery of adaptive-QOS flows to mobile

[3] In this paper we focus on adaptive quality of service support for programmable broadband cellular communications. Currently, mobile ad hoc support is under investigation by the COMET group [11,17].

[4] Mobiware programmable objects also include active transport objects based on Java classes that ''plug-in'' to the wireless transport data paths at access points to provide value-added quality of service adaptation support. Currently, the active wireless transport supports active media scaling and adaptive FEC.

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 signalling and quality of service adaptation algorithms (viz. QOS controlled handoff, flow bundling and mobile soft state) as illustrated in Fig. 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.

Two per-mobile proxy objects support the adaptation and handoff of flows in mobiware: (i) *QOS Adaptation Proxy (QAP)* objects play an integral role in allowing mobile devices to probe and adaptive to changing resource availability over the wireless link; and (ii) *Routing Anchor Proxy (RAP)* objects support the ‘‘bundling’’ (i.e., aggregation) of flows to and from mobile devices for fast, efficient and scalable handoff.

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. In the following we outline the function of some of the mobiware objects and their interface definitions. For full details on the objects and interfaces see Ref. [22].

A *mobile device object* abstracts the operation of mobile devices and provides APIs for querying beaconing information, registering with new access points, establishing flows, renegotiating quality of service and handing off flows. It also includes the ability to dynamically control the transport system at the access points, e.g., to set the media scaling or error control level for video flows. The mobile device object state mainly consists of quality of service specification (viz. utility function and adaptation policy, session preferences) for all the flows transported to/from the mobile device and routing information including the source/destination address and current RAP and access points addresses.

An *access point object* supports APIs for binding to wireline network objects (e.g., mobility agent) on behalf of mobile stations, propagating CORBA calls and for the establishment and periodic refreshing of local wireless flow state as illustrated in Fig. 4. This object plays an important role in QOS controlled handoff and interacts with the transport system for the injection of active transport objects.

A *mobility agent object* provides flow, adaptation and mobility management services. It interacts with per-mobile RAP and QAP state in the switch servers and supports APIs for retrieving network topology information from a router object (e.g., location of the crossover switch) and for interacting with the switch servers (see below) to establish, maintain and handoff flows in the cellular network.

A set of *switch server objects* [30] abstract and represent physical ATM switches/routers and are 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) is used at the access points and switches/routers for accessing the switch tables.

### *3.2. QOS controlled handoff*

QOS controlled handoff gracefully scales flows 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 helps reduce the handoff dropping probability. While the style of handoff is entirely programmable [2], 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 a 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 points 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,

```
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);

  // initialize mobile related states during registration
  void mobileRegistration(in long fbi,
    const char *rapName, const char *cmName,
    const char *msName)
   raises(Reject);
};
```

Fig. 4. CORBA IDL for the access point object.

rerouting of flows and quality of service adaptation is accomplished by signalling objects and associated APIs outlined in Section 3.1. Signalling APIs are programmable [5] allowing various styles of handoff to be tailored toward particular radio environments.

The QOS controlled handoff object interactions are illustrated in Fig. 5. Mobile device objects periodically ‘‘hunt’’ for beacon signals from neighboring access points. Beacons are made programmable by the mobiware toolkit 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 [6] indicated in the beacon from the current

[5] The programmable distributed object oriented nature of the mobiware signaling system makes it easy to ‘‘program’’ different styles of handoff algorithms (e.g., network initiated handoff) that can operate in parallel to other styles of handoff over the mobinet. For full details of the programming interfaces and results from the different styles of programmable handoff see Ref. [2].

[6] In addition to monitoring delay, loss and bandwidth characteristics of flows a mobile device object receives beacons that inform it of the state of the wireless link. 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 are derived from the receiver's automatic gain control settings. Beacon messages are augmented with a 16 bit field that indicates the available bandwidth at the access point. The mobile device can use this to scale down its 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, to which we have low level access for programming the beacons.

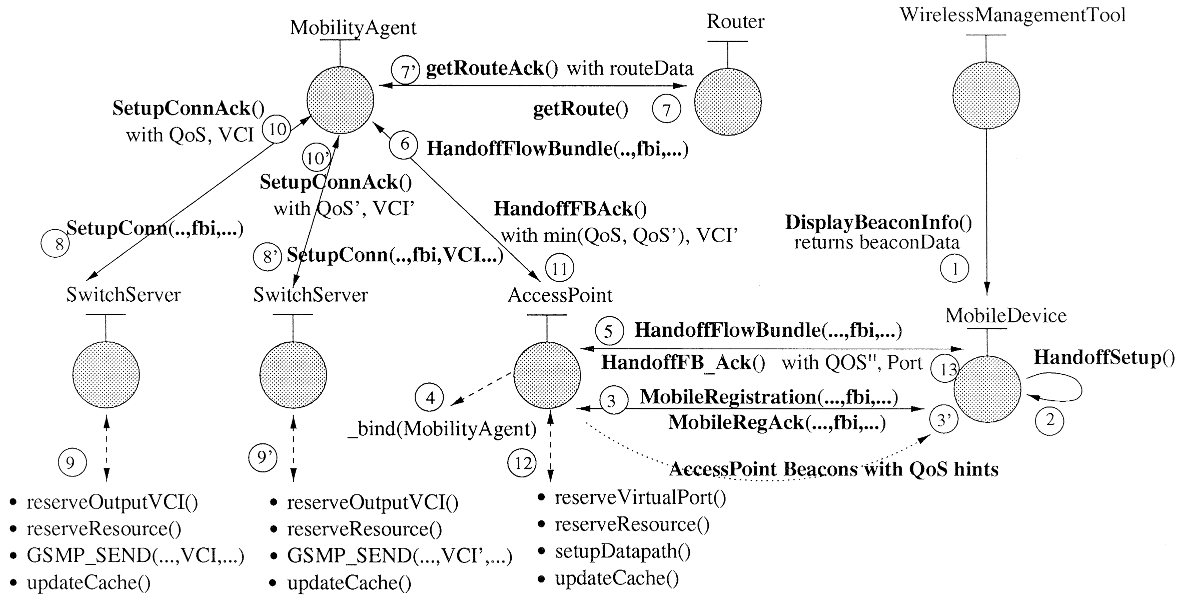

MobilityAgent
Router
WirelessManagementTool
7' getRouteAck() with routeData
getRoute() 7
6 HandoffFlowBundle(..,fbi,...)
SetupConnAck() with QoS, VCI 10
10' SetupConnAck() with QoS', VCI'
HandoffFBAck() with min(QoS, QoS'), VCI'
8 SetupConn(..,fbi,...)
8' SetupConn(..,fbi,VCI...)
11
DisplayBeaconInfo() returns beaconData 1
SwitchServer
SwitchServer
AccessPoint
MobileDevice
5 HandoffFlowBundle(...,fbi,...)
HandoffFB_Ack() with QOS'', Port 13
HandoffSetup() 2
3 MobileRegistration(...,fbi,...)
MobileRegAck(...,fbi,...) 3'
4 _bind(MobilityAgent)
AccessPoint Beacons with QoS hints
9
9'
12
• reserveOutputVCI()
• reserveResource()
• GSMP_SEND(...,VCI,...)
• updateCache()
• reserveOutputVCI()
• reserveResource()
• GSMP_SEND(...,VCI',...)
• updateCache()
• reserveVirtualPort()
• reserveResource()
• setupDatapath()
• updateCache()

Fig. 5. QOS controlled handoff object interaction.

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) and (3) in Fig. 5.

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 devices, mobile devices or on the switches. Mobility agents are the main controllers for managing handoff in mobiware. Mobility management is a fully distributed algorithm that includes one or more mobility agents for scalability. Currently, we allocate a single mobility agent to manage handoff in the experimental mobinet. 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.

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 Fig. 5. This entails the mobility agent invoking the route object (7) to determine the location of the crossover switch. Switch server objects are used to re-establish new flow state at all switches between the crossover switch and the new access point. The re-routing phase includes name space reservation (viz. outgoing VCI/VPI pair) 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 mobility 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 (8) (8′) (11) resulting in a speed up of the re-routing phase of the handoff algorithm over conventional hop-by-hop signalling. After the re-routing of the flow bundle is complete the mobility agent informs the new access point of the negotiated quality of service and flow bundle VCI/VPI mappings (11). The new access point interacts with the active wireless transport to provide active media filters based on the available bandwidth at the air interface [3].

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 renegotiation in comparison to a full name space renegotiation 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 signalling overhead associated with maintaining the network state. Flow bundling [27] provides a common routing representation for all the flows to and from a mobile device as illustrated in Fig. 3. This is similar to the virtual path concept in ATM networks or tunnelling in IP networks. Flow bundling is a general method for encapsulating and routing. 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 signalling overhead. Using flow bundling, QOS controlled handoff simply discovers a single crossover switch, and 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 HandoffFlowBundle( ) method once for the flow bundle minimizing the signalling overhead at the air-interface as illustrated in Fig. 5. 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, and mobility agent objects treat each flow independently during handoff. As discussed in the evaluation, this increases the signalling overhead and the handoff latency. During each invocation a separate crossover switch needs to be located, using a shortest path algorithm and individual flows need to be re-routed and signalled independently.

The mobility agent interacts with the switch servers to re-establish flows and update switch tables for all switches between the crossover switch and the mobile device. 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 to GSMP to support flow bundling are considered in Section 4. The first enhancement has no impact on the current GSMP client–server interaction. The mobility agent simply invokes the GSMP client for each flow in a flow bundle 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 set up 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 crossover switch must be removed. Mobile devices resident in cells also need to scale flows in accordance with channel conditions whenever 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 QAP. Mobiware supports the idea of soft-state [10] in mobinet to refresh the network state. The periodic refresh messages sent by a mobile device as part of the soft-state probing mechanism are sent on a per-flow bundle basis not a per-flow basis. This means a mobile device sends a single probe message for all flows supported at the mobile device rather than one probe per flow. During the refresh phase mobile devices respond to any changes in allocated bandwidth (based on utility functions) to a flow bundle.

In Ref. [8] 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 crossover switch. In this case, mobile soft-state timers located at the switches and the old access point timeout and release resources automatically. Mobile devices resident at the old access point compete for these available resources thereby potentially 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 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 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 discretely-adaptive flow support (see Fig. 1) where a base layer (BL) and one or more enhancement layers (viz. E1, E2) would be signalled 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 and the timer expires, resources deallocated and states are removed. This reservation–adaptation style of probing and response is implemented at the object level as a set of refreshFlowBundle( ) method invocations (2) (3) (3′) as illustrated in Fig. 6. 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 device and its QAP.

## 3.5. Active filtering

Media scaling is implemented in mobiware using a set of filter control objects (implemented as distributed CORBA objects) and active filters (implemented as Java classes). The goal of active filtering is to exploit the intrinsic scalable properties of multi-resolution sessions (e.g., audio flows, video flows and real-time images) and the knowledge of user supplied scaling preferences to actively filter sessions at the access point in order to best utilize the available bandwidth and to seamlessly deliver media with smooth change in the perceptual quality to mobile devices.

Active filters are written as Java bytecode classes and can be dispatched, configured and executed at access points. During QOS-controlled handoff existing active filters are propagated along with flow-state information from the old to the new access point using the wireline portion of mobinet. Active filters are driven by the mobile-soft state mechanism discussed above and are periodically ''tuned'' via an active filter interface to match the changing resource conditions found at the access point.

As illustrated in Fig. 7 the active filtering object model shows the interaction of these filter servers, filter control (FilterDaemon) objects and active filters. The media scaling object model is divided into three operational modes:

- *filter control* is a distributed signalling algorithm which comprises mobiware filter control objects. These objects are resident at the access points and

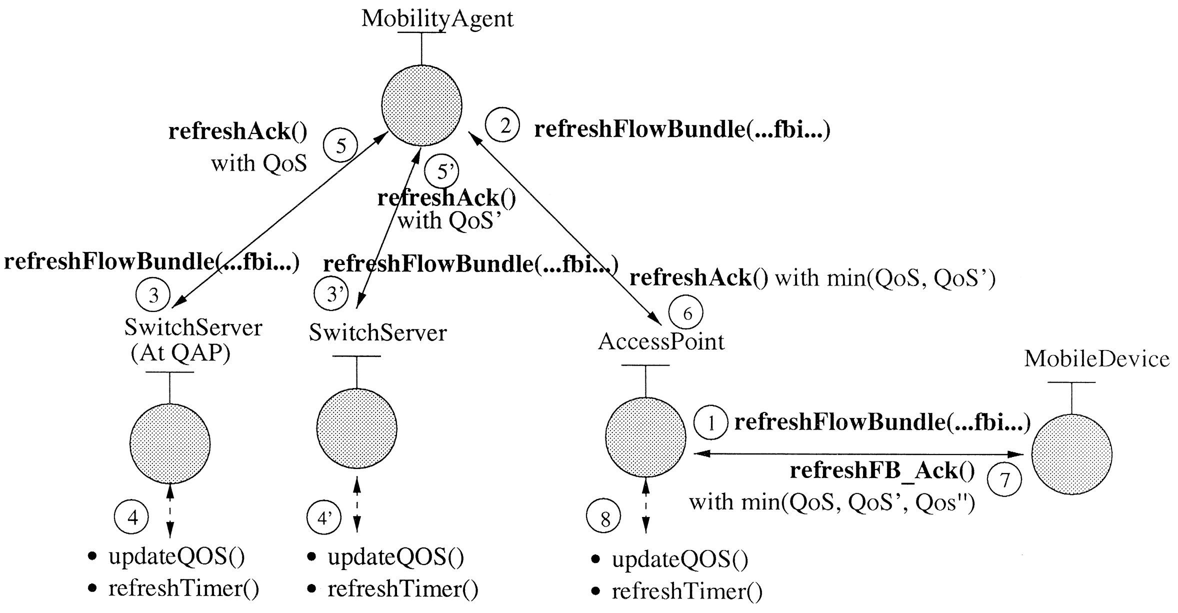


Fig. 6. Mobile soft-state object interaction.

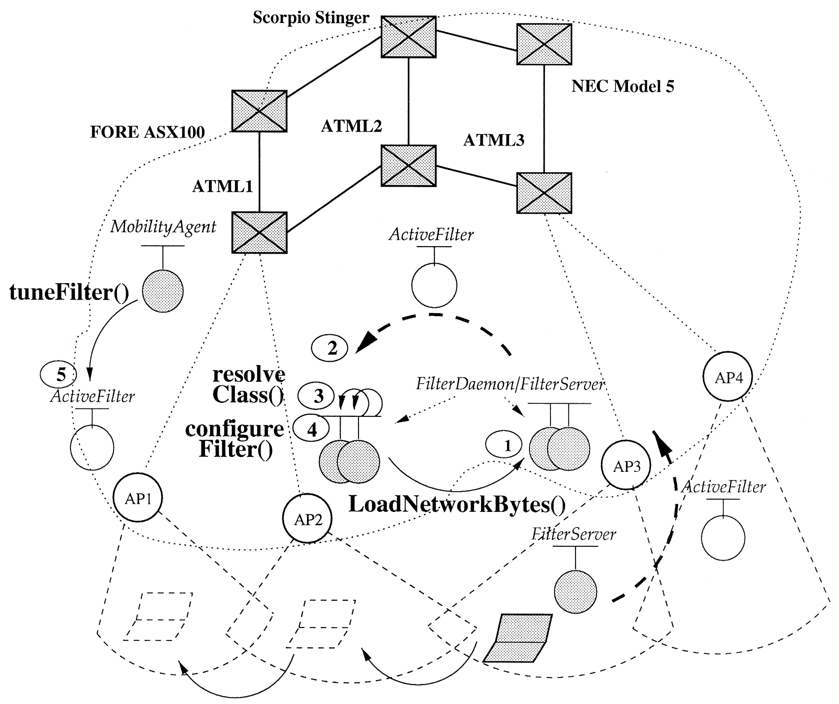


Fig. 7. Active filtering object interaction.

mobile devices. Filter control objects support a set of methods to select (1), dispatch (2) and configure (4) active filters;

- *filter instantiation* fetches remote Java bytecode classes and bootstraps (3) filters into the Java virtual machine environment. Once an active filter has been loaded and booted into an access point the local filter control object initiates a configure (4) operation to complete the instantiation phase. At this point active filters act autonomously in the active filtering styles; and
- *flow filtering algorithms* operate in the flow filtering mode where autonomous active filtering algorithms interact with the mobile soft-state mechanism to periodically tune (5) flows to match the scaling level to available wireless capacity.

One of the fundamental attributes of the active filtering approach is to provide mobile devices with the flexibility to select (1) active filter algorithms that best suit the application media scaling needs and the coding semantics of transported flows. Three classes of active filters are supported by mobiware for manipulating the rate of MPEG-1, MPEG-2 and MPEG-4 coded video:

- *media selector filters* [31] operate on the flow as it traverses the access points to ensure that the appropriate combination of base layer and enhancement layers are forwarded across the wireless link. Media selector filters only select and drop resolutions. They do not process the media as in the case of the next two more computationally intensive filters;
- *dynamic rate shaping filters* [13] are used to adapt the rate of compressed audio, video and digital images (MPEG, H261, MJPEG) to the dynamically varying rate constraints of the wireless environment. Unlike media selectors, rate shaping filters can shape flows to meet *any* bandwidth availability but are computationally intensive in comparison to media selectors; and
- *content and object-based filters* [6] provide a content-based approach for further video segmentation beyond motion vectors and low and high frequency DCTs. Objects or important scene changes in the video stream can be identified and transported over the wireless link. Such segmentation of video is being investigated as part of MPEG-4.

The choice of active filter style (e.g., discrete or continuous) is driven by the utility curves specified by applications using the adaptive-QOS API discussed in Section 2.1.

Mobiware active filtering control has been designed so that new active filters [7] only need to be dispatched when a session (i.e., a flow) is created or during QOS-controlled handoff. Once an application has selected an active filter it is dispatched (2) from a filter server resident at the mobile device or in the network. In this case the filter server interacts with the filter control object at the current access point to dispatch the selected active filter. The transfer of the Java bytecode class is achieved through the interaction of a network loader object to fetch the program code.

Once an active filter has been dispatched it needs to be bootstrapped and configured before it is operational in the data path and can execute on the Java Virtual Machine. Once a dispatch acknowledgement message has been received from the current access point's filter control object then the bootstrap process (3) is initiated. After bootstrapping is complete filter control configures the active filter by forwarding to it an application-specific filter-spec [3]. This filter-spec indicates the bandwidth required to support the base and enhancement layers for media selector filters or the operational bandwidth range for dynamic rate shaping filters. This allows the active filter to configure itself and complete the instantiation phase.

Active filters are autonomous and self-adjusting objects which are driven by the mobile soft-state refresh adapt mechanism discussed in Section 3.4. No interaction with filter control is required for active filters to adjust to changing network conditions. The advertised rate in the refresh message indicates whether there is sufficient resources to provide enhanced quality to the mobile. The mobility agent interacts with active filters at access points to achieve this. Active filters interpret the available

[7] While this is a design decision in mobiware, there is nothing restricting an application changing the style of active filter (viz. media selector, dynamic rate shaping and content-based) during the course of an existing session. This is a minor change to the source code distribution [22].

bandwidth in a filter dependent manner. Discretely adaptive filters use this rate to add or remove enhancement layers. In addition, the filter-spec carries the adaptation policy, which is used by active filters to meet the users specified adaptation preferences as discussed in Section 2.1.

## 4. Evaluation

To evaluate the programmable mobile network performance, we have 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 active filtering algorithms. Through the course of measurement, evaluation and re-design we have substantially improved the system performance of our initial baseline design.

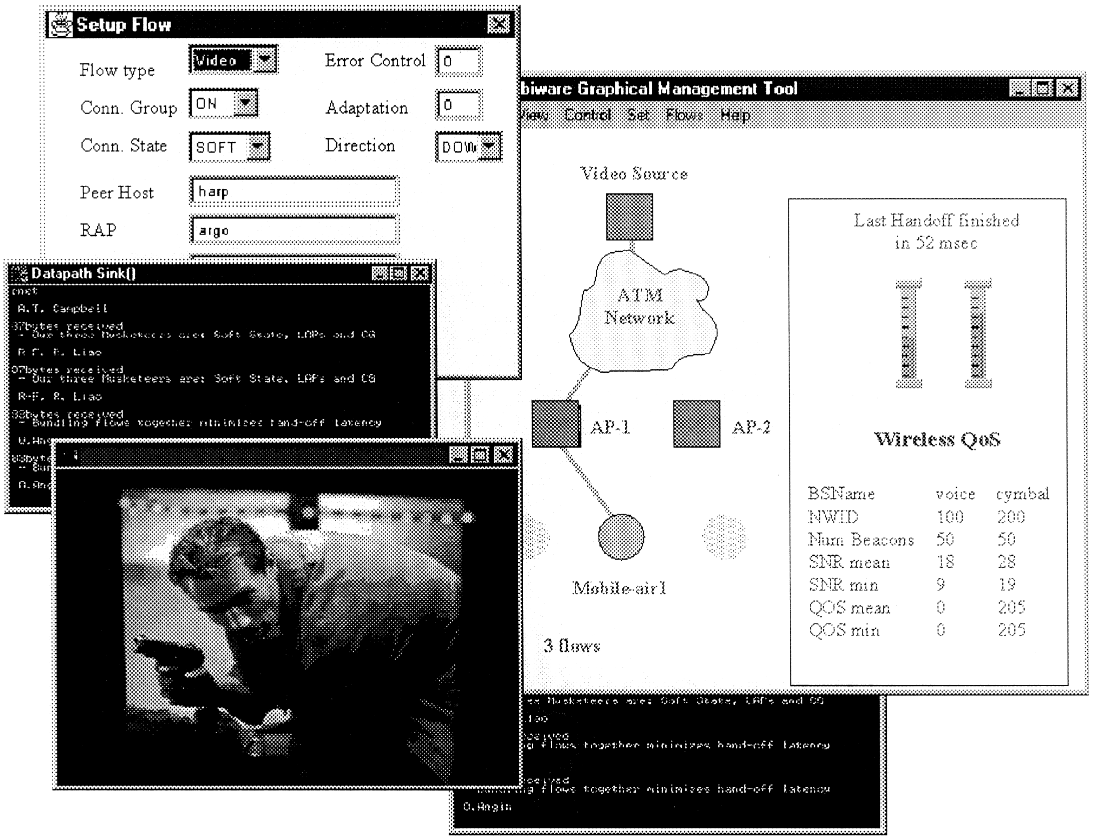
### *4.1. The experimental environment*

At the network level, mobinet provides wireless access to the Internet and comprises four ATM switches [8] (viz. ATML Virata, Fore ASX/100, NEC model 5, Scorpio Stinger switches) and four wireless access points as illustrated in Fig. 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 IP/ATM switched access 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 at 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 2.4 GHz band. A future version of mobinet will operate at 25 Mbps in the 5 GHz NII/SUPERNet unlicensed band. Mobile capable switches, access points and mobile devices are abstracted as programmable CORBA objects. Mobiware requires IIOP CORBA for mobile signalling and adaptation management. For the results provided in this paper the mobile devices and access points use the IONA's Orbix implementation of CORBA running under Windows NT and the mobile capable switches/routers use IONA's Orbix running under UNIX or other proprietary operating systems.

We have designed and implemented *mobiman* a Java-based management tool for mobile networking. Mobiman drives experiments, displays recorded statistics, and provides a 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 each 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 set up 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 the mobiman wireless management tool comprises flow information, quality of service measurements (e.g., signal level 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 Fig. 8. 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-air#1'' as illustrated in Fig. 8. In this example, the mobile-air#1 is running mobiman and the three flows correspond to the playout of the ''True Lies'' video and two low-resolution text windows. A flow setup panel appears in the top-left corner of Fig. 8.

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

[8] We modify the concept of switchlets [19] to provide an extended network for mobiware evaluation. Switchlets allow multiple virtual network elements to be operational within the same physical nodes. Three ATM switches (ATML 1, 2, and 3 as shown in Fig. 3) in our network are switchlets physically co-located at the same physical switch. For example, packets traversing three switchets 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 a different name space managing its own resources and controls connections independently from others switches.



Flow type: Video
Conn. Group: ON
Conn. State: SOFT
Error Control: 0
Adaptation: 0
Direction: DOW
Peer Host: harp
RAP: argo

```
cnet
 A.T. Campbell
87bytes received
 - Our three Musketeers are: Soft State, LAPs and CG
 R.-F. R. Liao
07bytes received
 - Our three Musketeers are: Soft State, LAPs and CG
 R-F. R. Liao
88bytes received
 - Bundling flows together minimizes hand-off latency
 O.Angin
```

Last Handoff finished in 52 msec

Wireless QoS

| BSName | voice | cymbal |
|---|---|---|
| NWID | 100 | 200 |
| Num Beacons | 50 | 50 |
| SNR mean | 18 | 28 |
| SNR min | 9 | 19 |
| QOS mean | 0 | 205 |
| QOS min | 0 | 205 |

Fig. 8. Mobiman, a Java-based management tool.

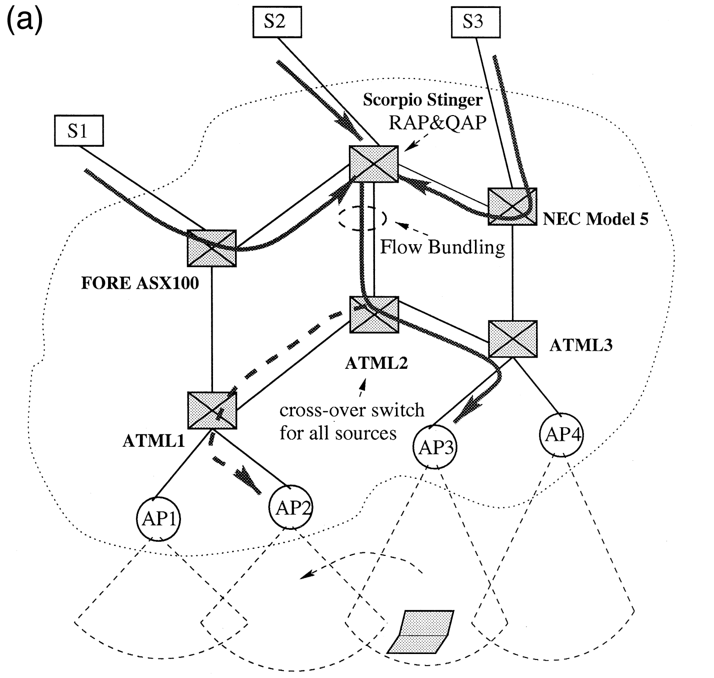
uses modular components called filters and filter graphs. Typically, a filter graph consists of a source filter that provides the system with multimedia data, a transform filter that performs data decompression and a rendering filter. Active Movie's filter graph has been enhanced with an appropriate mobiware static transport object to perform synchronization of flows during handoff, 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 mobiware programmable mobile network in supporting mobile multimedia communications. The use of CORBA for mobile signalling, wireless adaptation and mobile network programmability is a novel aspect of our implementation. CORBA objects run at the edges and in the mobile network to support wireless-QOS and mobile-QOS. An important aspect of our evaluation was to determine if such distributed object technology is viable in supporting mobile signalling and adaptation management.

### 4.2.1. Handoff analysis

An objective of this experiment was to measure the handoff latency and understand how the signalling 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) as shown in Fig. 9a and Fig. 9b. 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 was measured to be 171 ms. This measurement broke down into 102 ms for mobile registration and object binding, 30 ms for



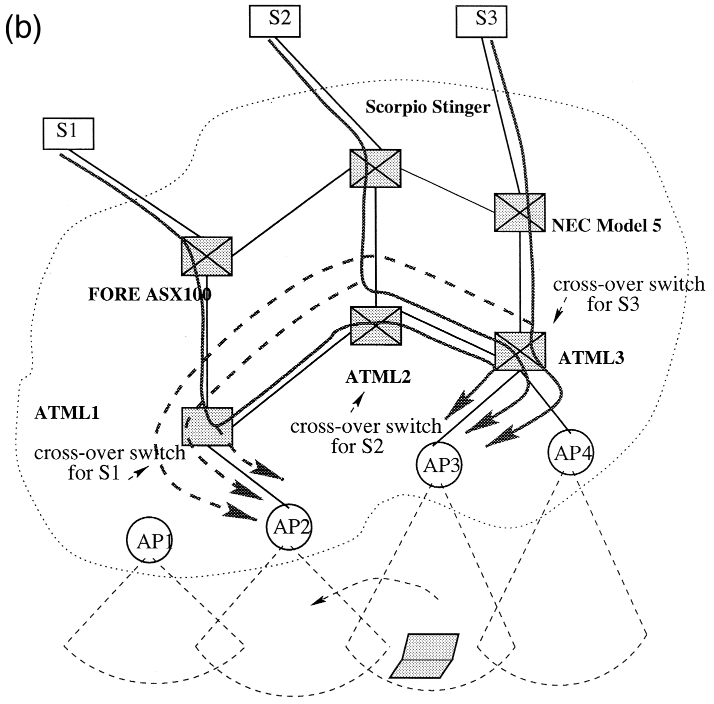


Fig. 9. a: Handoff with flow bundling ON. b: Handoff with flow bundling OFF.

wireless ATM connection setup and 39 ms 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 objects located at the access point were collapsed into a single CORBA object. This reduced the number of CORBA requests across the air-interface reducing the binding time from 102 ms to 42 ms. Collapsing objects in this manner reduced the overall handoff latency to 111 ms as illustrated in Fig. 10. Next, by reducing the number of CORBA RPCs the overhead between objects during handoff was further reduced. An RPC across the air-interface between the mobile and access point took an average of 15 ms to complete. The number of RPCs between the mobile and access point was reduced from four to two (viz. registration, and handoff request). Reducing the number of CORBA RPCs during handoff decreased the handoff latency by 28 ms to 83 ms as illustrated in Fig. 10.

The final enhancement to the baseline handoff code exploited the concept of caching object bindings. In order to eliminate the binding latency, we set up and cached bindings between remote CORBA objects prior to handoff being initiated; we call this *pre-binding*. All access points periodically broadcast their beacons that include address information, signal strength and available bandwidth resources at the access point. 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 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 availability of 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 bindings between all access point

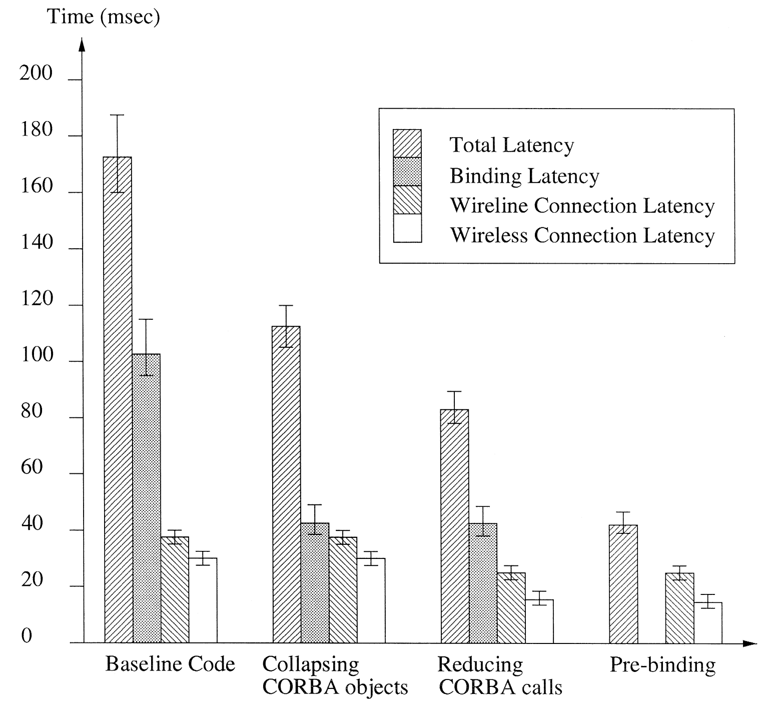


Fig. 10. Handoff latency measurement results.

objects in a domain and its associated mobility agent object. This final enhancement reduced the average handoff latency from 83 ms to 41 ms.

#### 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 (as illustrated in Fig. 9b) 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 a per-mobile QAP located at the Scorpio switch and re-routed during handoff via a single crossover switch located at the ATML2 switch as illustrated in Fig. 9a.

In this experiment, we vary the number of video streams transported to/from a mobile device from one to ten flows with bundling enabled and disabled and measure the handoff latency. We observe that the results from the baseline measurement highlight the performance improvement (i.e., speed up in handoff) gained using flow bundling techniques in the access network as illustrated in Fig. 11. As indicated in the figure, the benefit of flow bundling becomes more pronounced as the number of flows increases. For example, the handoff latency for two flows is 200 ms with flow bundling enabled and 250 ms when bundling is disabled. For ten flows with and without flow bundling enabled the handoff latency is 320 ms and 780 ms, respectively. The benefit of flow bundling reduces the handoff latency and, importantly, simplifies the state management of flows in the cellular access network. 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 Fig. 9a) the handoff latency converges whereas with flow bundling disabled the latency increases almost linearly as the number of flows increase. These results indicate the benefit of using flow bundling to reduce handoff latency and signalling overhead. This is mainly due to the fact that all interactions between objects during handoff

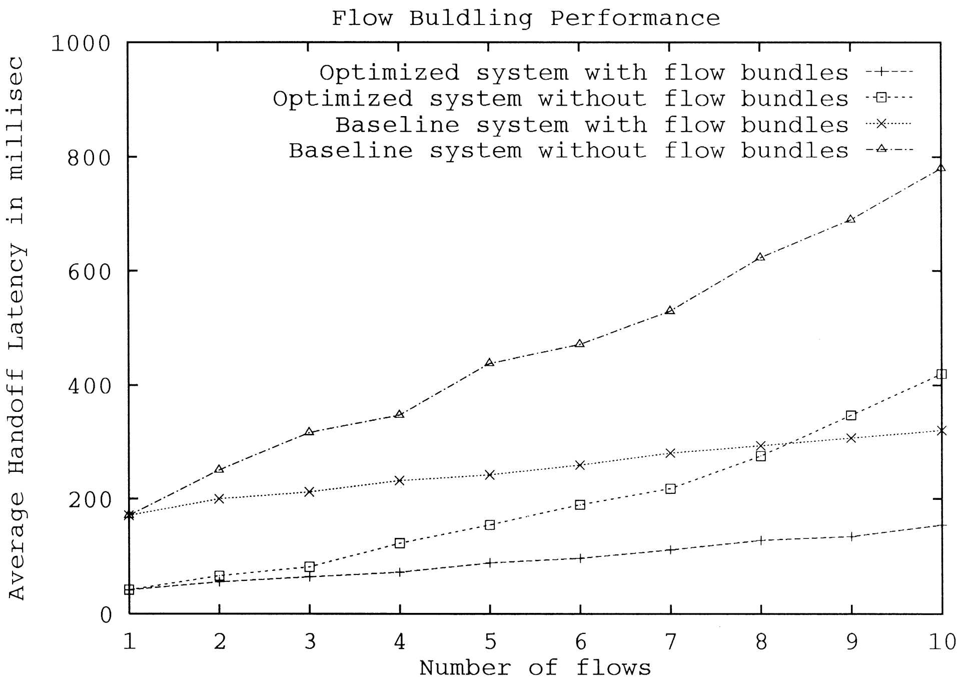


Fig. 11. Performance gain with flow bundling.

deal with aggregated signalling rather than per-flow signalling.

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 is 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 than 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. The GSMP enhancements 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 $\mu$s without aggregation and 511 $\mu$s with aggregation, showing a 40% improvement. With increasing number of flows, the total gain obtained by aggregation increases to 70% for ten flows (3907 $\mu$s vs. 1184 $\mu$s).

The baseline code was enhanced to support the optimization discussed in Section 4.2.1. In addition, the GSMP messaging between the switch server and switch hardware provided some incremental improvements as discussed above. Considering the enhanced code the handoff latency was reduced to 56 ms with bundling and to 67 ms without bundling for two flows showing a 16% improvement. In contrast, the handoff latency for ten flows was 155 ms and 420 ms with and without bundling, respectively, showing a 63% improvement.

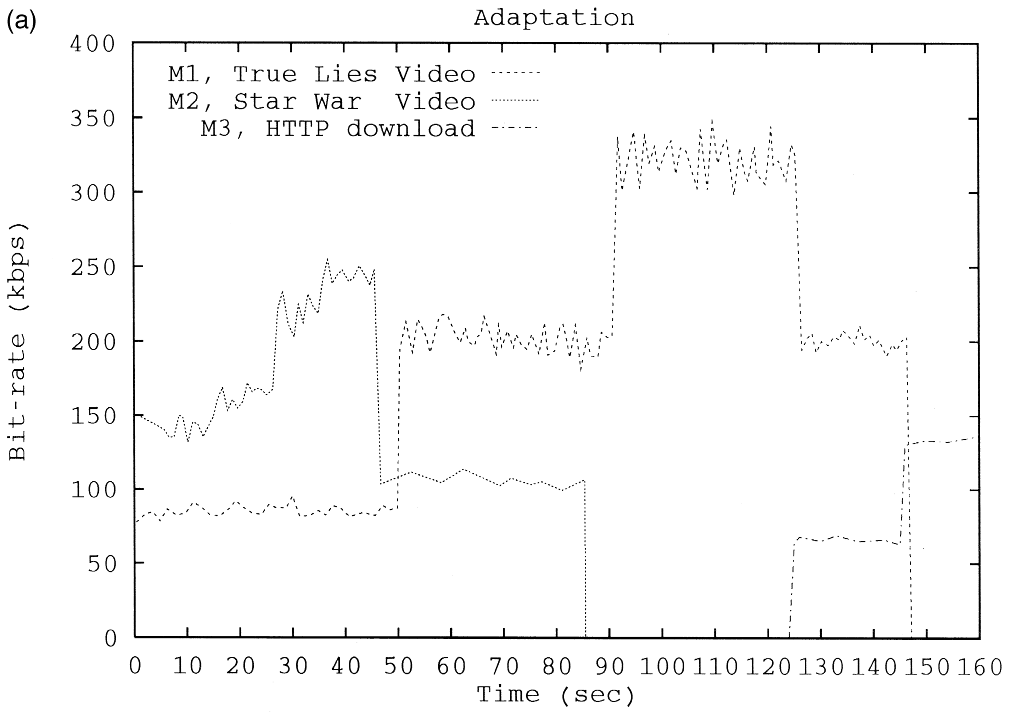
#### 4.2.3. Mobile soft-state analysis

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 an adaptation policy. In this experiment we present two scenarios that illustrate the benefit of mobile soft-state and QOS adaptation management in wireless and mobile environments.

The first scenario illustrated in Fig. 12a 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 receives a base layer (BL) at 80 Kbps and M2 receives a base and enhancement layer (E1) at 150 Kbps. Currently, the adaptive-QOS service gives priority to support the minimum bandwidth requirements of multi-resolution flows [7]. 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 M2 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 M2 from 150 Kbps to 250 Kbps. For the experiment, the maximum capacity of the air-interface is set to 330 Kbps and at approximately 45 s into the scenario the M2 video is adapted back to the base layer with FEC only. Resources released by M2 are consumed by M1 increasing its utility at 50 s [9] into the scenario. This situation remains constant until M2 handoffs to a new access point after 80 s allowing the access point to deliver another enhancement layer to M1. Note that mobile initiated adaptation to released resources (i.e., scaling up) is somewhat dependent on the refresh/probe interval. When the new mobile device M3 enters the cell around 120 s mobile device M1 explicitly scales back by dropping an enhancement layer. Towards the end of the scenario M3 probes and scales up to a better perceptible quality as M1 hands off to a new access point. At 140 s into the scenario, mobile device M3 sets up a new flow to access web services and downloads a GIF file at a rate of 70 Kbps scaling up to 135 Kbps.

[9] Mobile device M1 probes and adapts to the available bandwidth within a single refresh interval that is currently set to 10 s. 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. Currently, we are investigating the coupling of the probing and adaptation time scales to the application level adaptation policy [4].

(a)



(b)

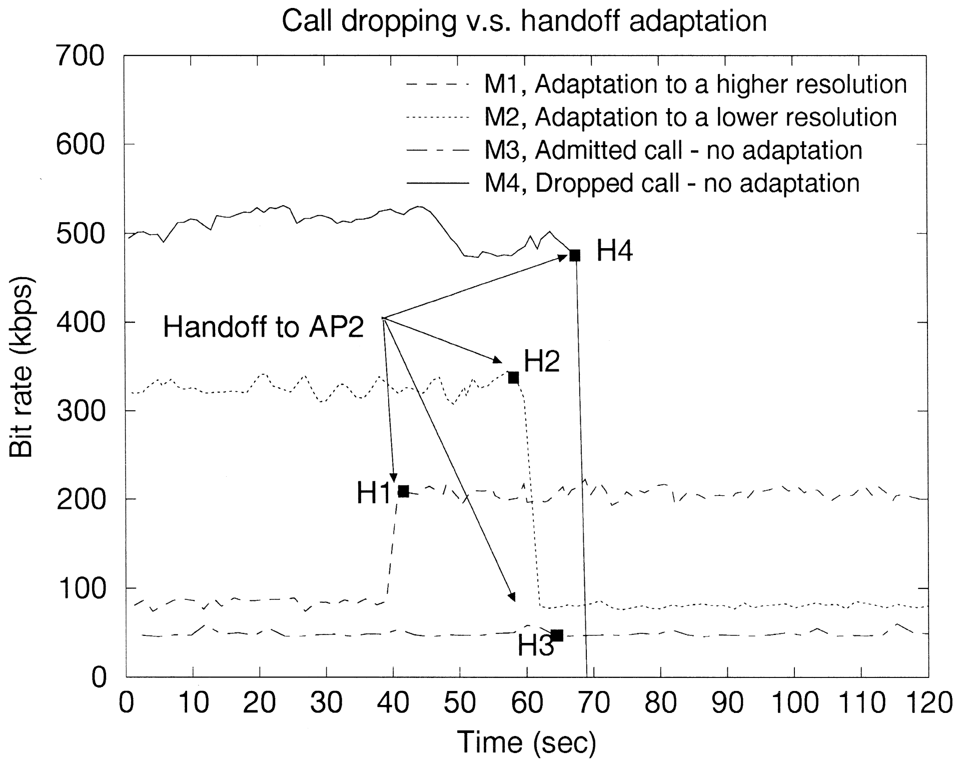


Fig. 12. a: QOS adaptation within a single cell. b: Handoff driven QOS adaptation.

In related work [4] 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 M1 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 handoff. In this experiment, mobile devices handoff 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 Fig. 12b. The type of adaptation that takes place after the handoff points, which are marked as H1 through H4, is illustrated in Fig. 12b. 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's adaptation policy is to only scale after handoff. At point H2 mobile device M2 hands off to the access point AP2 and is forced to scale 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.

### 4.2.4. *Active filtering analysis*

One of the key performance issues related to active filter technology is the time taken to dispatch, bootstrap and configure new filters over the wireless and wireline interfaces. Another important performance concern relates to the performance penalty paid by flows as they are filtered at access points. The amount of delay introduced by such operations as flows traverse active filters is dependent on the computational complexity of the filter, the additional overhead of calls to the Java Virtual Machine and the cost of derailing packets from the data path and filtering them.

The current source code distribution includes (i) media selector filters that drop either E1 (e.g., P pictures) and E2 (e.g., B pictures) frames based on the available resources; and (ii) a dynamic rate shaping filter which serves as a good comparison between the simplicity of frame dropping techniques versus more sophisticated processing in the media coding domain. Media selectors do not process the media as in the case with dynamic rate shaping filters. Therefore, media selector filter algorithms have the least impact among the mobiware active filters. Results from the implementation of a dynamic rate shaping algorithm indicate a maximum additional delay of the 3–4 frame inter-arrival times.

Our initial focus has been to analyze the impact of using active filters during QOS controlled handoff. We wanted to determine the additional delay incurred when using active filter technology and its impact on handoff. We investigated two scenarios related to handoff:

- the time taken to dispatch, bootstrap, configure and start (on the Java Virtual Machine) a new active filter transferring it over the air interface from a mobile device to the new access point; and
- the time taken to dispatch, bootstrap, configure and start (on the Java Virtual Machine) an existing active filter from the old access point to the new one transferring it over wireline network during QOS controlled handoff.

Table 1 summarizes the average measurements for each scenario. In the first scenario a number of active filters are dispatched over the air-interface at AP2 as illustrated in Fig. 7. In contrast, the second scenario an active filter is propagated between two access points (i.e., between AP2 and AP3) over three ATM switches using 25 and 155 Mbps links.

The results indicate that for the three types of active filters tested additional delays ranged from 38–160 ms for dispatching, bootstrapping and configuring active filters over the air interface. In the

Table 1
Measured delays for active filters

| Active filter type | No. lines of Java code | Code size (bytes) | Dispatch time (ms) air-interface | Dispatch time (ms) wireline |
|---|---|---|---|---|
| Media selector | 560 | 9980 | 38 | 4 |
| Dynamic rate shaping | 1480 | 18570 | 80 | 8 |
| Content-based filter | 2350 | 39500 | 160 | 17 |

case of propagating the active filters between two base stations over 3 ATM switches we measured that additional delay ranged between 4 and 17 ms. In both scenarios background traffic was active in the cell under measurement and cross traffic was active in the wireline access network. These figures look promising in the first instance. The result illustrates that transfer of active filters over the wireline network limits the latency introduced to QOS controlled handoff. Currently, the handoff duration between two access points connected through a single switch is 20 ms. In this case the use of simple active filters such as media selectors through wireline transfer does not significantly impact the handoff duration. However, using dynamic rate shaping and content-based filters over a 2 Mbps wireless link would introduce additional delays in the handoff latency. The impact of this would be eased for faster wireless links.

Media selectors are computationally simple and attractive filters, which can significantly reduce the data rate of audio, video and real-time images. Some of the positive attributes of media selectors are the small processing delays incurred at the access points during media scaling. Only headers of incoming frames are examined during media scaling. Instantiation time is also small due to the length of media selector's bytecode which is small in comparison with other filters such as dynamic rate shaping and content-based filters. The latter issue is important, because in cases when filters are transmitted through low-bandwidth air-interfaces, loading times can be comparable to the intervals required for handoff completion, which is of course an undesirable situation.

The dynamic rate shaping algorithm was observed to have a great impact on the reduction in size of the frames processed. In order to avoid certain syntactic complications like recoding the coded block patterns and recasting the DC prediction loops [13], the algorithm ensures that at least one DCT coefficient per block is retained in every picture.

As an example, Table 2 illustrates the picture sizes of the frames for a 10 picture GOP video sequence (true_lies.mpg) before and after rate shaping. On an average, the dynamic rate shaping algorithm was found to retain most parts of the I and the B frames but discarded mostly run-length codes from the P frames resulting in a near 50% picture size reduction. It can be seen that for the second frame in

Table 2
Relative size reduction of I, P and B pictures after dynamic rate shaping

| Frame # / sub-stream | Frame type | Frame size before DRS (bytes) | Frame size after DRS (bytes) | % shaping (reduction) |
|---|---|---|---|---|
| 1 BL | I | 2306 | 1718 | 25.5 |
| 2 E1 | P | 1044 | 405 | 61.2 |
| 3 E2 | B | 382 | 307 | 19.6 |
| 4 E2 | B | 406 | 322 | 20.7 |
| 5 E1 | P | 802 | 445 | 44.5 |
| 6 E2 | B | 539 | 398 | 26.1 |
| 7 E2 | B | 591 | 420 | 28.9 |
| 8 E1 | P | 956 | 494 | 48.3 |
| 9 E2 | B | 621 | 424 | 31.7 |
| 10 E2 | B | 578 | 408 | 29.4 |

Table 2 (i.e., P frame) almost 62% of the frame has been discarded by rate shaping, while the percent reduction remains the same for the I and the B frames. While, the media selector provides the mobile user with just three discrete operational levels by selecting one or more resolutions (E1 and E2) in a coarse operation, the dynamic rate shaping on the other hand, operates on a more continuous scale of bandwidth, which is dynamically specified in the adapt messages as part of the mobile soft-state protocol.

## 5. Discussion

We have analyzed the performance of mobiware's QOS controlled handoff, flow bundling, mobile soft state and active filtering algorithms. While the baseline code raised some initial performance concerns about the viability of using distributed CORBA 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 ms to 41 ms for the handoff of a single flow through two ATM switches making handoff through a single switch in the order of 20 ms.

While it is difficult to compare results from different testbeds running different signalling systems we highlight some measurements from comparable systems found in the literature for the purpose of qualitative comparison only. In Refs. [24] and [20] handoff latencies were measured to be 10 and 30 ms for a single flow through a single crossover switch respectively.

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 ms when flow bundling was enabled and 420 ms when disabled. This clearly shows the advantage of such aggregation techniques. Mobiware's flow bundling compares favorably to the literature. In Ref. [20] a handoff latency of 125 ms for ten flows using native ATM signalling code was reported.

Mobile soft-state also exploits aggregation techniques provided by flow bundling. This allows resource probing to be based on flow bundles rather than a single flow. In this paper QOS adaptation techniques clearly demonstrate the benefit of mobile soft-state in sharing resources among competing mobiles in a cell.

Mobiware's active filtering system provides application specific support for scaling media at access points providing a general mechanism for loading generic filters into the datapath on the fly. The systems use CORBA technology for filter control signalling and Java for data path processing. While it is clear that loading active filters into the access point provides dynamic and flexible means of adding new mechanisms to access points and in this case dealing with time-varying bandwidth and supporting QOS controlled handoff, the performance of the filtering systems is only currently viable when using the wireline network for moving filters around during handoff. This is because the mobiware handoff is in the order of 20 ms between two access points and extra delay could be added by the wireless transfer of filters impacts this for anything other than simple media selector filters for 2 Mbps wireless links. One solution we are investigating is to use ''filter caches'' at the access point. In this case filter control would first check the cache before requesting new filters from filter servers or before attempting to use air-interface to retrieve filters from the mobile device if resident there. Clearly, faster radios would ease the problem of retrieving filters from mobile devices. However, we believe that filters will be made available from servers with wireline attachments to the network; hence, alleviating the need to use wireless bandwidth to acquire filters.

## 6. Conclusion

In this paper we have discussed 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. Mobiware reduces the handoff dropping probability by using adaptation and media scaling techniques to allow new mobiles to enter a new cell. It also provides mechanisms to handle the complexity of handing off multiple flows to and from mobile devices using flow bundling and QOS controlled handoff.

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 signalling community [26] 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 networking. The mobiware technology we have developed over the last two years marks a considerable software effort. To our knowledge we are the first research group to build an open programmable mobile network [22]. Mobiware objects execute on the mobile devices, at the access points and on switches/routers exposing open APIs that can be programmed to support mobile signalling, QOS adaptation management and wireless transport. We observe that once the wireless and mobile APIs have been designed, the programming of new mobile network algorithms, e.g., QOS controlled handoff, flow bundling and mobile soft-state is straightforward engineering. The full source code distribution for mobiware v1.0 can be freely downloaded from Ref. [22] for experimentation.

## Acknowledgements

We would like to thank Aurel A. Lazar and the xbind team for providing the xbind broadband kernel on which mobiware is built. Also, we would like to thank the following colleagues for their major contributions 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 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.

## References

[1] O. Angin, A.T. Campbell, M.E. Kounavis, R.R.-F. Liao, Open programmable mobile networks, Proc. 8th Int. Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV), Cambridge, England, July 1998.

[2] O. Angin, A.T. Campbell, M.E. Kounavis, R.R.-F. Liao, Enabling the creation, control and management of adaptive wireless services in programmable mobile networks, Technical Report, Center for Telecommunications Research, June 1998.

[3] A. Balachandran, A.T. Campbell, M.E. Kounavis, Active filters: delivering scalable media to mobile devices, Proc.7th Int. Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV), St. Louis, May 1997.

[4] R.R.-F. Liao, A.T. Campbell, On programmable universal mobile channels in a cellular Internet, Proc. ACM/IEEE Int. Conf. on Mobile Computing and Networking (MOBICOM'98), Dallas, TX, October 1998.

[5] G. Bianchi, A.T. Campbell, A programmable MAC, Proc. Int. Conf. on Universal Personal Communications (ICUPC), Florence, October, 1998.

[6] P. Bocheck, S.-F. Chang, Content-based modeling for scalable variable bit rate video, Proc. 6th Int. Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV), Japan, April 1996.

[7] A.T. Campbell, D. Hutchison, C. Aurrecoechea, Dynamic QOS management for scalable video flows, Proc. 5th Int. Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV), Durham, New Hampshire, 1995.

[8] A.T. Campbell, G. Coulson, QOS adaptive transports: delivering scalable media to the desktop, IEEE Network Magazine, March 1997.

[9] A.T. Campbell, Mobiware: QOS-aware middleware for mobile multimedia communications, Proc. 7th IFIP Int. Conf. on High Performance Networking, White Plains, NY, April 1997.

[10] D. Clark, D. Tennenhouse, Architectural considerations for a new generation of protocols, Proc. SIGCOMM '90, September 1990.

[11] M.S. Corson, A.T. Campbell, Toward supporting quality of service in mobile ad hoc networks, Work in progress session, 1st IEEE Conf. on Open Architectures and Network Programming (OPENARCH), San Francisco, CA, April 1998.

[12] The Daedalus/BARWAN project, UC Berkeley, http://daedalus.cs.berkeley.edu/index.html.

[13] A. Eleftheriadis, D. Anastassiou, Meeting arbitrary QOS constraints using dynamic rate shaping of coded digital video, Proc. 5th Int. Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV), Durham, New Hampshire, April 1995.

[14] R.H. Katz, Adaptation and mobility in wireless information systems, IEEE Personal Communications Magazine 1 (1) (1994) .

[15] A.A. Lazar, Programming telecommunication networks, IEEE Network, October 1997.
[16] K. Lee, Adaptive network support for mobile multimedia, Proc. MOBICOM'95, Berkeley, CA, November 1995.
[17] S.-B. Lee, A.T. Campbell, INSIGNIA: inband signaling for mobile ad hoc networks, Proc. 5th Int. Workshop on Mobile Multimedia Communications (MoMuC), Berlin, Germany, October 1998.
[18] S. Lu, K.-W. Lee, V. Bharghavan, Adaptive service in mobile computing environments, Proc. 5th IFIP Int. Workshop on Quality of Service (IWQOS), New York, May 1997.
[19] J.E. Merwe, I. Leslie, Switchlets and dynamic virtual ATM networks, in: A.A. Lazar, R. Saracco, R. Stadler (Eds.), Integrated Network Management V, Chapman and Hall, New York, 1997.
[20] P. Mishra, Implementation and experimental evaluation of mobility-enhanced ATM signaling, OPENSIG, Fall '97 Workshop on Open Signaling for ATM, Internet and Mobile Networks, New York, October 1997.
[21] Panel on QoS in the Next Generation Mobile Internet: What is Feasible? Chaired by A.T. Campbell, 3rd Annual ACM/IEEE Int. Conf. on Mobile Computing and Networking (MobiCom '97), Budapest, October 1997.
[22] Mobiware v1.0 Source Code Distribution: http://comet.columbia.edu/mobiware.
[23] M. Naghshineh, M. Willebeek-LeMair, End-to-end QOS provisioning in multimedia wireless/mobile networks IEEE Network, March 1997.
[24] J. Naylon, Radio handover measurements, OPENSIG Spring '97 Workshop on Open Signaling for ATM, Internet and Mobile Networks, Cambridge, England, April 1997.
[25] Object Management Group (OMG), The Common Object Request Broker: Architecture and Specification, Rev. 2.2, February 1998.
[26] http://comet.columbia.edu/opensig/.
[27] J. Porter, A. Hopper, D. Gilmurray, O. Mason, J. Naylon, A. Jones, The ORL radio ATM system, architecture and implementation, ORL Technical Report, 1995.
[28] M. Satyanarayanan, Mobile information access, IEEE Personal Communications Magazine 3 (1) (1996) .
[29] D.L. Tennenhouse, J.M. Smith, W.D. Sincoskie, D.J. Wetherall, G.J. Minden, A survey of active network research, IEEE Communications Magazine 35 (1997) 80–86.
[30] xbind Broadband Kernel code distribution: http://comet.columbia.edu/xbind.
[31] N. Yeadon, F. Garcia, D. Hutchison, D. Shepherd, Filters: QoS support mechanisms for multipeer communications, IEEE Journal on Selected Areas in Computing (JSAC) on Distributed Multimedia Systems and Technology, May 1996.
[32] B. Zenel, D. Duchamp, A general proxy filtering mechanism applied to the mobile environment, Proc. MOBICOM '97, Budapest, Hungary, September 1997.

**Andrew T. Campbell** (www.comet.columbia/ ~ campbell) joined the E.E. faculty at Columbia as an Assistant Professor in January 1996 from Lancaster University where he conducted research in multimedia communications as a British Telecom Research Lecturer. Before joining Lancaster University, Dr. Campbell worked for 10 years in industry focusing on the design and development of network operating systems, communication protocols for packet-switched and local area networks, and tactical wireless communication systems. Dr. Campbell is a member of the COMET Group at Columbia's Center for Telecommunications Research where he is conducting research in wireless media systems (www.comet.columbia.edu/wireless). His current research interests include the development of programmable mobile networks, cellular IP networks and programmable router technologies.

**Michael E. Kounavis** (www.comet.columbia/ ~ mk) is a Ph.D candidate and Graduate Research Assistant at COMET Group, Columbia University. He received his Diploma in Electrical and Computer Engineering from the National Technical University of Athens, Greece (NTUA) in 1996 and his M.Sc. degree from Columbia in 1998. His current research focuses on the technology of programmable virtual networks. Over the past two years he has been actively involved in mobile network programmability and active transport over wireless links.

**Raymond R.-F. Liao** (www.comet.columbia/ ~ liao) joined the COMET Group, Columbia University in 1996 as a Ph.D. student and Graduate Research Assistant. Before that, he worked at Newbridge Networks, Canada for 3 years on ATM product performance analysis and traffic management. He received the M.A.Sc. degree in fault-tolerant ATM switch design and queueing analysis from the Dept. of Electrical and Computer Engineering, University of Toronto, Canada, in 1993, and the Bachelor degree from Huazhong University of Science and Technology, China, in 1990. His current research focuses on realizing adaptive-QOS in wireless/mobile multimedia networks with middleware methodologies including distributed computing and network economics.