Design, Implementation and Evaluation of Programmable Handoff in Mobile Networks

Michael E. Kounavis[1], Andrew T. Campbell[1], Gen Ito[2], and Giuseppe Bianchi[3]

[1]Comet Group, Center for Telecommunications Research
Columbia University, New York, NY 10027
{mk, campbell}@comet.columbia.edu

[2]NEC Corporation, Japan
itogen@sns.abk.nec.co.jp

[3]University of Palermo, Italy
bianchi@morgana.elet.polimi.it

Abstract

We describe the design, implementation and evaluation of a programmable architecture for profiling, composing and deploying handoff services. We argue that future wireless access networks should be built on a foundation of open programmable networking allowing for the dynamic deployment of new mobile and wireless services. Customizing handoff control and mobility management in this manner calls for advances in software and networking technologies in order to respond to specific radio, mobility and service quality requirements of future wireless Internet service providers. Two new handoff services are deployed using programmable mobile networking techniques. First, we describe a 'multi-handoff' access network service, which is capable of simultaneously supporting multiple styles of handoff control over the same physical wireless infrastructure. Second, we discuss a 'reflective handoff' service, which allows programmable mobile devices to freely roam between heterogeneous wireless access networks that support different legacy signaling systems. Evaluation results indicate that programmable handoff architectures are capable of scaling to support a large number of mobile devices while achieving similar performance to that of native signaling systems.

1. Introduction

As the wireless Internet rolls out over the next several years there will be an increasing demand for new mobile devices, services and radios that can meet the needs of mobile users. Recent trends indicate that a wide variety of mobile devices will be available each requiring specialized services and protocols. Competition between emerging Wireless Internet Service Providers (WISPs) is likely to hinge on the speed at which one service provider can respond to new market demands over another. Existing mobile and wireless networks have limited service creation environments, however. Typically, service creation is a manual, ad hoc and costly process in wireless networks. Mobile network services and protocols cannot be easily extended or modified because they are generally implemented using dedicated firmware, embedded software, or constitute part of the low-level operating system support. For example, it is difficult to dynamically modify the handoff prioritization strategy in PCS networks or to introduce new handoff control algorithms (e.g., mobile assisted handoff) in wireless LANs. In addition, the incompatibility of signaling systems and physical layer radio technologies prevents mobile devices from roaming between heterogeneous wireless networks. Such factors limit the vision of seamless mobility. For example, a CDMA cellular phone cannot be easily connected to an IEEE 802.11 wireless LAN. Furthermore, access network protocols make specific assumptions about the capability of mobile devices. For example, both Mobile IP [2] and Mobile ATM [3] approaches assume that handoff control is located at the mobile device. Such mobile-controlled handoff schemes may not be suitable for many low-power devices that are incapable of continuously monitoring channel quality measurements.

These observations call for new communication methodologies and software technologies for mobile networks. Software engineering has progressed to the point where systems and standards can be used for implementing

platform-independent, component-based, distributed software. Such advances have enabled the development of programmable [4] and active [5] network toolkits for the deployment of new services. Programmable networks [4] separate the communications hardware from the control software, allowing the modeling of hardware resources using open programmable interfaces. In this manner third-party software providers can enter the market for telecommunications software. Work on software radios [6, 7] has shown that wireless physical layers can be created dynamically by introducing code into programmable base stations with wide-band tunable front-ends. In this paper, we focus on the control plane and address the problem of making handoff programmable for the introduction of new services. We argue that the creation, deployment and management of new wireless and mobile network services should be automated using programmable networking techniques.

Programmable handoff services can be deployed using general-purpose computers or specialized equipment, and can be used to control handoff for a wide range of mobile network architectures. We present a programmable architecture for profiling, composing and deploying handoff services. The programmable handoff architecture discussed in this paper consists of a binding model and a service creation environment. The binding model describes how collections of distributed algorithmic components can be combined in order to compose programmable handoff services. The service creation environment allows network architects to design and dynamically deploy handoff services taking into account user, radio and environmental factors. Code reuse eases the process of service creation allowing wireless networks to offer different types of services with simple software upgrades. To make handoff programmable we have identified a number of common features among different styles of handoff. These features provide a basis for the formulation of a binding model and a set of building blocks for composing handoff services on-demand.

In this paper, we introduce two new handoff services that highlight the flexibility and benefit of an open programmable mobile networking approach. First, we show how a 'multi-handoff' access network service can simultaneously support different styles of handoff control over the same wireless access network. This programmable approach can benefit emerging WISPs who will need to be able to satisfy the mobility management needs of a wide range of mobile devices from cellular phones to more sophisticated palmtop and laptop computers. We enhance the Mobiware [12] testbed to support multi-handoff and show that programmable handoff services can perform uniformly well under diverse mobility conditions. Second, we describe a 'reflective handoff' service that allows access networks to dynamically inject signaling systems into mobile devices before handoff. Thus, mobile devices can seamlessly roam between wireless access networks that support radically different mobility management systems. We show through experimentation how mobile devices can use reflective handoff to roam between an enhanced Mobiware [12] wireless access network and a Cellular IP [13] wireless access network.

This paper is structured as follows. In Section 2 we discuss the related work in the area of programmable mobile networks. In Section 3, we provide a description of the programmable handoff architecture. Following this, in Sections 4 and 5 we present the implementation and evaluation of our architecture, respectively, focusing on the deployment of the multi-handoff and reflective handoff services. In Section 6, we provide some concluding remarks.

2. Related Work

Programmable mobile networks represent an emerging area of research. In [6, 7, 14, 15] the programmability of the physical layer is addressed based on digital signal processing techniques and wide-band analog-to-digital conversion. Software radios allow access points and mobile devices to implement physical layer functionality (e.g., modulation, equalization, channel coding) in software. The degree of programmability software radios offer depends on the type of technology used. Software radios can be implemented using Application Specific Integrated Circuits (ASICs), Field Programmable Gate Arrays (FPGAs), Digital Signal Processors (DSPs) and general-purpose processors. Among the many software radio technologies Virtual Radios [7] perform all digital signal processing in user space using off-the-shelf PCs and workstations enabling the composition of portable signal processing systems. Another class of software radios called Cognitive Radios [15] employ model-based reasoning about users, multimedia content and communications context in order to achieve a specified-level of performance.

In [8,12,16,17] the problem of programming quality of service (QOS) support into mobile networks is addressed. A programmable MAC framework [8] supports adaptive real-time applications over time-varying and bandwidth-limited networks allowing mobile applications to map their service requirements into groups of 'bearer'

traffic classes supported by a programmable scheduler. The programmable MAC is extensible supporting the introduction of new adaptive services on-demand. In [12] the Mobiware Toolkit, a programmable mobile middleware platform is presented that captures the requirements of adaptive mobile multimedia applications in terms of bandwidth-utility curves, adaptation policies and session preferences. Mobiware utilizes distributed CORBA objects to exert quality of service control over wireless access networks and active transport objects to introduce new adaptive algorithms into mobile devices, access points and mobile-capable switches/routers.

Active networking services for wireless and mobile networks are discussed in [18]. Active networking services address the bandwidth limitations and physical radio layer impairments characterizing wireless networks by allowing applications to install filters at wireless gateways. Active networking solutions include adaptive FEC protocols and application-specific filtering. The ARRCANE project [19] is investigating how data packets should be routed in wireless ad hoc networks using active networking techniques. Active ad-hoc networks support the coexistence of different routing protocols operating in parallel, as well as the capability of wireless ad-hoc nodes to switch from one routing protocol to another.

While the research community has addressed the programmability of the physical layer and quality of service control in mobile networks, little work has been done on service creation. In this paper, we investigate technology for introducing new services in mobile and wireless networks. Our methodology focuses on a framework for making handoff programmable. Handoff represents an important service in wireless networks characterizing the capability of access networks to respond to mobile user requirements. We believe that the capability of making mobile networks programmable will help accelerate the deployment of new services in mobile and wireless networks and speed innovation.

3. Programmable Handoff Architecture

An architecture that supports the profiling, composition and deployment of programmable handoff services is illustrated in Figure 1. The architecture comprises a binding model and a service creation environment. The binding model describes how distributed objects can be combined to form programmable handoff services on-demand. The service creation environment allows network architects to design and dynamically deploy handoff services taking into account user, radio and environmental factors.

The service creation environment offers interfaces for the dynamic introduction and modification of network services through transportable code. Programmable handoff services are implemented as collections of distributed objects. In our framework, middleware technologies enable network-wide system programmability and interoperability, separating the definition of programmable handoff objects from their implementation. Programmable handoff objects expose control interfaces allowing the creation of bindings at run time. Binding is the process through which an object obtains a reference to another object in order to request its services. An object reference can be described in many different ways. For example, an object reference can be represented by a hostname where the object is activated, and by a TCP/IP port number where the object listens for service requests. Many software technologies support binding including distributed systems platforms (e.g., CORBA, DCOM, Java RMI) and localized mechanisms (e.g., dynamic link libraries).

The service creation environment comprises a profiler and a set of service controllers. The profiler drives the service creation process creating representations of programmable handoff services over a defined access network topology. As illustrated in the figure, the profiler interacts with a set of service controllers using a well-defined profile scripting language to create or modify handoff services. Service controllers compile profiling scripts, resolve object bindings, and create handoff control and mobility management systems. Modification of programmable handoff services takes place after bindings are removed or objects are deleted.

Service creation requires a comprehensive architectural model for binding distributed objects to create new services. Objects are selected from component repositories. A binding model characterizing the composition of programmable handoff services is shown in Figure 1. The binding model comprises a handoff control model, a mobility management model and a software radio model. Service controllers realize each model separately. In this manner, each model can use different binding mechanisms. For example, handoff control systems can be programmed using CORBA [26] distributed systems technology (as discussed in Section 4), whereas MAC layers can be composed using dynamic link libraries (DLLs), as discussed in [9]. The binding model supports the

separation of handoff control from mobility management, the decomposition of the handoff control process and the programmability of the physical and data link layers. The handoff execution interface separates handoff control from mobility management. Handoff adapters integrate handoff control systems with mobility management services. In what follows, we describe each component of the binding model in detail.

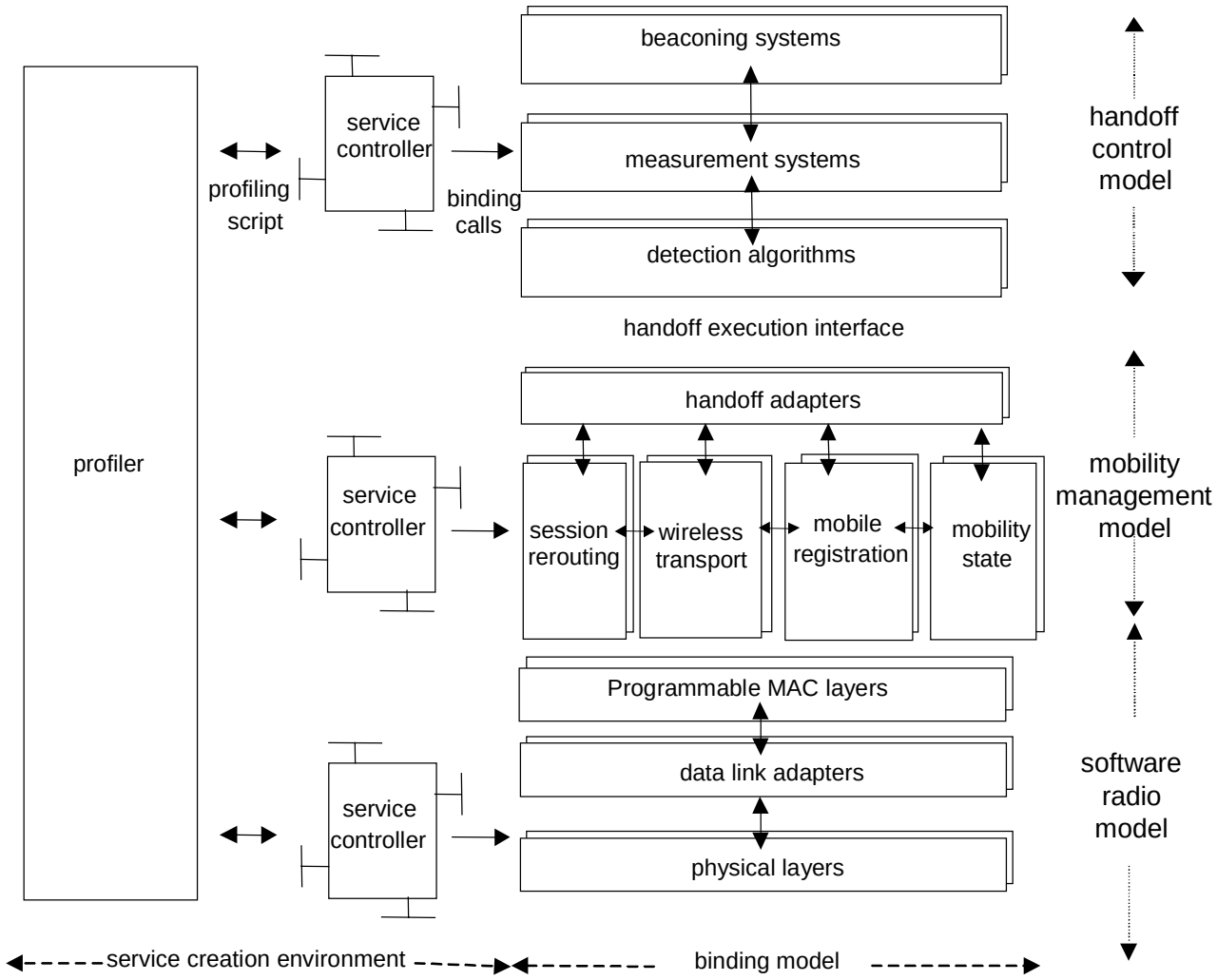


Figure 1: Programmable Handoff Architecture

3.1 Handoff Control Model

A handoff control model separates the algorithms that support beaconing, channel quality measurement and handoff detection, as illustrated in Figure 1. Typically, these functions are supported as a single 'monolithic' software structure in existing mobile systems. By separating the handoff detection from wireless channel quality measurements, we allow for new detection algorithms to be dynamically introduced in access networks and mobile devices. For example, detection algorithms specific to overlay networks can be introduced into mobile devices allowing them to perform vertical handoffs, or detection algorithms specific to micro-cellular networks can be selected to compensate against the street-corner effect [20]. By separating the collection of wireless channel quality measurements from the beaconing system, mobile networks can support different styles of handoff control over the same wireless infrastructure. For example a WISP may want to offer a network-controlled handoff service (e.g., supported in AMPS [21] cellular systems) for simple mobile devices, mobile-assisted handoff service (e.g., as in GSM [21]) for more sophisticated mobile devices involved in the process of measuring channel quality and mobile-controlled handoff service (e.g., the handoff scheme considered by the Mobile IP Working Group) for more sophisticated laptop or palmtops mobile computers.

The handoff control model comprises the following services:

- *detection algorithms*, which determine the most suitable access points that a mobile device should be attached to. Wireless access points can be selected based on different factors including channel quality measurements, resource availability and user-specific policies [22]. A mobile device can be attached to one or more access points at any moment in time.
- *measurement systems*, which create and update handoff detection state. By handoff detection state we mean the data used by detection algorithms to make decisions about handoff. Detection algorithms and measurement systems use the same representation for handoff detection state.
- *beaconing systems*, which assist in the process of measuring wireless channel quality. Programmable beacons can be customized to support service-specific protocols like QOS-aware beaconing [12] or reflective handoff as discussed in Section 4.

3.2 Mobility Management Model

The mobility management model reflects the composition of services that execute handoff, as illustrated in Figure 1. We adopt a generalized architectural model that is capable of supporting the design space of different mobile networking technologies. To program mobility management systems one needs to be able to introduce new forwarding functions at mobile capable routers/switches (e.g., Cellular IP [13] or HAWAII [23] forwarding engines) as well as distributed controllers that manage mobility (e.g., Mobile IP foreign agents). The mobility management model discussed in this paper is limited to supporting handoff services only. Other mobility management functionality (e.g., location, fault and account management) typically found in mobile networks will be considered as future work.

We identify the following services as part of the handoff execution process:

- *session rerouting mechanisms*, which control the datapath in access networks in order to forward data to/from mobile devices through new points of attachment. Rerouting services may include admission control and QOS adaptation for the management of wireless bandwidth resources.
- *wireless transport objects*, which interact with the physical and data link layers in mobile devices and access points to transfer active sessions between different wireless channels. A channel change may be realized through a new time slot, frequency band, code word or logical identifier. Transport objects can provide valued-added QOS support (e.g., TCP snooping [24]).
- *mobile registration*, which is associated with the state information a mobile device exchanges with an access network when changing points of attachment.
- *mobility state*, which can be expressed in terms of a mobile device's connectivity, addressing and routing information, bandwidth and name-space allocations and user preferences.

3.3 Software Radio Model

The software radio model defines the composition of physical and data link layer services, as illustrated in Figure 1. The software radio model supports functions such as the dynamic assignment of channel locations and widths, and the selection of modulation and coding techniques used on each channel. Software radios allow mobile devices to dynamically 'tune' to the appropriate air-interface of the serving access network, while roaming between heterogeneous wireless environments. MAC layer protocols can be made programmable [8] allowing for services that support different QOS requirements. Physical and data link layer modules can be implemented in various ways [6, 7, 14, 15]. Data link 'adapters' separate data link layer modules from the lower physical layer components. For example, data link adapters allow programmable MAC protocols to operate on top of any type of channel coding or modulation scheme, as discussed in [9].

3.4 Handoff Execution Interface

A handoff execution interface, illustrated in Figure 1, separates handoff control from mobility management. Handoff control and mobility management systems are implemented as separate programmable architectures. By hiding the implementation details of mobility management algorithms from handoff control systems the handoff detection state (e.g., the best candidate access points for a mobile device) can be managed separately from handoff execution state (e.g., mobile registration information). This software approach can be used to enable inter-system handoff between different types of wireless access networks. The basic idea behind realizing inter-system handoff is that the same detection mechanisms operating in mobile devices and access networks can interface with multiple types of mobility management architectures that operate in heterogeneous access networks (e.g., Mobile IP [2], Cellular IP [13], Mobiware [12] and HAWAII [23] access networks). Handoff control systems issue a number of generic service requests through the handoff execution interface, which mobility management systems execute according to their own programmable implementation. For example**,** a generic 'pre-bind' method call to a candidate access point would be executed by establishing a signaling channel in a Mobiware architecture [12], or by joining a multicast group specific to a mobile device and buffering packets in a Daedalus/BARWAN [24] architecture. In one extreme case where the location of the handoff control system is at the mobile device, different mobility management protocols can be dynamically loaded into mobile devices allowing them to roam between heterogeneous access networks in a seamless manner.

Examples of handoff execution methods include:

- *handoff* methods, which map down to mobility management services that execute handoff (e.g., register the care-of address of a mobile device with its home agent).
- *pre-bind* methods, which initiate 'priming actions' at candidate access points associated with a mobile device (e.g., load an active filter for transport adaptation [12], start buffering packets, etc.).
- *configure* methods, which bind new signaling systems or delete existing ones (e.g., replace the mobile device's Mobiware control plane with a Cellular IP one during reflective handoff).

We observe that it is difficult to consider a single handoff execution interface that is capable of encompassing all existing and future wireless systems. Rather, it is more likely that a handoff execution interface will support particular 'family' of wireless technology. In this case, network architects can select execution methods that satisfy the set of handoff control and mobility management systems which they wish to program.

3.4 Handoff Adapters

Programmable access networks allow different styles of handoff control (e.g., mobile controlled, mobile assisted and network controlled handoff) to seamlessly share the same mobility management services offered. However, seamless integration of handoff control systems with mobility management services is difficult to realize. In our framework, we introduce the concept of 'handoff adapters' which represent an intermediate layer of distributed objects that can be used for integrating the handoff control model with the mobility management model. Handoff adapters represent a set of distributed objects that serve as the 'glue' between handoff control systems and mobility management services. Handoff adapters and mobility management services collectively implement handoff execution algorithms. Handoff adapters can be centralized (i.e., running in a single host or network node) or distributed. In the distributed case, handoff adapters are deployed at mobile devices, access points or mobile capable routers/switches.

Handoff adapters are an important part of our programmable handoff architecture. First, handoff adapters control the handoff execution process. Handoff adapters invoke mobility management services in an order that is specific to the handoff style being programmed. Mobility management services (i.e., session rerouting, wireless transport, mobile registration and mobility state management services) are invoked as part of the handoff execution process. For example, in a forward mobile controlled handoff, an adapter would invoke a radio link transfer service before session rerouting. In the backward, mobile assisted handoff the order of this execution would be reversed. Each handoff style uses a separate adapter. To invoke mobility management services, adapters distribute method invocations to the network nodes or hosts where mobility management services are offered. Handoff is usually detected at a single host or network node (e.g., a mobile device, access point, or mobile capable router/switch). In

contrast, mobility management services can be offered at multiple hosts or network nodes inside a wireless access network (e.g., at wireless access points or by mobility agents in the network).

Second, handoff adapters 'translate' the handoff execution interface to the interfaces supported by specific mobility management architectures. In this manner, adapters hide the heterogeneity of mobility management architectures enabling inter-system handoffs. Adapters may interact with distributed mobility agents, databases supporting mobile registration information, or open network nodes to realize handoff in many different ways. For example, a 'Mobile IP' adapter would interact with a Mobile IP scheme to support connectivity for mobile devices (e.g., acquiring a care-of address through DHCP and registering the care-of address with a home agent). A Cellular IP or HAWAI 'adapter' would transmit control messages for establishing mobile-host specific routing entries in the access network. The role of adapters is further discussed in the Section 4, where we describe distributed algorithms for programmable handoff.

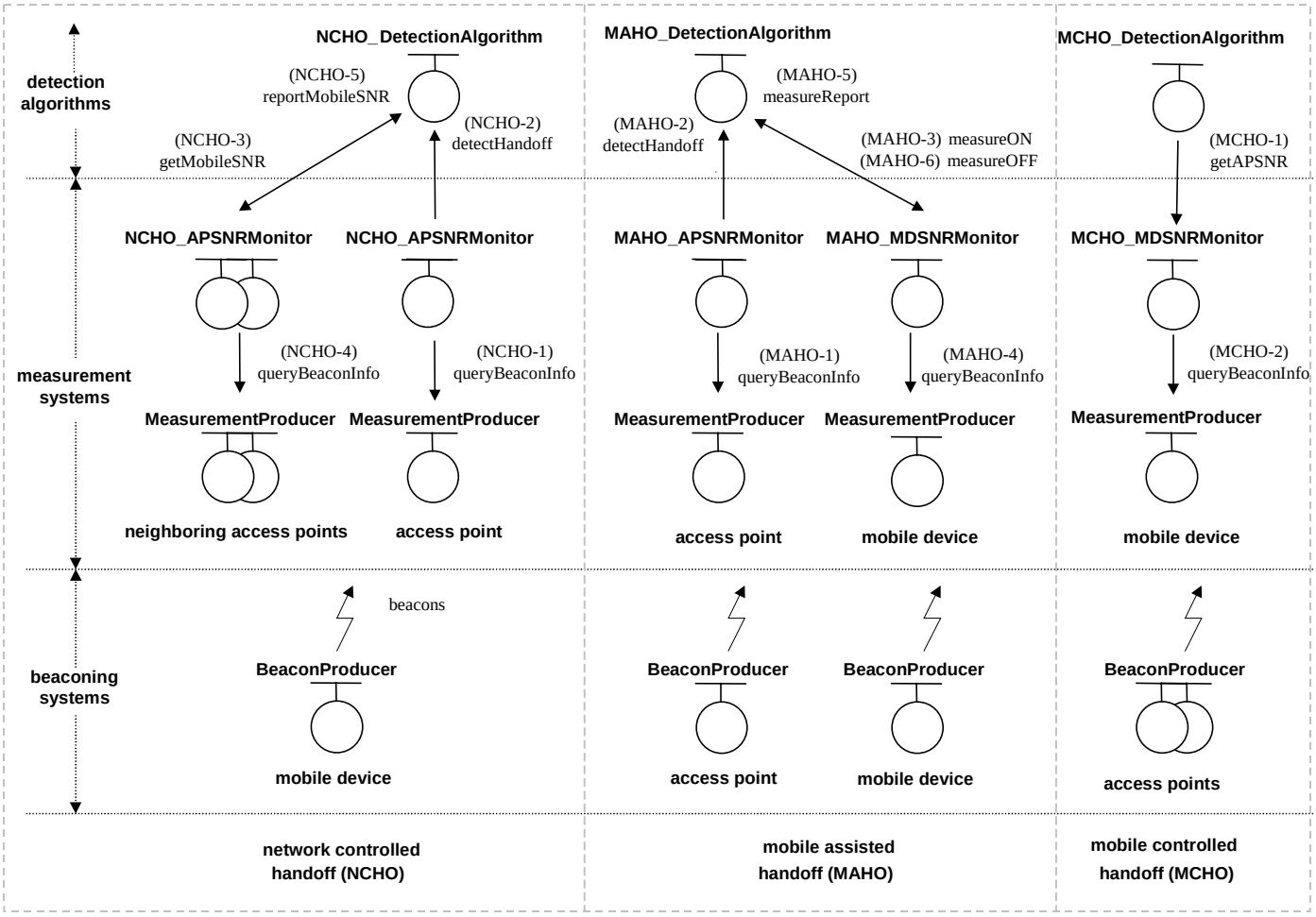


Figure 2: Implementation of the Handoff Control Model

4. Design and Implementation

We have designed and implemented two new handoff services based on the programmable handoff architecture. A multi-handoff access network service simultaneously supports three styles of handoff control over the same physical wireless access network that are commonly found in mobile networks: *Network Controlled HandOff (NCHO)*, *Mobile Assisted HandOff (MAHO)* and *Mobile Controlled Handoff (MCHO).* In addition, a reflective handoff service allows mobile devices to reprogram their protocol stacks in order to seamlessly roam across

heterogeneous wireless environments. Wireless access networks dynamically load signaling system support into mobile devices. We call this service 'reflective' in this context because mobile devices can identify the mobility management architectures and radios supported by neighboring wireless access networks, and customize their signaling systems and wireless links in order to interact with disparate access networks.

In order to support the dynamic introduction of handoff control and mobility management services, we have implemented a service creation environment that explicit supports transportable code by dynamically selecting, deploying and binding distributed objects. The service creation environment is implemented using CORBA [26] middleware. The service creation process allows the network architect to create new objects using inheritance of abstract classes (e.g., an abstract handoff detection algorithm class). Service controllers activate objects invoking binding calls on object control interfaces for the deployment of services.

Programmable handoff services are composed using profiling scripts. The scripting language is simple and supports command, assignment and exception handling statements. Command statements are used for declaring programmable handoff services, adding, deleting and customizing programmable handoff objects, and creating and removing object bindings. Network architects can customize objects during the profiling process. In this case parameters characterizing the operation of a service (e.g., user, service specific or environmental parameters) can be passed in objects at run-time through the profiler and service controllers.

In what follows, we present the design and implementation of the multi-handoff and reflective handoff services, and discuss the profiling process.

4.1 Multi-handoff Access Network Service

4.1.1 Objects

A multi-handoff access network service is composed from a set of distributed CORBA objects. We have deployed a multi-handoff access network service over an experimental programmable mobile network based on WaveLAN radios. Figure 2 shows the implementation of the handoff control model for the multi-handoff access network service. As discussed earlier, the handoff control model separates the algorithms that support beaconing, channel quality measurement and handoff detection. Objects shown in Figure 2 are grouped into beaconing systems, measurement systems, and detection algorithms, which are the components of the handoff control model. Figure 2 also shows object interactions and their invocation order (e.g., NCHO-1, NCHO-2, etc). The handoff control objects comprise:

- *beacon producer* (BeaconProducer) and *measurement producer* (MeasurementProducer) objects, which invoke low-level wireless LAN utility functions. Beacon producer objects transmit beacons at specified frequencies. Measurement producer objects generate 'raw' channel quality measurements. Measurement and beacon producer objects can simultaneously participate in multiple styles on handoff control.
- *signal strength monitor* (*_APSNRMonitor, *_MDSNRMonitor) objects, which collect and average wireless signal strength measurements. SNR represents only one of the many measurements that can be used for handoff decision-making.
- *detection algorithm* (*_DetectionAlgorithm) objects, which make decisions for handoff based on signal strength measurements. Each handoff style employs its own set of signal strength monitors and detection algorithms in order to determine the best access points that mobile devices should be attached to.

Our implementation of the mobility management model is based on an extended Mobiware [12] architecture. Mobiware is programmable, promoting the separation between signaling, transport and state management. Mobility management services include session rerouting, mobility state management and wireless transport configuration, as discussed in Section 3. All sessions that operate between a mobile device and an associated Internet gateway are abstracted as a single state entity called a 'flow bundle' in a Mobiware wireless access network. Flow bundles are used during handoff to switch IP flows in Mobiware access networks and provide general purpose encapsulating and routing services similar to ATM virtual paths or IP tunnels. Open programmable switches allow the establishment, removal, rerouting and adaptation of flow bundles. Thus, the access network behaves as a pool of resources allowing different handoff styles to operate in parallel. Mobiware comprises the following mobility management objects:

- *mobility agent* (MobilityAgent) objects, which reroute sessions to/from mobile devices when mobile devices change their points of attachment. Mobility management is a fully distributed algorithm that includes one or more mobility agents for scalability. Using flow bundles a mobility agent object only has to discover a single crossover switch and reroute all sessions to/from the new access point of a mobile device. All handoff styles can simultaneously use the same mobility agent objects.
- *mobile registration database* (MobileRegistrationDB) objects, which cache flow-bundle state in wireless access points. Unique flow bundle identifiers characterize mobile devices and their associated state. Flow bundle state is expressed in terms of namespace (i.e., VCI/VPI) allocations, QOS adaptation profiles and active transport preferences.
- *datapath* (AP_Datapath) objects, which configure wireless transport mechanisms in wireless access points and mobile devices. For example, datapath objects support the dynamic introduction of valued-added QOS algorithms (e.g., media scaling and adaptive FEC) in wireless access points to compensate against time-varying impairments.

The handoff execution interface comprises a generic 'pre-bind' method that initiates any 'priming' actions at wireless access points and a generic 'handoff' method that executes handoff. The arguments passed into the handoff method include the access point and mobile device identifiers that participate in the handoff operation. Access points are identified by name, IP address and WaveLAN NWIDs where NWIDs are logical channel identifiers. Mobile devices are only identified by name and IP address. No NWID is required because the mobile device's NWID changes during handoff. The name and IP address of a wireless interfaces (i.e., an access point or mobile device) is included in WaveLAN beacons. In this manner, mobile devices can identify access points in a wireless access network and access points can detect the presence of mobile devices in coverage areas.

Currently, our handoff execution interface is simple and supports programmable handoff services using WaveLAN-based radios. Handoff adapters have been implemented for the network controlled, mobile assisted and mobile controlled handoff styles. Handoff adapters comprise adapter objects deployed in mobile devices, access points and in the network. The order in which mobility management services are invoked is dependent on the specific handoff style executing.

4.1.2 Distributed Algorithms

Using the distributed objects described above we implemented a set of algorithms to support alternative styles of handoff control in a multi-handoff access network. Figures 3-5 illustrate a set of distributed algorithms that realize network controlled, mobile assisted, and mobile controlled handoff styles, respectively. Each algorithm comprises handoff detection and handoff execution processes. The object interaction and invocation order for each handoff algorithm is shown in Figure 2. Handoff control objects implement the handoff detection process, and mobility management objects and handoff adapters implement the handoff execution process. Handoff styles differ in the location where handoff control takes place and is executed.

Network Controlled Handoff

In the network controlled handoff scheme a signal strength monitor object, running in a wireless access point, continuously measures the signal strength of a mobile device, as indicated by NCHO-1 in the handoff detection algorithm shown in Figure 3(a). The signal strength monitor initiates handoff (NCHO-2) when the signal strength drops below a certain threshold. A detection algorithm, running in the wireless access network, queries and compares mobile-device measured signal strength from all neighboring access points (NCHO-3 to NCHO-5). Wireless SNR values determine the best candidate access point for the mobile device. Neighboring monitors report average the SNR measured over two consecutive beacon query intervals. Beacon query intervals are user defined, typically being 300 msec in duration. Handoff execution is initiated when a candidate access point is detected with a better service quality than the current point of attachment. The handoff detection process is repeated once handoff execution is complete. Network controlled handoff moves most of the complexity for controlling handoff from the mobile device to the network. This style of handoff simplifies the mobile device software design.

The wireless access network initiates the handoff execution process. The detection algorithm interacts with a handoff adapter object to realize handoff. The handoff adapter invokes a getFlowBundleState() method on a

mobile registration database located at the mobile device's old access point to obtain the flow bundle state information, as indicated by NCHO-6 in Figure 3(b). As discussed previously, the flow bundle state represents the aggregate state information for all flows/sessions to and from a mobile device. This state information is used to speed handoff in a Mobiware access network [12]. The mobility agent object interacts with a router object and open programmable switch servers to calculate the cross-over switch on behalf of a mobile device and reroute the mobile device's active sessions (NCHO-7) represented by the flow bundle state. Following this the handoff adapter object invokes a setUpDatapath() method to create a channel in the new access point accommodating active transport 'plug-ins' for value-added QOS support (NCHO-8). Once this is complete, the wireless radio link transfer takes place at the mobile device (NCHO-9, NCHO-10). Channel change is realized as a change in WaveLAN NWID. Finally, the handoff adapter registers the mobile device with the new access point (NCHO-11).

```
HANDOFF-DETECTION  //Network-Controlled HandOff (NCHO)
( access_point AP, mobile_device mobile, int thres ) {

   let am be a NCHO_APSNRMonitor object located at AP;
   let mp  be a  MeasurementProducer object located at AP;
   let da be a NCHO_DetectionAlgorithm object located
   inside the access network;

   while (true) {
NCHO-1:  am invokes mp.queryBeaconInfo  (mobile);
      am calculates the average signal strength SNR-avg for mobile;
      if (SNR-avg <= thres) {
NCHO-2:  am invokes da.detectHandoff  (AP, mobile);
         break; }
      else {
         sleep (am.queryInterval); }}
      for (each neighboring access point AP-neigh) {
         let am-neigh be a NCHO_APSNRMonitor object located at AP-neigh;
NCHO-3:  da invokes am-neigh.getMobileSNR();  }

NCHO-4,5: obtained SNR values are compared
   da calculates the best candidate access point AP-best for mobile;
   HANDOFF-EXECUTION (AP-best, AP, mobile);
   HANDOFF-DETECTION (AP-best, mobile, thres); }
```

(a) handoff detection

```
HANDOFF-EXECUTION
( access_point AP-new, access_point AP-old, mobile_device mobile) {

   let ha be a NCHO_Adapter object located inside the access network;
   let ha-old be a NCHO_Adapter object located at AP-old;
   let ha-new be a NCHO_Adapter object located at AP-new;
   let ma be a MobilityAgent object located inside the access network;
   let rdb-old be a MobileRegistrationDB object located at AP-old;
   let rdb-new be a MobileRegistrationDB object located at AP-new;
   let dp-new be a AP_Datapath object at AP-new;

   //query for mobility state
NCHO-6:  ha invokes rdb-old.getFlowBundleState  (mobile);
   let state-old be the flow-bundle state returned by rdb-old;

   //session re-routing
NCHO-7:  ha invokes ma.handoffFlowBundle (state-old, mobile);
   ma calculates the cross-over switch for mobile;
   ma reroutes the flow-bundle associated with mobile;
   let state-new be the flow-bundle state returned by ma;

   //wireless transport configuration
NCHO-8:  ha invokes dp-new.setUpDatapath  (state-new, mobile);
NCHO-9:  ha-old invokes mobile.handoffNotice (AP-new);
NCHO-10:  mobile invokes this.radioLinkTransfer (AP-new);

   //mobile registration
NCHO-11:  ha invokes rdb-new.mobileRegistration  (state-new, mobile);
NCHO-12:  ha-new binds to mobile; }
```

(b) handoff execution

Figure 3: Network Controlled Handoff

Mobile Assisted Handoff

The mobile assisted handoff scheme moves some of the functional support, and therefore complexity, to the mobile device. In this scheme, an access point continuously measures the signal strength of a mobile device, as indicated by MAHO-1 in Figure 4(a). If the signal level drops below a certain threshold, a detection algorithm running in the mobile device's serving access point invokes a MeasureON() method to initiate signal strength measurements at the mobile device (MAHO-3). Measurements are reported via a MeasureReport() invocation (MAHO-5). The detection algorithm does not need to collect any additional measurements in support of handoff. Rather, the handoff decision is based on data collected by the mobile device. Our implementation of mobile assisted handoff is based on the GSM mobile assisted handoff scheme [21]. In the mobile assisted handoff scheme, handoff execution is driven by an adapter object running at the mobile device's old access point and not in a server operating in the wireless access network. Handoff execution includes flow-bundle rerouting, wireless transport management and mobile registration, as in the case of network controlled handoff, as indicated by MAHO-9 to MAHO-14 in Figure 4(b). In the case of mobile assisted handoff the handoff execution algorithm is distributed resulting in shorter handoff completion times. When handoff execution is complete, the handoff adapter and detection algorithm objects operating in the new access point bind to the mobile device (MAHO-7, MAHO-15).

```
HANDOFF-DETECTION  //Mobile-Assisted HandOff (MAHO)
( access_point AP, mobile_device mobile, int thres ) {

   let am be a MAHO_APSNRMonitor object located at AP;
   let mp be a MeasurementProducer  object located at AP;
   let da be a MAHO_DetectionAlgorithm object located at AP;
   let mm be a MAHO_MDSNRMonitor object located at mobile;
   let mp-mobile be a MeasurementProducer  object located at mobile;

   while (true) {
MAHO-1:  am invokes mp.queryBeaconInfo(mobile);
      am calculates the average signal strength SNR-avg for mobile;
      if (SNR-avg <= thres) {
MAHO-2:  am invokes da.detectHandoff  (AP, mobile);
         break; }
      else {
         sleep (am.queryInterval); }}
MAHO3:  da invokes mm.measureON();
   while (true) {
      //mobile measures signal strength from all neighboring access points
MAHO-4:  mm invokes mp-mobile.queryBeaconInfo();
MAHO-5:  mm invokes da.measureReport();
      da calculates the best candidate access point AP-best for mobile;
      if (AP-best != AP) {
MAHO-6:  da invokes mm.measureOFF();
         break; }
      sleep (mm.queryInterval); }

 //handoff execution and binding
   HANDOFF-EXECUTION (AP-best, AP, mobile);
   let da-best be a MAHO_DetectionAlgorithm object
   located at AP-best;
MAHO-7:  da-best binds to mm;
MAHO-8:  mm binds to da-best;
   HANDOFF-DETECTION (AP-best, mobile, thres); }
```

(a) handoff detection

```
HANDOFF-EXECUTION
( access_point AP-new, access_point AP-old, mobile_device mobile) {

   let ha-old be a MAHO_Adapter object located at AP-old;
   let ha-new be a MAHO_Adapter object located at AP-new;
   let ma be a MobilityAgent object located inside the access network;
   let rdb-old be a MobileRegistrationDB object located at AP-old;
   let rdb-new be a MobileRegistrationDB object located at AP-new;
   let dp-new be a AP_Datapath object at AP-new;

   //query for mobility state
MAHO-9:  ha-old invokes rdb-old.getFlowBundleState  (mobile);
   let state-old be the flow-bundle state returned by rdb-old;

   //session re-routing
MAHO-10:  ha-old invokes ma.handoffFlowBundle (state-old, mobile);
   ma calculates the cross-over switch for mobile;
   ma reroutes the flow-bundle associated with mobile;
   let state-new be the flow-bundle state returned by ma;

   //wireless transport configuration
MAHO-11:  ha-old invokes dp-new.setUpDatapath  (state-new, mobile);
MAHO-12:  ha-old invokes mobile.handoffNotice (AP-new);
MAHO-13:  mobile invokes this.radioLinkTransfer (AP-new);

   //mobile registration
MAHO-14:  ha-old  invokes rdb-new.mobileRegistration  (state-new, mobile);
MAHO-15:  ha-new binds to mobile; }
```

(b) handoff execution

Figure 4: Mobile Assisted Handoff

Mobile Controlled Handoff

In the mobile controlled handoff scheme the signal strength measurements are taken by the mobile device, as indicated by MCHO-1 in Figure 5(a). If a candidate access point having better signal strength is detected then the handoff execution process is initiated. The mobile controlled handoff scheme moves most of the complexity for detecting handoff to the mobile device alleviating the network from centralized control of the handoff process. In this respect mobile controlled handoff is scalable and more distributed than the other schemes. However, it assumes that the mobile device (e.g., a wireless laptop device) can continuously monitor signal strength measurements.

A handoff adapter object located at the mobile device drives handoff execution. Mobile controlled handoff is executed as a forward, soft handoff. To accomplish soft handoff, our radios based on WaveLAN operate in promiscuous mode so that the mobile device simultaneously receives data from multiple access points. First, wireless radio link transfer takes place (MCHO-6) as a change in WaveLAN NWID. Following this a mobility agent object is invoked via adapter objects located at the mobile device and new access point. The mobility agent object reroutes the flow bundle, which is specific to a mobile device (MCHO-8). Finally, mobile registration (MCHO-9) and wireless transport configuration steps take place (MCHO-10). Forward handoff requires the creation of a binding between an adapter object running in the mobile device and an adapter object operating at the new access point. These adapter objects are used to contact the mobility agent operating in the wireless access network. To eliminate the latency introduced by object bindings we setup and cache bindings prior to handoff execution (MCHO-3 to MCHO-5). An object binding is established at the best candidate access point when the signal strength in the main communication path drops below a certain threshold. The ‘pre-bind’ method call supported by the handoff execution interface is used for this purpose. Different combinations of handoff adapters result in different styles of programmable handoff (e.g., backward, hard handoff, etc).

Our system implementation has been optimized to reduce the binding and Remote Procedure Call (RPC) overhead (i.e., latency) associated with open signaling [4]. Oneway CORBA calls have been used to increase the level of parallelism for the interaction of programmable handoff objects. A mobility agent has been designed to setup wireline connections in the wireline access network by sending parallel invocations to switch servers [11]. This results in a speed up of the re-routing phase of the handoff algorithm over conventional hop-by-hop signaling Mobile registration databases have been implemented as hash tables to allow information retrieval in $\Theta(1)$ time. Flow bundle identifiers have been used as hash keys. Mobility agents have been implemented in 'multi-threaded' as well as 'sequential' modes. For the experiments reported in this paper, mobility agents operate in a sequential mode (e.g., reroute flow bundles one-by-one). Results from our experiments are discussed in Section 5.

```
HANDOFF-DETECTION  //Mobile-Controlled HandOff (MCHO)
( access_point AP, mobile_device mobile, int thres) {

   let mm be a MCHO_MDSNRMonitor object located at mobile;
   let mp be a MeasurementProducer object located at mobile;
   let da be a MCHO_DetectionAlgorithm object located at mobile;
   let ha be a MCHO_Adapter object located at mobile;

   while (true) {
      //mobile measures signal strength from all neighboring access points
MCHO-1: da invokes mm.getAPSNR();
MCHO-2: mm invokes mp.queryBeaconInfo();
      da calculates the best candidate access point AP-best for mobile;
      if (AP-best != AP) {
         HANDOFF-EXECUTION (AP-best, AP, mobile);
         HANDOFF-DETECTION (AP-best, mobile, thres);
         return; }
      let SNR-avg be the average signal strength from AP
      if ((AP-best == AP) && (SNR-avg <= thres)) {
         //pre-bind
         da calculates the second best access point AP-sec for mobile;
         let ha-sec be a MCHO_Adapter object located at AP-sec;
MCHO-3: mobile invokes this.radioLinkTransfer (AP-sec);
MCHO-4:  ha binds to ha-sec;
MCHO-5:  mobile invokes this.radioLinkTransfer (AP); }
         sleep (da.queryInterval); }}
```

(a) handoff detection

```
HANDOFF-EXECUTION
( access_point AP-new, access_point AP-old, mobile_device mobile) {

   let ha be a MCHO_Adapter object located at mobile;
   let ha-new be a MCHO_Adapter object located at AP-new;
   let ma be a MobilityAgent object located inside the access network;
   let rdb-new be a MobileRegistrationDB object located at AP-new;
   let dp-new be a AP_Datapath object at AP-new;
MCHO-6: mobile invokes this.radioLinkTransfer (AP-new);

   //flow-bundle state is stored at mobile devices as well as access points
   let state-old be the flow-bundle state stored at mobile;
MCHO-7 ha invokes ha-new.handoffExecution  (AP-old, state-old, mobile);
MCHO-8: ha-new invokes ma.handoffFlowBundle (state-old, mobile);
   ma calculates the cross-over switch for mobile;
   ma reroutes the flow-bundle associated with mobile;
   let state-new be the flow-bundle state returned by ma;

   //mobile registration
MCHO-9: ha-new invokes rdb-new.mobileRegistration  (state-new, mobile);

   //wireless transport configuration
MCHO-10: ha-new invokes dp-new.setUpDatapath  (state-new, mobile);
MCHO-11: ha-new invokes mobile.handoffNotice (AP-new);  }
```

(b) handoff execution

Figure 5: Mobile Controlled Handoff

4.2 Reflective Handoff Service

We have implemented reflective handoff as a mobile controlled handoff scheme. Access points transmit beacons that additionally carry globally unique identifiers designating specific access networks. A reflective detection algorithm uses access network identifiers to determine whether a mobile device is likely to move to the coverage area of a new access network. Each mobile device maintains a local cache of signaling system modules. Signaling system modules are collections of objects supporting mobility management services in mobile devices. Before a mobile device performs a handoff to a new access network, it checks whether a signaling module associated with the new candidate access network is cached. If a signaling module is not cached it is dynamically loaded. Access points support module loaders deployed during the service creation process. A signaling system is loaded from the old access network. A two-way handshake mechanism is used for loading signaling modules, which are loaded before reflective handoff is executed. Access networks schedule the transmission of signaling modules over the air interface, to avoid flooding the wireless network.

Reflective handoff is managed by handoff adapters, which activate or deactivate signaling modules on-demand. The handoff execution interface for the reflective handoff service includes a 'configure' method, which is used for binding new signaling systems. Parameters associated with access network state (e.g., the address of the gateway to the Mobile IP Internet) are passed into signaling modules upon activation. Module loaders transmit access network state when loading signaling system support into mobile devices. Reflective handoff may involve the loading of

entire mobility management protocol stacks or configuration scripts, which customize objects already cached at mobile devices.

Two distinct types of access networks support reflective handoff in our testbed as shown in Figure 6: Mobiware and Cellular IP access networks. Cellular IP [13] delivers fast local handoff control in datagram oriented access networks. In addition, Cellular IP supports per-mobile host state, paging, routing and handoff control in a set of access networks that are interconnected to the Internet through gateways. In Cellular IP, packets sent from mobile hosts create routing caches pointing to the downlink path so that packets destined to a mobile device can be routed using the route cache. Mobiware and Cellular IP signaling modules use the IP protocol to communicate with access networks. Mobiware and Cellular IP access networks support the same wireless data link and physical layers (WaveLAN) in our testbed but use different mobility management systems. Future work will include extending reflective handoff to allow access networks to load software radio-based physical and data link layer support into mobile devices.

In our testbed, a Mobile IP enabled internetwork connects the Mobiware and Cellular IP wireless access networks via gateways. Mobile IP is used for managing macro-level mobility between gateways, whereas the Cellular IP and Mobiware wireless access support fast local handoff control. Hierarchical mobility management in IP-based mobile networks has been widely reported in the literature [13, 16, 23, 29]. A mobile device attached to an access network uses the IP address of the gateway as its care-of address. Access networks provide mechanisms for initiating Mobile IP-based inter-gateway handoffs and for establishing datapaths between gateways and access points where mobile devices are attached.

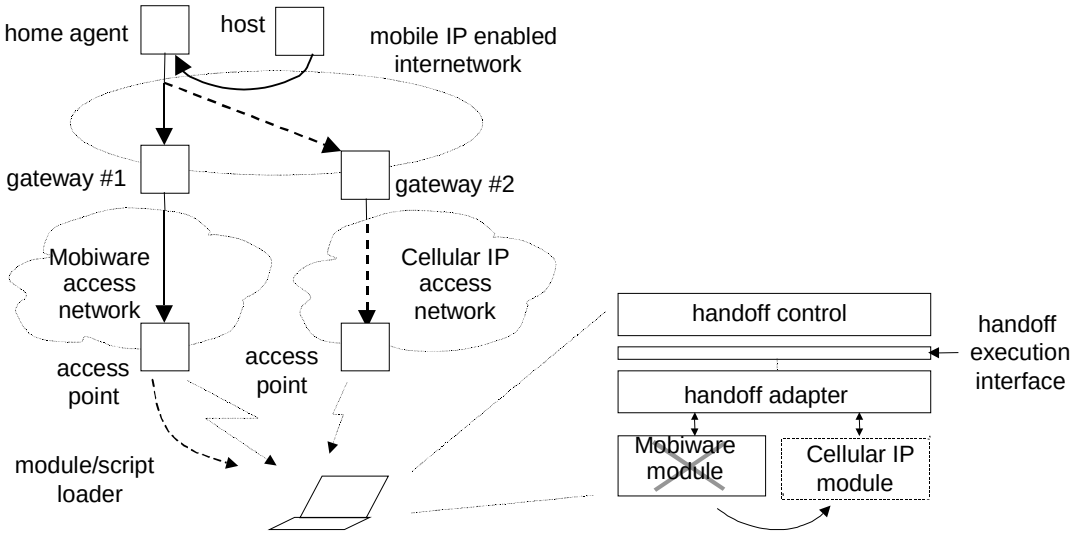


Figure 6: Reflective Handoff Service

Signaling modules implement generic handoff execution functions as dynamic link libraries using the Windows NT operating system. Three types of handoff are supported: (i) 'internal' handoffs, which take place between access points of the same access network; (ii) 'entry' handoffs, which take place when a mobile device is attached to a new access network; and (iii) 'exit' handoffs, which take place when a mobile device leaves an access network. The handoff execution interface supports method calls for internal, entry and exit handoffs.

Reflective handoff requires that all the signaling modules associated with the handoff process are loaded and activated. Reflective handoff has been implemented as the process of invoking an 'exit' handoff on the signaling system of the old access network and an 'entry' handoff on the signaling system of the new wireless access network. Access networks realize execution calls in different ways and support mechanisms for registering the care-of address of mobile devices (i.e., gateway IP address) with their corresponding home agents. Care-of address registration has been realized as part of 'entry handoff' or 'exit handoff'. When an entry handoff takes place, access networks check whether the care-of address of a mobile device has been registered with its home agent. If the care-

of address is not registered then the access network initiates registration. Registration support during 'exit' handoff is optional.

We have programmed our handoff control system to load signaling modules as soon as the mobile device detects that it is inside the coverage area of an access network where the signaling system is not cached. This loading algorithm minimizes the probability of handoff failure due to absence of a signaling module at the mobile device. A soft state mechanism used for managing stored signaling modules is used to avoid having large caches. A timer associated with a module is refreshed while a mobile device remains inside the coverage area of an access network associated with a particular module or set of modules. Mobile devices can permanently cache signaling modules associated with access networks, however.

activate_service	*activates all objects within a service*
delete_service	*deletes a service*
load_object	*activates an object associated with a service at a specified location*
delete_object	*deletes an object from a specified service*
bind_objects	*creates a binding between two activated objects*
remove_binding	*removes a binding*
get_paramater	*gets the value of a parameter used by an object*
set_parameter	*sets the value of a parameter used by an object*

Table 1: Command Statements for Profiling Handoff Services

```
# part 1: service declarations
new_service beaconing_system;
new_service measurement_system;
new_service detection_algorithm;
new_service handoff_adapter;
new_service mobility_management_system;

# part 2: variable declarations
access_points = {“cymbal.comet.columbia.edu”, “voice.comet.columbia.edu”};
mobility_agent = “cvn1.comet.columbia.edu”;
beacon_interval = 100;
query_interval = 300;
max_requests = 500;

# part 3: object activation
load_object “BeaconProducer” $beaconing_system $access_points $beacon_interval;
load_object “MeasurementProducer” $measurement_system $access_points $query_interval;
load_object “MAHO_APSNRMonitor” $measurement_system $access_points $max_requests;
load_object “MAHO_DetectionAlgorithm” $detection_algorithm $access_points $max_requests;
load_object “MAHO_Adapter” $handoff_adapter $access_points;
load_object “MobilityAgent” $mobility_management_system $mobility_agent $max_requests;
load_object “AP_Datapath” $mobility_management_system $access_points $max_requests;
load_object “AP_MobileRegistrationDB” $mobility_management_system $access_points $max_requests;

# part 4: object bindings
bind_objects “MAHO_APSNRMonitor”@$access_points => “MAHO_DetectionAlgorithm”@$access_points;
bind_objects “MAHO_APSNRMonitor” @$access_points => “MeasurementProducer”@$access_points;
bind_objects “MAHO_DetectionAlgorithm”@$access_points => “MAHO_Adapter”@$access_points;
bind_objects “MAHO_Adapter”@$access_points -> “MobilityAgent”@mobility_agent;
bind_objects “MAHO_Adapter”@$access_points -> “AP_Datapath”@$access_points;
bind_objects “MAHO_Adapter”@$access_points -> “AP_MobileRegistrationDB”@$access_points;
end;
```

Figure 7: Mobile Assisted Handoff Profile

4.3 Profiling Handoff Services

Service creation requires the specification of objects for transmitting beacons, collecting channel quality measurements, detecting handoff and managing mobility. Currently, all the objects involved in the service creation

process need to be declared and their location defined. Such an approach works well for small access networks and simple handoff services but does not scale well. An alternative approach separates the 'binding rules' defining handoff control and mobility management architectures (e.g., a rule for placing handoff detection algorithms inside the network) from the 'binding data' (e.g., network topology, user preferences, etc.). In this case, programmable handoff services represent the instantiation of a set of binding rules over some binding data and are composed using profiling scripts.

A list of command statements used by our scripting language is illustrated in Table 1. An example profiling script for a mobile assisted handoff service is shown in Figure 7. The profiling script describes the instantiation of a mobile-assisted handoff service over a wireless access network that has only two access points. The profiling script is created by a profiler and passed into service controllers to instantiate the service. Mobile device software is deployed separately (e.g., during reflective handoff). In the first part of the script new services are declared. Services represent contexts that are associated with programmable handoff objects. Five services are declared (viz. beaconing_system, measurement_system, detection_algorithm, handoff_adapter, mobility_management_system), which reflect the binding model characterizing a programmable mobile network. In the second part of the profiling script local variables are declared, which are used as arguments in subsequent statements. The access_points variable is a list that contains the names of the access points in the wireless access network. The mobility_agent variable represents the name of the host where the mobility agent object (discussed in Section 4) is located. The beacon_interval and query_interval variables represent the time intervals between successive transmissions of beacons and collections of channel quality measurements, respectively. The max_requests variable represents an upper bound on the number of mobile devices, which are served by the wireless access network

In the third part of the profiling script objects are activated using the load_object statement. Object activation is realized as the spawning of a server process that supports the object implementation. Programmable handoff objects are distinguished on the basis of name, location and offered service. In the example script, eight types of objects are activated. These objects constitute a mobile assisted handoff service. With the exception of the MobilityAgent object, mobile assisted handoff objects are activated in wireless access points. We consider programmable handoff objects as both clients and servers. The final part of the profiling script describes object bindings. An object reference is described using the Interoperable Object Reference (IOR) format [26], which is part of the CORBA specification. An IOR stores the information needed to locate and communicate with an object. For example, an IOR stores the hostname where an object is activated and the object's TCP/IP port number. Two binary operators are used for specifying object bindings in our scripting language. The 'one-to-one' (=>) operator describes 'one-to-one' bindings between objects, which are on the left-hand side of the operator, and objects, which are on the right-hand side. The 'many-to-many' operator (->) binds every object on the left-hand side to every object on the right-hand side. The bindings are necessary to make the mobile assisted handoff service operational: First, MAHO_APSNRMonitor objects bind to MAHO_DetectionAlgorithm and MeasurementProducer objects. Following this the MAHO_DetectionAlgorithm objects bind to MAHO_Adapter objects. Finally, MAHO_Adapter objects bind to MobilityAgent, AP_Datapath and AP_MobileRegistrationDB objects. Once object bindings are created, the handoff service starts executing in the multi-handoff access network.

5. Evaluation

To evaluate the programmable handoff architecture we use a mixture of testbed implementation and emulation in order to study proof of concept and scalability issues associated with our design. We extend the Mobiware [12] and Cellular IP testbeds [13] developed at Columbia University to support the multi-handoff access and reflective handoff services. The Mobiware testbed is used to implement and evaluate the multi-handoff access service while the reflective handoff service is evaluated using a combination of Cellular IP and Mobiware wireless access networks. In what follows, we discuss the results from the implementation and evaluation.

5.1 Experimental Platform

Our experimental environment is illustrated in Figure 8. The Mobiware testbed provides wireless access to the Internet and comprises four ATM switches (viz. ATML Virata, Fore ASX/100, and NEC model 5 switches) and

four wireless access points. To provide a larger network testbed for programmable handoff evaluation, switches allow multiple virtual network elements to be operational within the same physical nodes. Three virtual ATM switches (ATML 1, 2, and 3 shown in Figure 8) in our network are 'switchlets' [30] physically co-located at the same physical switch. Each switchlet corresponds to a different CORBA server with a different name space and manages its own resources independently from other co-located switchlets. Access points are multi-homed 300 MHz Pentium PCs that provide radio access to a wireline IP switched access network. High performance notebooks provide support for mobile applications and mobile access to network services. Wireline ATM links operate at OC-3 rates between the switches and at 25 Mbps between the switches and access points. Our testbed radios are based on WaveLAN operating in the 2.4-2.8 GHz ISM band. We use 2 Mbps WaveLAN cards for which we have a low-level radio utility API for programming beacons, getting signal-strength measurements and controlling some radio aspects. The Cellular IP access network consists of three base stations based on multi-homed 300 MHz Pentium PCs. One of the base stations serves as a gateway router to the Internet. Interconnects between Cellular IP base stations are 100 Mbps full duplex links. Mobiware and Cellular IP access networks are connected to a 100 Mbps Ethernet LAN supporting Mobile IP.

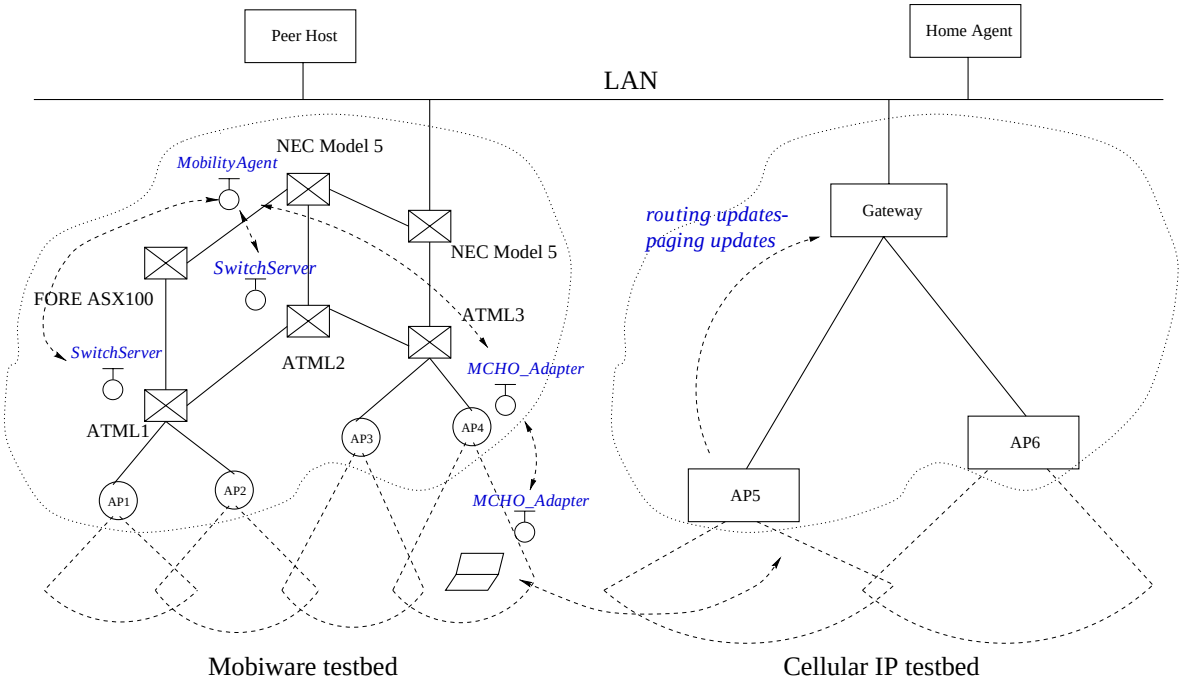


Figure 8: Experimental Environment

Access points and switches of the Mobiware testbed support our service creation environment, programmable handoff control and mobility management systems. For the results provided in this paper the mobile devices and access points use IONA's Orbix v2.0 CORBA running Windows NT and UNIX operating systems. Cellular IP base stations, gateways and mobiles use the Cellular IP protocol [31].

We augmented the experimental testbed platform discussed above with an emulation mode to evaluate the scalability of the programmable handoff architecture to support large numbers of mobile devices. Our aim is to analyze how well our middleware architecture performs in meeting the requirements imposed by different types of mobile environments. An emulation platform has been built consisting of a mobility emulator and the programmable handoff architecture operating in the access points and mobile capable switches of the extended Mobiware testbed.

The mobility emulator, shown in Figure 9, emulates the random movement of mobile devices and supports different levels of mobility and signaling load. The only major modification made to the access network source code used for the experimental evaluation relates to the measurement producer objects, which measure channel quality in wireless access points. Measurement producer objects have been modified to suppress the generation of

real channel quality measurements. Rather producer objects receive data from a mobility emulator. Emulated mobile devices have been programmed to remain inside the same cell for an exponentially distributed interval and to move with the same probability to any neighboring cell. The propagation model used by the mobility emulator is based on the path loss component of signal fading only.

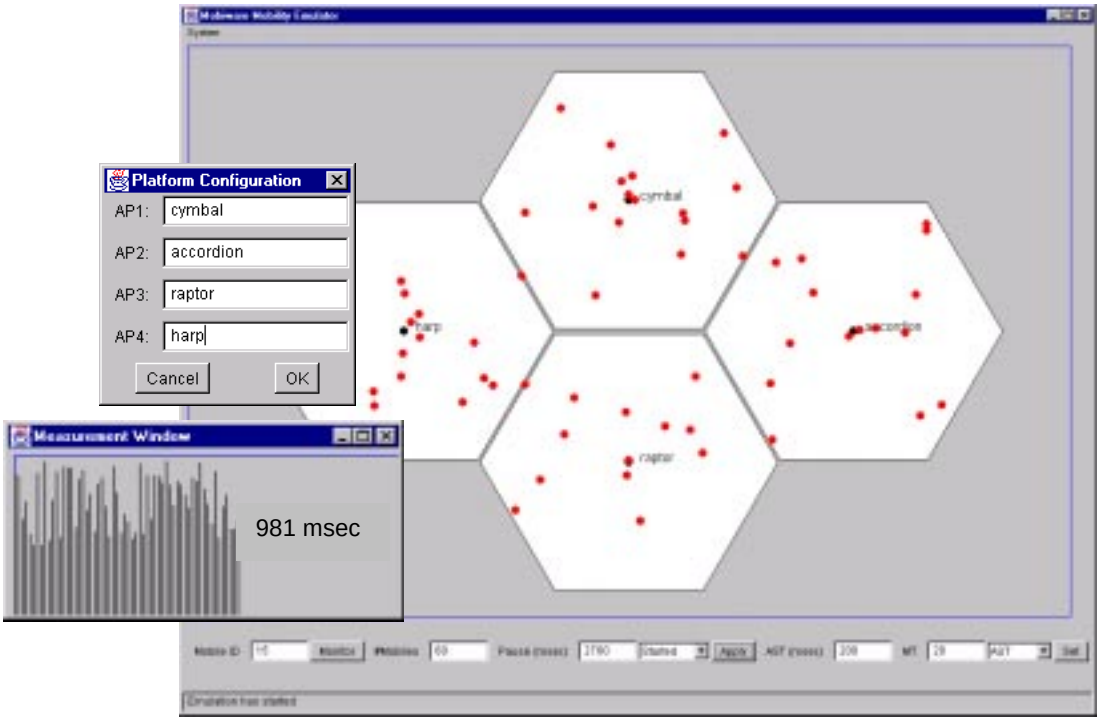


Figure 9: Mobility Emulator

5.2 Multi-handoff Access Service Analysis

A number of experiments provide insight into the performance of the multi-handoff access network service. An important objective of the evaluation of service is to measure the latency associated with various handoff styles. For these experiments, we streamed a single video flow (i.e., true_lies.mpg at 350 kbps [12]) from a fixed network server to three mobile devices and performed successive handoffs with these mobile devices between access points AP2 and AP3, as illustrated in Figure 8. The system is lightly loaded and the cross over switch located at the ATML2 switch. The network is programmed to simultaneously support mobile controlled, mobile assisted and network controlled handoff styles with each mobile device being programmed to support a different style of handoff under consideration.

Table 2 summarizes our handoff latency measurement results for each one of the three schemes. Twenty measurements for each handoff style were recorded. The average handoff latency measured for the mobile-controlled handoff is 41 msec. This measurement comprised 22 msec for wireline connection setup and 19 msec for wireless connection setup between the access point and mobile device. Mobiware active filters are not used in the experiments. For an evaluation of the Mobiware active filtering system see [12]. Mobile controlled handoff is associated with the least amount of latency. In this scheme a mobile device periodically takes signal strength measurements and initiates handoff. The average handoff latency measured for the mobile assisted handoff scheme is 750 msec. The greatest portion of the latency (709 msec) is absorbed by the measuring process, which records neighboring access point signal strength from the perspective of the mobile device. The ‘hunt’ period over which measurements are collected at the mobile device is set to 300 msec, whereas beacons are transmitted by access points and mobile devices every 100 msec. Our detection algorithm initiated handoff if a candidate access point with better SNR was present over two successive hunt periods. The average network controlled handoff latency

measured is 683 msec. The measurement collection component of the handoff is 641 msec. These results indicate that the performance of a multi-service access network is satisfactory when the network is lightly loaded. Our system performs well because binding latencies are eliminated. We experienced higher latencies when bindings between objects were not setup or cached prior to handoff.

To test the scalability of our system we used the mobility emulator. We emulated the movement of 120 mobile devices and took measurements over a period of 10 min. Half the emulated mobile devices used mobile assisted handoff and the other half network controlled handoff. We measured the average handoff latency observed by each emulated mobile device and varied the average time between successive handoffs, which resulted in different cell crossing rates and signaling loads. The average handoff latency experienced by a single mobile device as a function of the cell crossing rate (characterizing the mobile environment) is presented in Figure 10(a). The figure shows that the average handoff latency for the mobile assisted and network controlled handoff styles is much higher when the system operates under signaling load. The average handoff latency experienced by a mobile device when performing mobile assisted handoff is 750 msec in the case of a lightly loaded testbed. However, the measured latency for the same handoff scheme is between 1.03 and 2.23 sec when the average cell crossing rate ranged between 2 handoffs/sec and 6 handoffs/sec. In the case of a network controlled handoff the average handoff latency measured is between 1.29 and 3.63 sec.

	wireline connection latency (msec)	*Wireless connection latency (msec)*	*measurement collection latency (msec)*	*total latency (msec)*
Mobile Controlled Handoff	22 ± 1	19 ± 1	-	41 ± 1
Mobile Assisted Handoff	21 ± 1	20 ± 1	709 ± 65	750 ± 65
Network Controlled Handoff	22 ± 2	20 ± 1	641 ± 59	683 ± 59

Table 2: Handoff Latency Measurement

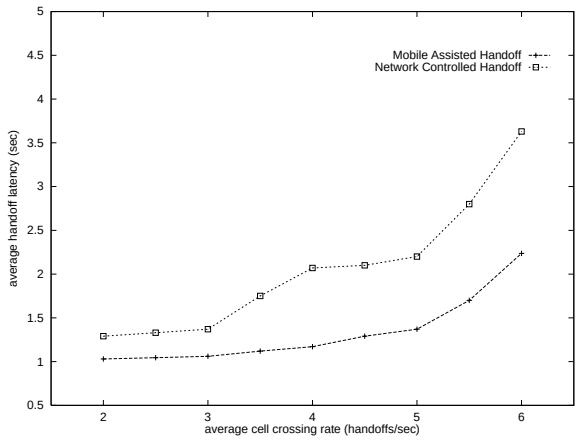
The increase in average handoff latency observed when the system operates under signaling load is partially caused by the fact that signaling functions are implemented as interactions between distributed objects. A common feature of many implementations in CORBA is that remote method invocations are queued until they are served in a first come-first served basis. However, in many distributed systems, such as programmable signaling platforms, service requests occur in bursts. In this case, significant latency can be introduced between the time a request is issued and the time it is served. Latency is introduced during the handoff process in the case of programmable handoff control. This latency is represented as the period between which the handoff execution call is issued by the detection algorithm and the point at which the mobility agent services the call. In the case of a network controlled handoff, this latency is compounded by the fact that access points transmit measurements reporting signal strength associated with mobile devices.

To minimize such latency, we enhanced mobile assisted handoff adapters to transmit aggregated service requests to mobility agents using a single remote method invocation (i.e., signaling interaction). In addition, we enhanced signal strength monitors to transmit aggregated signal strength reports. Aggregated calls carry all service requests that have been issued in the interval between two successive invocations. We call this interval between two successive aggregated calls the aggregated signaling time. We experimented with different values of aggregated signaling time and measured the average handoff latency experienced by mobile devices using mobile assisted and network controlled handoff. We took care that access points were not synchronized when transmitting aggregated calls, avoiding further increase in handoff latency.

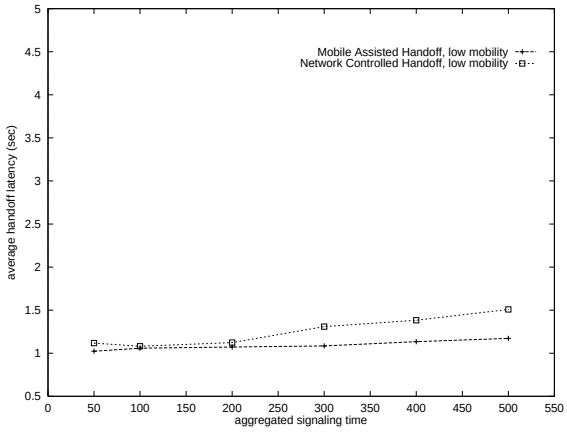
Three sets of experiments evaluate the benefit of aggregated signaling techniques. In the first experiment, the average cell crossing rate of the mobile environment is 2 handoffs/sec. In the next experiment, the average cell

crossing rate of the mobile environment is 4 handoffs/sec corresponding to more frequent mobility. In the third experiment, the average cell crossing rate of the mobile environment is 6 handoffs/sec. The results from these experiments are shown in Figures 10(b)-10(d).

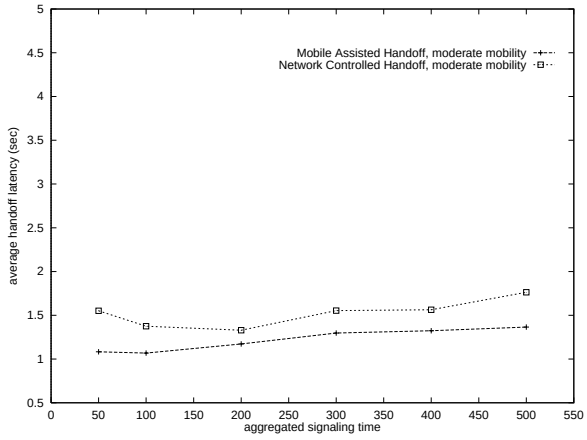
Aggregation of CORBA calls at access points results in significant reduction in the average handoff latency experienced by mobile devices. Optimal values for the aggregated signaling time ranged between 50 and 200 msec. Higher aggregation signaling times result in increased latencies. Aggregation reduces the average handoff to 980 msec for the mobile assisted handoff scheme and 1.08 sec for the network controlled handoff scheme in the case of lowest cell crossing rate. In the case of the highest cell crossing rate the handoff latency is reduced to 1.10 sec for the mobile assisted handoff scheme and 2.41 sec for the network controlled handoff scheme. While it is difficult to compare results from different signaling systems operating under different load conditions we report for the purpose of qualitative analysis that the average handoff latency in MAHO cellular systems is around 0.9 sec. Network controlled handoff latency varies between 5-10 sec in different cellular systems.



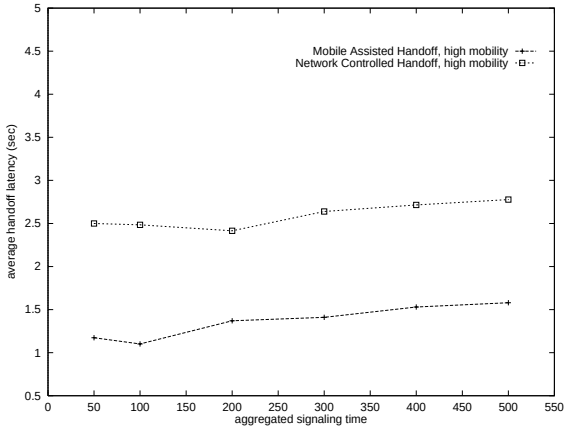
(a) Handoff Latency as a Function of the Average Cell Crossing Rate



(b) Handoff Latency as a Function of the Aggregated Signaling Time (Low Mobility Case)



(c) Handoff Latency as a Function of the Aggregated Signaling Time (Moderate Mobility Case)



(d) Handoff Latency as a Function of the Aggregated Signaling Time (High Mobility Case)

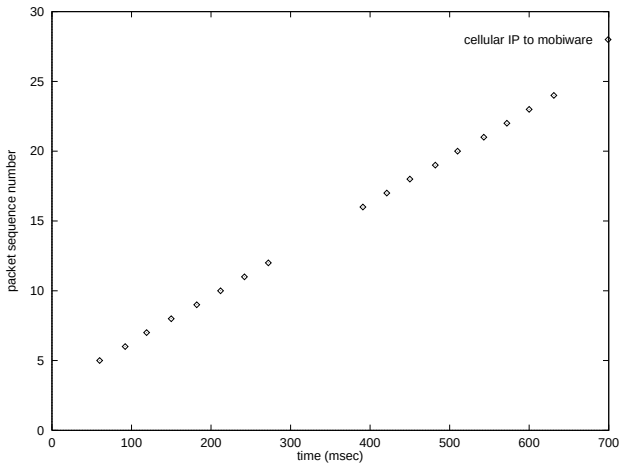
Figure 10: Handoff Latency Under Load Conditions

5.3 Reflective Handoff Service Analysis

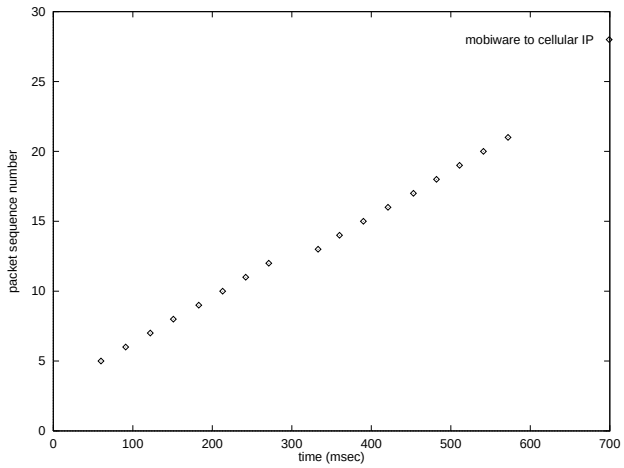
In this section we evaluate the reflective handoff service between Mobiware and Cellular IP wireless access networks. Mobiware and Cellular IP represent rather different wireless access network architectures and their implementation platforms support strikingly different signaling protocols. In this respect supporting seamless handoff between these two networks is good test scenario for reflective handoff. For full details on Mobiware and Cellular IP handoff algorithms see [12] and [13], respectively.

An illustrative example of the performance of our video application is shown in Figure 11. Packet traces are shown for a video stream (i.e., true_lies.mpg) delivered to a mobile device on the downlink during reflective handoff. To study the effect of packet losses on reflective handoff we disable the promiscuous mode of WaveLAN radios. In this way, mobile devices are unable to simultaneously receive data from multiple access points. In the experiments, we successively force handoff between Mobiware and Cellular IP access networks. Reflective handoff latency ranges between 60 and 100 msec. In the example shown in Figure 11, a Cellular IP to Mobiware handoff takes about 90 msec to complete, whereas a Mobiware to Cellular IP handoff takes approximately 60 msec, including entry and exit handoff. A significant part of the overall handoff latency is absorbed by the Mobile IP signaling component (42 msec over a single hop). Typically, loading times do not affect handoff performance because signaling modules are loaded prior to handoff execution. Activation latencies are 10 msec for the Mobiware and Cellular IP modules. Cellular IP only initiates care-of-address registration during 'entry handoffs'. Mobiware supports care-of address registration during both entry and exit handoffs.

Packet loss during Cellular IP to Mobiware reflective handoff is greater than Mobiware to Cellular IP because the care-of address registration occurs after the wireless radio link transfer takes place (i.e., a change in WaveLAN NWID). Mobiware to Cellular IP handoff is more efficient because care-of address registration occurs first and is initiated by the Mobiware access network during exit handoff. Some forwarding delay is introduced at the gateway that connects the Cellular IP access network with the Mobile IP enabled core network. Forwarding delay is introduced temporarily when compensating against the time required to complete reflective handoff. This forwarding delay is 60 msec in the Mobiware to Cellular IP reflective handoff example as shown in Figure 11. The performance cost of reflective handoff is increased jitter which many adaptive applications are capable of absorbing.



(a) cellular IP-to-mobiware handoff



(b) mobiware-to-cellular IP handoff

Figure 11: Reflective Handoff Performance of a Video Application

Table 3 summarizes the characteristics of the signaling modules used in our experiments. The Mobiware signaling module has been implemented using CORBA technology (Iona's ORBIX), while the Cellular IP module does not use CORBA. Signaling modules can have varying sizes and memory footprints, which affect their downloading time and performance. The size of the Mobiware and Cellular IP signaling modules are 1054 Kbytes

and 69 Kbytes, respectively. Mobiware implements a complex QOS-adaptive mobility management architecture, while Cellular IP implements a simple in-band handoff scheme without QOS support.

The implementation of the Mobiware signaling module is based on a commercial Object Request Broker (ORB), which is not customized for wireless environments or for on-demand code loading. This results in a significant footprint and loading time (7 sec). Clearly, this result indicates that reflective handoff is not a practical solution for the current ORB and Mobiware signaling module, especially when one considers a 2 Mbps air interface. Using radios with speeds of 10 and 25 Mbps will reduce the loading times to 1.4 sec and 0.56 sec, respectively. Future work will include porting the programmable handoff architecture to a 'light-weight' microORB [32], which will further decrease the footprint of programmable signaling modules.

Our implementation of reflective handoff operates well in our testbed where signaling modules are loaded over slow time scales. Soft state timers associated with module caches operate over tens of minutes. We plan to investigate scalability issues associated with the reflective handoff scheme and its applicability to mobile devices with varying loading requirements and processing capabilities.

signaling module	*Size (Kbytes)*	*Loading time (sec)*
Mobiware	1054	7 ± 1
Cellular IP	69	0.4 ± 0.1

Table 3: Signaling Modules

6. Conclusion

In this paper we have presented a framework for the programmability of wireless access networks and described a proof of concept design, implementation and evaluation of programmable handoff architecture. Two new services were designed and programmed using the programmable handoff architecture: multi-handoff access network and reflective handoff services.

A service creation environment can construct handoff services through profiling and composition techniques. Our results indicate that the programmable handoff architecture can scale to support increased signaling load while achieving similar performance to native signaling systems discussed in the literature. Our platform allows systems designers to architect their own handoff control architectures and program them using a set of well-defined APIs and objects. The platform is capable of supporting multiple handoff control architectures over the same physical programmable access network.

In our work we are investigating the notion of making mobile networks more programmable for the deployment of new services. We have argued that mobile services can be constructed in a more flexible manner and introduced more rapidly than they can today. In this paper, we have investigated how our methodology can be used to construct services with a focus toward 'programming' new handoff control planes. Clearly many new services can be considered and we only present a small but interesting set in this work. The multi-handoff access network service and reflective handoff service use the same profiling, service creation and programming environments. This indicates that our approach is more generally applicable to the introduction of new in mobile networks.

Acknowledgement

This work is supported in part by the National Science Foundation (NSF) under CAREER Award ANI-9876299 and with support from Intel, Nortel Networks and NEC under grants for programmable mobile networking. The authors would also like to thank Raymond Liao for his contribution to this work.

References

[1] K. Buchanan, R. Fudge, D. McFarlane, T. Phillips, A. Sasaki, and H. Xia, "IMT-2000: Service Provider's Perspective", *IEEE Personal Communications Magazine*, August 1997

[2] P. Bagwat, C. Perkins, and S. Tripathi, “Network Layer Mobility: an Architecture, and Survey”, *IEEE Personal Communications Magazine*, June 1996.
[3] D. Raychaudhuri, “Wireless ATM Networks: Architecture, System, Design and Prototyping”, *IEEE Personal Communications Magazine*, August 1996.
[4] A. A. Lazar, “Programming Telecommunications Networks”, *IEEE Network*, October 1997.
[5] A. Tennehouse, and D. Wetherall, “Towards an Active Network Architecture”, *Multimedia Computing and Networking*, San Jose, CA, 1996.
[6] J. Mitola, “Technical Challenges in the Globalization of Software Radio”, *IEEE Communications Magazine*, February 1999.
[7] V. Bose, M. Ismert, W. Wellborn, and J. Guttag, “Virtual Radios”, *IEEE Journal on Selected Areas in Communications*, Special Issue on Software Radios, 1998.
[8] G. Bianchi, and A. T. Campbell, "A Programmable Medium Access Controller for Adaptive Quality of Service Control", *IEEE Journal of Selected Areas in Communications (JSAC)*, Special Issue on Intelligent Techniques in High Speed Networks, March 2000.
[9] M. E. Kounavis, A. T. Campbell, G. Ito, and G. Bianchi, “Accelerating Service Creation and Deployment in Mobile Networks”, *Third International Conference on Open Architectures and Network Programming, (OPENARCH’00)*, Tel-Aviv, Israel, March 2000.
[10] M. E. Kounavis, A. T. Campbell, G. Ito, and G. Bianchi, “Supporting Programmable Handoff in Mobile Networks”, *Sixth International Workshop on Mobile Multimedia Communications (MoMuC’99)*, San Diego, CA, November 1999.
[11] M. C. Chan, and A. A. Lazar, “Designing a CORBA-based High Performance Open Programmable Signaling System for ATM Switching Platforms”, *IEEE Journal on Selected Areas in Communications*, September 1999.
[12] A. T. Campbell, M. E. Kounavis, and R. R.-F. Liao, "Programmable Mobile Networks", *Computer Networks and ISDN Systems*, April 1999
[13] A. G. Valko, J. Gomez, S. Kim, and A. T. Campbell, "On the Analysis of Cellular IP Access Networks", *Sixth IFIP International Workshop on Protocols for High Speed Networks* , Salem, 25-27 August 1999.
[14] V. Bose, D. Wetherall, and J. Guttag, “Next Century Challenges: Radioactive Networks”, *Fifth ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM’99)*, Seattle, Washington, 1999.
[15] J. Mitola, “Cognitive Radio for Flexible Mobile Multimedia Communications”, *Sixth International Workshop on Mobile Multimedia Communications (MoMuC’99)*, San Diego, CA, November 1999.
[16] R. R.-F. Liao, and A.T. Campbell, "On Programmable Universal Mobile Channels in a Cellular Internet", *Fourth ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM'98)* , Dallas, October 1998.
[17] R. R.-F. Liao, P. Bocheck, A. T. Campbell, and S.-F. Chang, “Utility-based network adaptation in MPEG-4 systems”, *Ninth International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV’99)*, Basking Ridge, New Jersey, 1999.
[18] A. B. Kulkarni, and G. J. Minden, “Active Networking Services for Wired/Wireless Networks”, *Eighteenth Annual Joint Conference of the IEEE Computer and Communication Societies (INFOCOM’99)*, New York, March 1999.
[19] The ARRCANE Project, http://www.docs.uu.se/arrcane/
[20] N. D. Tripathi, J. H. Reed, and H. F. Vanlandingham, “Handoff in Cellular Systems”, *IEEE Personal Communications Magazine*, December 1998.
[21] D. J. Goodman, “Wireless Personal Communications Systems”, *Addison-Wesley Wireless Communications Series*, 1997.
[22] H. J. Wang, R. H. Katz, and J. Giese, "Policy-Enabled Handoffs Across Heterogeneous Wireless Networks," *Second IEEE Workshop on Mobile Computing Systems and Applications (WMCSA '99)*, New Orleans, LA, February 1999.
[23] R. Ramjee, T. La Porta, S. Thuel, K. Varadhan, “HAWAII: A Domain-based Approach for Supporting Mobility in Wide-area Wireless Networks”, *Seventh International Conference on Network Protocols*, Toronto, Canada, 1999.

[24] S. Seshan, H. Balakrishnan, and R. H. Katz, “Handoffs in Cellular Networks: The Daedalus Implementation and Experience", *Kluwer International Journal on Wireless Communication Systems*, 1996.
[25] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, “Design Patterns: Elements of Reusable Object-Oriented Software”, *Addison-Wesley Professional Computing Series*, 1995.
[26] The Object Management Group, www.omg.org.
[27] M. E. Kounavis, and A. T. Campbell “The Metabus: Breaking the Monolith of the Software Bus”, *Technical Report*, Center for Telecommunications Research, Columbia University, April 1999.
[28] B. C. Smith, “Procedural Reflection in Programming Languages”, *PhD Thesis*, Massachusetts Institute of Technology, 1982.
[29] R. Caceres, and V. Padmanabhan, “Fast and scalable wireless handoffs, in support of mobile Internet audio”, *Mobile Networks and Applications*, 1998.
[30] J. Der Merwe, S. Rooney, I. Leslie, and S. Crosby, “The Tempest, A Practical Framework for Network Programmability”, *IEEE Network*, May 1998.
[31] Cellular IP, source code distribution: comet.columbia.edu/cellularip.
[32] Object Management Group, “Minimum CORBA”, Joint Revised Submission, *OMG Document*, orbos/98-08-04 ed., August 1998.
[33] A. T. Campbell, M. E. Kounavis, D Villela, J. Vicente, H. De Meer, K. Miki, and K. Kalaichelvan, “Spawning Networks”, *IEEE Network*, August 1999.

Author Biographies:

Michael E. Kounavis (mk@comet.columbia.edu) is a Ph.D Candidate and Graduate Research Assistant at COMET Group at 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 automating the creation deployment and management of network architectures. Over the past years he has been actively involved in mobile network programmability and the realization of adaptive middleware for mobile networks.

Andrew T. Campbell (campbell@comet.columbia.edu) see editorial

Get Ito (itogen@sns.abk.nec.co.jp) is with NEC Corporation, Japan. He has been working on switch software design and development for the last four years years. He spent 1998-1999 as been a Visiting Researcher at the COMET group at Columbia University, working under the supervision of Prof. Andrew T. Campbell on programmable handoff.

Giuseppe Bianchi received the Laurea degree in electronics engineering from Polytechnico di Milano, Milano, Italy in 1990 and a specialization degree in information theory from CEFRIEL, Milano, Italy, in 1991. He was a Researcher at CEFRIEL from 1991 to 1993 and an Assistant Professor at the Politechnico di Milano from 1993 to 1998. He has been an Associate Professor at the University of Palermo, Palermo, Italy since 1998. He spent 1992 as a Visiting Researcher at Washington University, St. Louis MO, and 1997-1998 as a Visiting Researcher at the Center for Telecommunications Research, at Columbia University, New York, NY. His research interests include design and performance evaluation of broadband and wireless networks and systems.