iMobile EE – An Enterprise Mobile Service Platform

Yih Farn Chen, Huale Huang, Rittwik Jana, Trevor Jim, Matti Hiltunen,
Sam John, Serban Jora, Radhakrishnan Muthumanickam, and Bin Wei

AT&T Labs - Research
180 Park Avenue
Florham Park, New Jersey 07932, USA
{chen, huale, rjana, trevor, hiltunen, sam_john, jora, rmuthu, bw}@research.att.com

Abstract

iMobile[1] is an enterprise mobile service platform that allows resource-limited mobile devices to communicate with each other and to securely access corporate contents and services. The original iMobile architecture consists of devlets that provide protocol interfaces to different mobile devices and infolets that access and transcode information based on device profiles. iMobile Enterprise Edition (iMobile EE) is a redesign of the original iMobile architecture to address the security, scalability, and availability requirements of a large enterprise such as AT&T. iMobile EE incorporates gateways that interact with corporate authentication services, replicated iMobile servers with backend connections to corporate services, a reliable message queue that connects iMobile gateways and servers, and a comprehensive service profile database that governs operations of the mobile service platform. The iMobile EE architecture was also extended to provide personalized multimedia services, allowing mobile users to remotely control, record, and request video contents. iMobile EE aims to provide a scalable, secure, and modular software platform that makes enterprise services easily accessible to a growing list of mobile devices roaming among various wireless networks.

1. Introduction

With the advances of wireless networking technologies and mobile devices, enterprises are looking for mobile solutions that empower their employees to work more productively while on the road. This paper describes iMobile Enterprise Edition (EE), a project that addresses research issues in building a mobile service platform that delivers end-to-end mobile solutions to large enterprises.

Several key issues arise in providing such an enterprise-level platform:

- **Scalable Services:** The platform must be able to handle a large number of service requests concurrently coming from various wireless and landline networks. Our initial aim is to handle up to 10,000 users and to ramp up to 200,000 users to accommodate the largest enterprises. The traffic mix may change dynamically and may include short

[1] iMobile was the working title of our original research project; the platform and its network of distributed components have recently been renamed to AT&T Mobile Network.

messages from cell phones, instant messages, emails, and HTTP, WAP, and GSM short message service (SMS) requests.

- **Corporate Authentication**: Since mobile users frequently have access only to the public Internet or wireless networks, the platform must provide a gateway or tunneling solution to allow mobile employees to access corporate information on their intranet. This requires the platform to interact with corporate authentication services (such as Microsoft Windows domain authentication or RADIUS, Remote Authentication Dial-In User Service [20]).
- **Security Policy**: Since the platform will act on behalf of the mobile user to access corporate resources, the platform must obtain authorization based on the user identity, channel security, and corporate policy before accessing corporate databases, directories and email servers, etc. The platform should log resource accesses and operation details for accounting purposes.
- **Dependability:** The platform must be able to reconfigure itself dynamically when certain machines fail or become overloaded and continue to deliver services satisfying appropriate performance guarantees.

This paper is organized as follows. In Section 2, we describe the logical view of the iMobile architecture, which reflects the actual implementation of the iMobile standard edition [1]. Section 3 describes how the original architecture was changed in iMobile EE to provide enterprise-level services, taking into account the issues described above. Selected issues in the design of iMobile EE are discussed in Section 4, while Section 5 provides concrete examples of complete applications implemented using iMobile EE as well as initial performance results. Section 6 discusses related work and Section 7 concludes with summary and future work.

2. iMobile: Logical Architectural View

As shown in Figure 1, the original architecture of iMobile [1] implements three key abstractions: devlets, infolets, and applets. A devlet is a driver or a protocol adaptor attached to the proxy that receives and sends messages through a particular protocol (e.g., Instant Messaging, Short Message Service, WAP, HTTP, Email) running on a mobile device. An infolet is responsible for creating an abstract view of an "information space" using an appropriate protocol (e.g., HTTP for the Web, JDBC for database access, X10 for home network control, and LDAP for directory services) to connect to a backend server. An applet implements the application logic by post-processing information obtained by the various infolets. The core of iMobile, the *proxy engine*, implements the basic framework for hosting devlets, infolets, and applets. It supports user and device profiles for personalization, performs appropriate content transcoding and adaptation, and invokes the proper applets and infolets to answer requests from a devlet. The iMobile architecture allows new mobile devices and protocols to be added to its framework without requiring any changes in the operational logic for information retrieval and delivery. iMobile effectively acts like a personal agent on the network that enables limited devices to access personalized mobile services.

iMobile's modular architecture allows it to incorporate new devices and technologies as they become available. Unfortunately, the original iMobile architecture was not scalable or reliable, because the proxy acted as a middleman in every message transaction. Moreover, it was not integrated with corporate authentication services. In the next section, we describe how we designed the new iMobile Enterprise Edition to meet the challenges of scalability, reliability, and security.

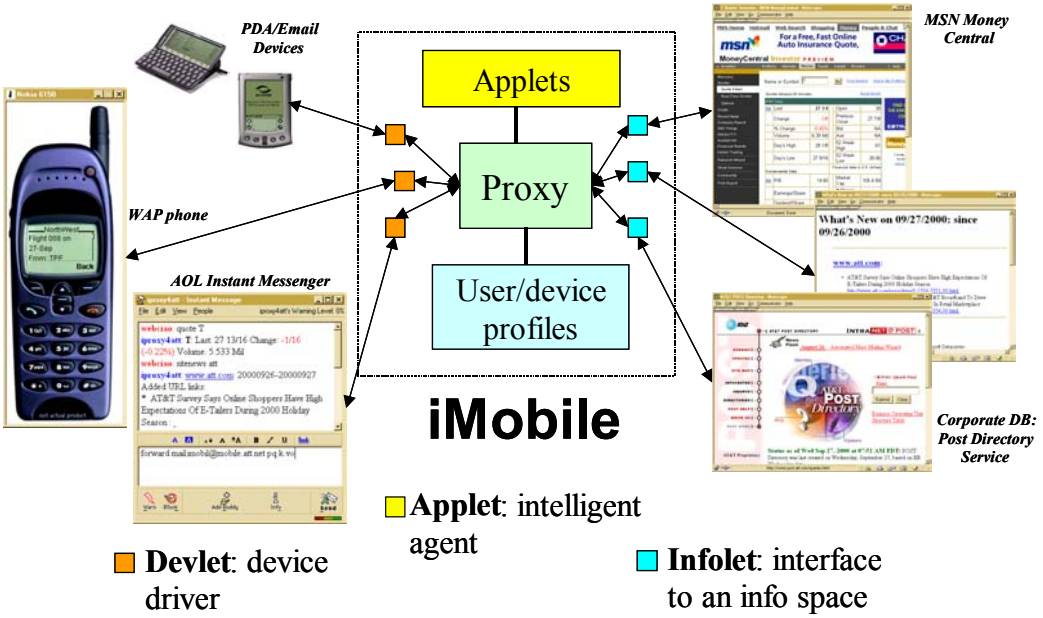

Figure 1. Original iMobile Architecture (Standard Edition)

3. iMobile Enterprise Edition

To address the needs of the mobile enterprise users, we redesigned the original architecture in several ways while keeping the original spirit of flexibility and modularity in iMobile. The new enterprise architecture is shown in Figure 2. A mobile device always interacts with an iMobile gateway to access iMobile services. A gateway authenticates a mobile user and puts each service request on the message queue. Typically, a cluster of iMobile servers can then pick up messages from the message queue in a round-robin fashion. Each server hosts a set of infolets for backend connections to corporate services. Both iMobile gateways and servers interact with the service profile database, which governs the transcoding and content delivery processes. The iMobile architecture conforms to design specifications already popular in Java enterprise applications, such as JMS[4], JDBC [8], JNDI[9], Servlet[39], WebDAV[22], XSLT[15], and XML. This allows us to interface with a broad spectrum of products used in the enterprise world; ideally, iMobile EE would simply be an add-on to the existing infrastructure in an enterprise.

The following subsections describe each component of iMobile EE in greater detail.

3.1 Gateways

The original iMobile devlets are replaced by *gateways*, with each gateway hosting one or more devlets. Each devlet implements the corresponding protocol interfaces. The number of gateways can be dynamically adjusted depending on the traffic load. Each gateway implements a protocol and authenticates the mobile user against iMobile's corporate authentication service. In the following, we will briefly describe some of the gateways in iMobile EE and demonstrate how the same AT&T Directory Service (Post) can be accessed from different gateways through the iMobile LDAP infolet.

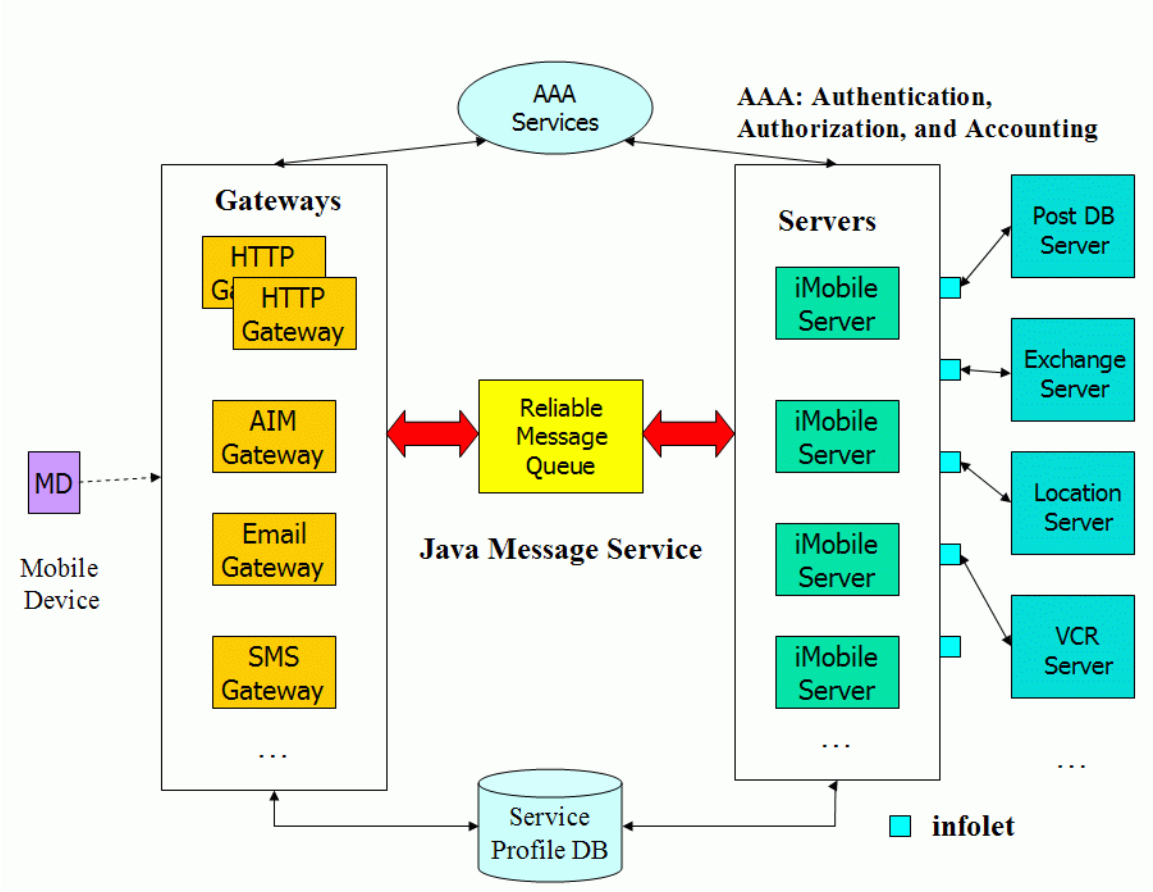

Figure 2. Architecture of the iMobile EE Service Platform

HTTP/WAP Gateway

The HTTP gateway handles HTTP service requests and associated user authentication (see Section 4.4) from mobile devices. It also supports WAP[2] [17], an open specification that offers a standard method to access Internet based content and services from wireless devices such as mobile phones and PDAs (Personal Digital Assistants). Figure 3 shows how an iMobile user would access the Post Directory Service through the iMobile HTTP gateway. Both regular web browsers and WAP browsers on mobile devices share the same iMobile HTTP gateway implementation; however, our deployment of the HTTP gateway for WAP browsers is slightly different due to security concerns described in detail in Section 4.5.

The HTTP gateway is implemented as a set of standard Java Servlets [39], platform-independent extensions of Web servers, running under any compatible Servlet container. It has been used under the Jakarta/Tomcat server [2] and Oracle's OC4J Web application container [32]. Because of this type of integration, a distinctive implementation aspect of the HTTP/WAP gateway is its resource allocation: here the Web container takes charge of the thread management and other

[2] The WAP forum has been renamed to Open Mobile Alliance, http://www.openmobilealliance.org/ .

aspects while in all the other iMobile gateways it is under full control of the respective iMobile box.

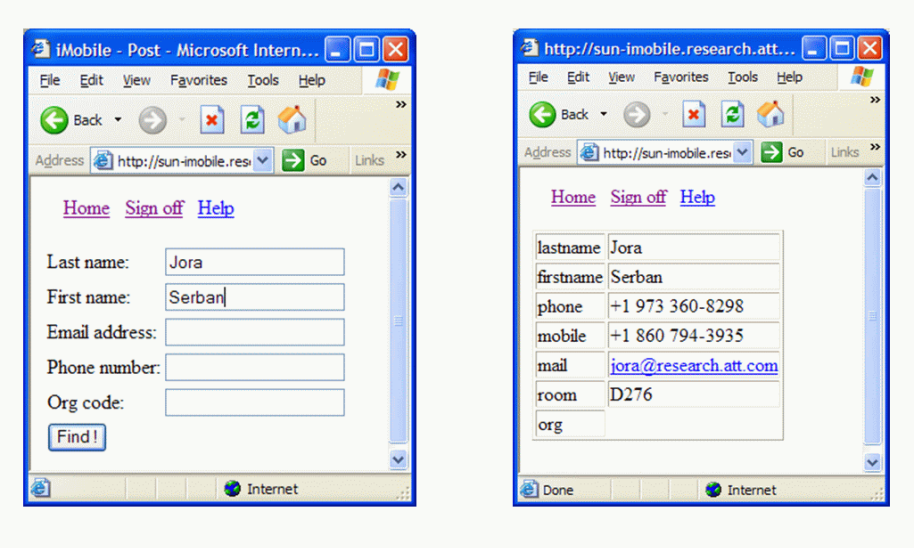

Figure 3. Accessing Employee Data in AT&T's Post Directory Service from the iMobile HTTP Gateway through the LDAP Infolet

Email Gateway

The iMobile email gateway allows mobile users to access corporate services by sending service requests as emails to an iMobile email service account. The requester's email address must map to a unique iMobile user ID according to the service profile database before a request can be authorized. The iMobile gateway periodically checks all email service accounts (using IMAP or other mail protocols) that it monitors for service requests and returns results as emails to the requesting mobile users. Note that a mobile user would need a secure device to send corporate email. Blackberry devices allow end-to-end encryption between the device and the corporate Microsoft Exchange server through the Blackberry Enterprise Server [27]. A mobile user can also use a VPN client such as Movian [36] on the mobile device to get back to the corporate intranet.

Figure 4 shows the typical architecture of an iMobile Email Gateway interacting with mobile users using always-on Blackberry email devices. For example, an iMobile/Blackberry user can send a service request in the form of firstname.lastname, such as "Serban.Jora", to **im-post@research.att.com,** an email account that represents the AT&T Post service. The iMobile mail gateway periodically polls that account, and sends any service request through JMS to an iMobile server, which then uses an LDAP infolet to retrieve the directory information from AT&T's Post server. The information is then converted into plain text for email delivery through the same mail gateway. The user will then receive an email that includes the directory information, such as phone number and email address of Serban Jora.

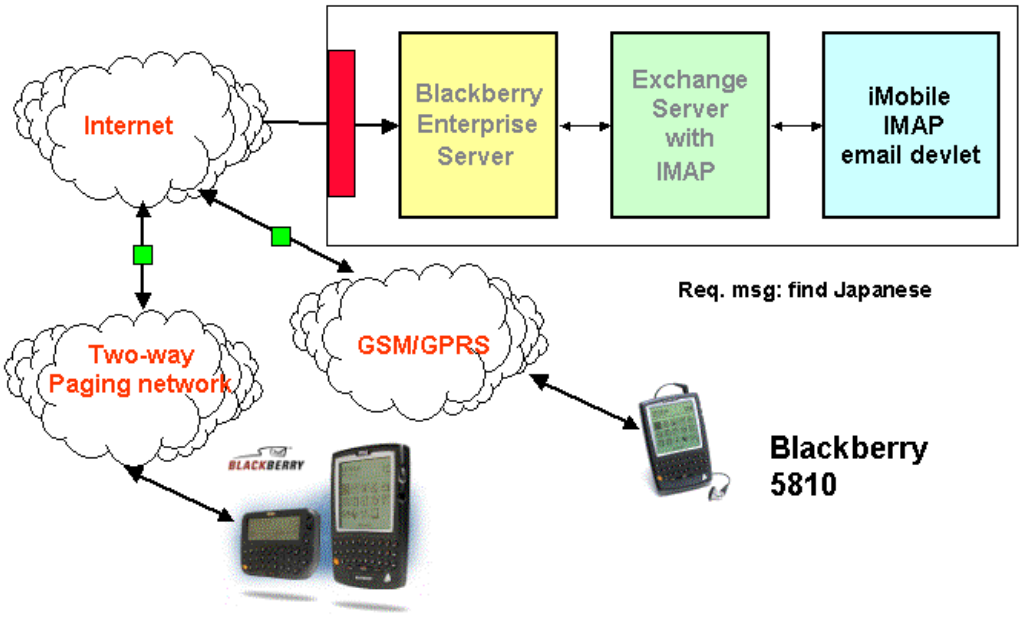

Figure 4. iMobile Mail Gateway

Note that it is relatively easy for an attacker to fake the sender's email ID and send email to an iMobile mail gateway on behalf of someone else who has registered with iMobile. In this case, iMobile will service the requests and send the responses back to the registered user (not the attacker). While this is annoying, it does not present significant security risks since the attacker still will not have access to corporate contents. While there is a potential for Denial of Service (DOS) attacks, it is an existing issue with or without the use of iMobile.

Instant Messaging Gateway

The Instant Messaging (IM) gateway includes an iMobile IM devlet that acts like an IM client, but interprets instant messages received as service requests and returns responses in instant messages. The IM gateway maps an IM screen ID to an iMobile ID before submitting a service request. We started with the AOL Instant Messaging (AIM) protocol, but due to the security limitations of the AIM channel (lack of encryption, insecure passwords, and server control), access to corporate data is prohibited. We have migrated to Jabber [33], which allows secure corporate instant messaging using SSL between instant messaging clients and the Jabber IM server, which we control. Figure 5 shows the architecture of an iMobile Jabber gateway and sample interaction screens on Post and Glossary services.

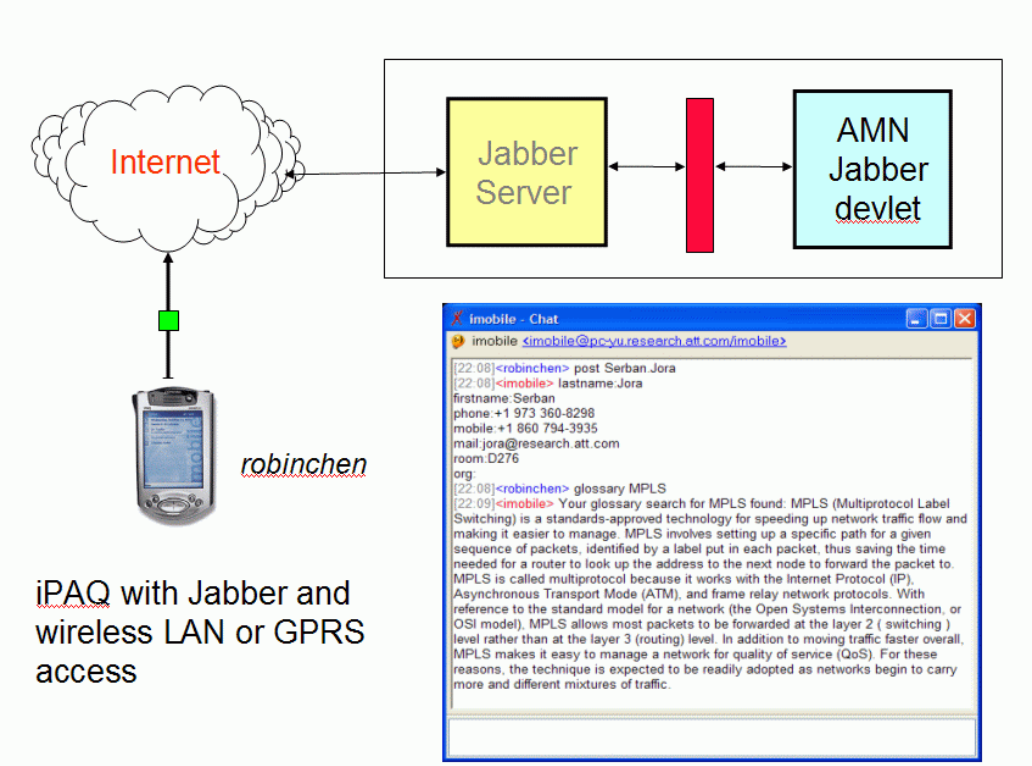

Figure 5. iMobile Jabber Gateway

SMS Gateway

The SMS gateway allows iMobile users to send a service request as a GSM short message to the iMobile SMS gateway (e.g., +1973960xxxx) and receive the response back as a short message, in a way very similar to the Email gateway. The SMS gateway communicates with the iSMS server[11], which can send and receive short messages through a GSM phone connected to its communication port. The SMS gateway can connect to multiple iSMS servers to handle larger volumes of SMS traffic, but this is not a true scalable solution. We are migrating to a new SMS gateway implementation based on the SMPP (Short Message Peer-to-Peer) protocol [34].

3.2 Reliable Message Queue

In order to have replicated iMobile servers for scalable services, we decoupled the iMobile gateways and servers by using the Java Message Service (JMS) [4]. The JMS API, supported by several JMS providers, supports both point-to-point and publish/subscribe messaging between distributed applications. iMobile uses the point-to-point (PTP) approach, in which an application is built around the concept of message queues, senders, and receivers. Each message is addressed to a specific queue, and receiving clients extract messages from the queue that holds their messages.

iMobile allocates three queues in the JMS provider: *imobile.request, imobile.reply*, and *imobile.routed*. Multiple gateways can place service requests on the *imobile.request* queue, while multiple iMobile servers can pick up messages from the same queue. Similarly, servers place responses on the *imobile.reply* queue, but only the originating gateways can pick up the

messages. The routed queue is used when an interactive session (discussed in Section 4.1) is involved so that subsequent requests from the same gateway are always routed to the same server.

We have experimented with three JMS providers: IBM MQ Series [5], Fiorano [6], and SonicMQ [21], with satisfactory performance results (see Section 5.3). Moreover, the design of the iMobile gateways and servers allows us to *hot swap* from one JMS provider context to the other without shutting down the platform. The use of an enterprise message service also allows us to build a scalable platform with high availability and flexibility in both gateway and server configurations. Currently, a JMS provider is rarely overloaded even under stress testing, but a cluster of JMS providers can be used if necessary to increase the availability and scalability of the messaging service itself.

3.3 iMobile Servers and Infolets

Each iMobile server hosts a set of infolets and applets that connect to backend databases, mail, directory, content, video, and home network servers. While each infolet typically provides access to one information source, an applet implements more complex application logic by invoking multiple simple infolets. For example, the *find* applet, which allows you to find a store near where you are, invokes the *location* infolet, which interfaces with a location determination technology, and then invokes a *yellow page* infolet to report the store near you.

Each infolet typically performs the following functions:

- *Information Access*: It provides a protocol interface to an information space. For example, JDBC is used to access a corporate database, LDAP is used to access a directory, HTTP is used to access Web content, and X10 is used to control home devices like X10 cameras, etc.
- *Transcoding*: The returned raw content (if there is any) is then transcoded according to one of the MIME types and display sizes acceptable by the receiving device (see Section 4.2 for details).

A large number of different infolets have been implemented for various iMobile experiments and demonstrations, including the following examples:

- The *post* infolet retrieves the contact information of a named employee from our company LDAP database.
- The *glossary* infolet retrieves the definition of technical terms (e.g., WSDL, MPLS) from a corporate information database.
- The *exchange* infolet (described in Section 5.1) provides access to the user's mailbox, calendar, and contact information stored in a Microsoft Exchange server.
- The *drawings* infolet retrieves technical design drawings, at different levels of resolution according to the device.
- The *vcr* infolet (described in Section 5.2) allows a user to remotely control (change channels, start/stop recording, replay) a VCR controlled by the iMobile system.
- The *weather* infolet, given a postal zip code, retrieves the current weather for the given location.
- The *quote* infolet retrieves the current stock quote for the given stock symbol.

Note that weather and quote infolets typically only access information from a company portal with license agreements from content providers - as the information sources are not owned by the corporation itself.

Even a relatively simple infolet can become a powerful business tool because the iMobile architecture allows access to these infolets through various devices. For example, a traveling corporate executive can get the contact information of any of the corporation's employees through his cell phone using the iMobile SMS gateway.

3.4 Profile Database

We have extended the original user and device profiles to a relational database with an Entity-Relationship data model [25] that includes services, users, devices, permissions, protocols, authoritative domains, etc., that govern all aspects of the operation of iMobile. iMobile administrators can populate the database during the system startup or the user provisioning process. We have experimented with both the Cloudscape [7] database from Informix (now owned by IBM) and Oracle 8, but can migrate to any other relational database system easily since we use SQL, the standard query language for relational databases.

Figure 6 shows a subset of the entities and relationships in our data model. Each iMobile user has a unique user id in the iMobile system, but may own multiple devices and accounts (such as a Jabber ID or an Exchange email account) for access to iMobile and various back-end services. Only a single user is allowed to own each device or account. A user also has access to a set of resources (X10 cameras, VCR, etc.), but multiple users can share certain resources. Similarly, a user is allowed to access multiple services through corresponding accounts. For example, the Windows domain account can be used to access the inbox, calendar, and contacts in the Microsoft Exchange 2000 Server. When iMobile is set up to use Single Sign On (SSO), a user will be authenticated just once. An iMobile server will pick up the request from the message queue and check to see whether it requires access to an Exchange inbox, calendar, or contact list. If so, it retrieves the encrypted Windows domain authentication information from the profile database and presents it to the Exchange server.

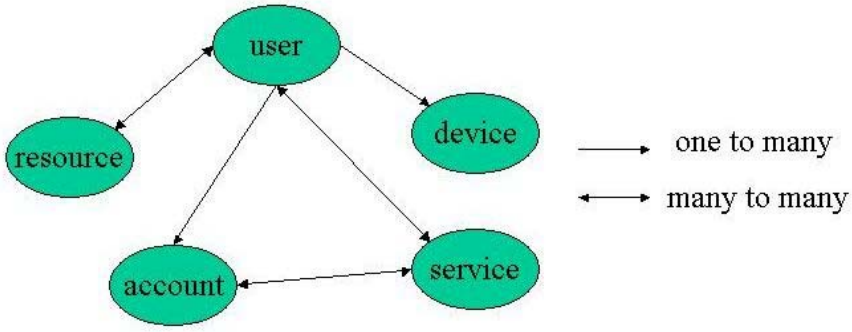

Figure 6. A Subset of the Data Model for the iMobile Profile Database

4. Discussions on Selected issues in iMobile EE

Due the space limitations, we cannot cover all the details of the system design and operation, but in this section we outline some interesting issues in iMobile EE.

4.1 Session Management

In most of the discussion so far, we have assumed that any iMobile server can pick up any iMobile service request placed by a gateway. For certain infolets that must maintain context during a dialog session, iMobile must make sure that the requests that belong to the same session are routed to the same server. Figure 7 provides details of session management in iMobile. The entities involved are gateways, front- and back-end sessions, JMS-based transport entities, and front- and back-end dispatchers.

Upon successful completion of user authentication, the gateway creates a new front-end session with a unique identification tag; this session is valid during the lifetime of the user interaction with iMobile through this gateway. Each protocol-specific request is transformed into a standard iMobile request that carries the front-end session identification tag. If a particular service (such as the dialog service *Eliza* [19]) requires session management for subsequent requests, then a back-end session is created and is identified by the same identification tag used in the front-end session.

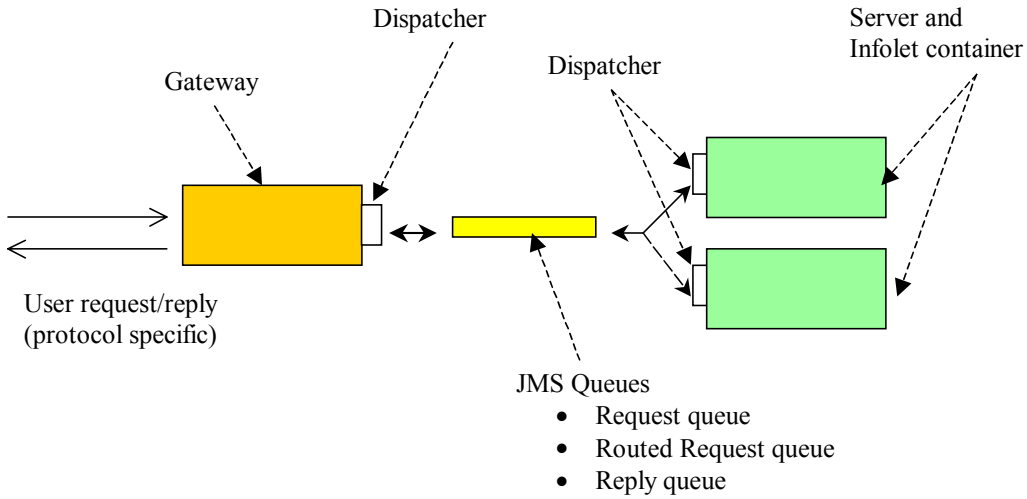

Figure 7. Session Management

The back-end session is valid for the server in which it was created and it is not available for use by other servers in the iMobile server cluster. For this reason, we have to guarantee that subsequent service requests of the same type from the same user are sent to the same server. This is achieved by enabling the request routing feature during the first reply, following the creation of a back-end session.

The necessary information for routing, namely the server name and the routed-request queue identifier, is embedded by the server-side dispatcher into the reply message. This information is picked up by the front-end dispatcher and stored in the front-end session. Subsequent requests

generated under this front-end session are sent based on the routing information and are guaranteed to be picked up by the same server. This information is valid only while the requests are sent to the same service; it is destroyed as soon as the user invokes another service. This behavior is important because request routing interferes with the *round-robin* request distribution dictated by the queuing policy of JMS[3]. Moreover, system messages need to be exchanged between the gateway and the server to clean up back-end sessions when a front-end session is dismissed. Alternatively, the server that runs the back-end session can employ a time-out mechanism.

4.2 Transcoding

Since iMobile provides services to a wide range of devices with different capabilities (e.g., the screen size) and the data format supported (e.g., simple text or HTML), transcoding the information retrieved by the infolets into an appropriate format is crucial for iMobile. The information returned by an infolet is very infolet specific, that is, the result of a *quote* infolet is very different from the result of the *drawings* infolet. Therefore, only the infolet knows how the result should be transcoded for display on the user's device.

Currently infolets use two different approaches for transcoding. Some infolets generate the final result formatted for the user's device directly. Some infolets convert the raw information retrieved into a *transcodable object* that encapsulates the essential abstract information of the content or service. The transcodable object provides methods for transforming its contents based on the MIME type specified by the service request. We typically use transcodable objects for web content that comes from a data source that we do not control or have direct access to, such as stock quotes and language translation services that are outside of the corporate domain.

iMobile uses several forms of transcoders for different response data types:

- *XML transcoder*: XML has become increasingly popular as the return type of many Web services that support protocols like WebDAV [22]. We describe transcoding of XML files obtained from Microsoft Exchange inboxes, calendars, and address books in detail in Section 5.1.
- *Image transcoder:* iMobile itself does not have image adaptation tools, but we are experimenting with the *Image Adapt* tool from Newstakes.com [23] to adjust image sizes and quality according to device profiles.
- *Video transcoder:* The VCR infolet [12] of iMobile allows mobile users to remotely record TV programs from any mobile device. The video is stored on a VCR server in MPEG-2 and then transcoded to H.263 [24] for low-bit rate adaptive wireless delivery through a video server.
- *Generic HTML transcoder:* This transcoder takes any HTML page and converts it to a form suitable for display on mobile devices. The transcoder filters out complex objects such as JavaScripts, replaces embedded images with hyperlinks, and splits long HTML pages into several pages. It preserves most HTML forms and allows simple interactions even on small browsers on PDA's or WAP phones.

4.3 Dependability and Scalability

[3] In the latest version, there is no need for a separate queue for routed requests. The same queue is used together with a filter for incoming messages based on a JMS message header field.

When a corporate user relies on iMobile services for business critical functions, the iMobile service must be highly *dependable*, that is, fault tolerant, secure, and timely. Since iMobile EE is designed to be used by large corporations with many employees, it must also be *scaleable*, that is, able to be smoothly extended to handle any number of users while preserving dependability. In particular, the system must be able to provide satisfactory response time even if the corporation has 100 000's of active users.

Scalability

The iMobile EE architecture is specifically designed for scalability since it supports multiple gateways of each type as well as any number of iMobile servers. Since JMS queues are used to pass requests to the iMobile servers, new iMobile servers can be added when needed without interrupting the operation of the system. Currently, users directly address the iMobile gateways, and therefore some users have to be explicitly directed to use a new gateway if one is brought up during system operation. However, standard load distribution techniques, including layer 4 redirectors such as Nortel Networks' Alteon Web Switch [40] or Cisco's Content Services Switches [41], can be used to transparently direct user requests to actual gateway servers. We are exploring the use of such techniques.

In addition to the gateways and iMobile servers, the architecture depends on databases and the JMS, both of which are COTS (Commercial Off The Shelf) components with standard interfaces. As a result, if either a database or the JMS turns out to be a performance bottleneck, it can be replaced with a higher performance product without making any changes in the iMobile code. For example, Oracle provides database clusters that can handle large loads, and the IBM MQ cluster provides a high-throughput implementation of the JMS interface. Alternatively, it is also easy to distribute the load between two or more JMS services; and the user and device profiles, which change infrequently, could be replicated to multiple database replicas for performance. We have started using simulation and queuing analysis to determine a configuration of iMobile EE that can satisfy given performance requirements [31]. The simulation and queuing models allow us to determine how many gateways of each type, JMS servers, and iMobile servers are required to provide the required response time given a specified system load. Future work will include dynamic runtime resource allocation.

Fault tolerance

The iMobile EE architecture provides fault tolerance through redundancy of the gateways and iMobile servers, as well as through the use of standard database and JMS products. Even if some gateways or iMobile servers fail, the iMobile system can still provide service if the system is running enough replicas of each gateway and enough iMobile servers. In the worst case, the client has to address requests explicitly to a backup gateway. Database clusters such as those provided by Oracle can tolerate the failures of some of the database replicas without service interruption. The JMS standard provides fault-tolerance features including support for persistent messages (i.e., messages that are guaranteed to be preserved even if the JMS server fails) and transactions. Some JMS implementations, such as IBM MQ clusters, also ensure the availability of the JMS service in spite of failures of some of the JMS servers. Our current prototype implementation, however, does not yet use such fault-tolerant versions of databases and JMS providers.

The iMobile gateways and servers also provide fault tolerance. In particular, the runtime system in the gateways and iMobile servers monitors the devlets and infolets (as well as other components) running in each and automatically restarts any component that has failed. Furthermore, the gateways implement a retransmission mechanism that resends a request to the

JMS when an iMobile server does not respond in a timely fashion. This mechanism will mask the failure of an iMobile server while it is processing a request. The gateway can also automatically use a backup JMS server in case the primary JMS server fails, thereby masking the failure of a JMS server. We are working on extending iMobile with support for transactional all-or-nothing execution of requests, that is, the request is guaranteed to execute exactly once to completion or not at all. Furthermore, while the failure of an individual gateway or iMobile server does not necessarily make the iMobile system unable to serve client requests, the session state of existing front- and back-end sessions may be lost and the user has to reestablish connection. We are exploring the use of Java 2 Enterprise Edition (J2EE) facilities and fault-tolerant service platforms such as BEA WebLogic [35] to ensure the preservation of session states through failures.

Finally, it is worth noting that the fact that a user of the iMobile system has multiple devices – and thus multiple protocols and access networks – to access iMobile services makes the system tolerant to failures of access networks. For example, if the cellular phone network is down in some location, so that the user cannot use the browser on his PDA to access iMobile through WAP and CDPD/GPRS, he may be able to access iMobile services using email on his Blackberry device that uses the paging network that is independent of the cellular network.

4.4 Security and Authentication for the HTTP Devlet

For the HTTP protocol, secure remote access to iMobile gateway from outside the firewall can be currently done using one of the following three schemes: Absent, RADIUS, or VPN.

Absent scheme

The Absent system from AT&T Labs allows Web users to authenticate themselves from the public internet (e.g., from an internet cafe) by using a one-time password scheme for client authentication and SSL for confidentiality [3]. The one-time password is needed because Absent can be used to access the intranet from machines that are only partially trusted, e.g., public workstations. When iMobile users access the intranet from their personal mobile devices, however, the added security of one-time passwords is less important. Therefore, iMobile offers a client implementation within the handheld device (such as an iPAQ or a Palm device with a CDPD modem) that interacts with the challenge-response mechanism in Absent. The mobile user only needs to type in a (reusable) secret pass phrase—without manually going through a complex key computation and data entry process. Figure 8 shows three screenshots: a) a regular Absent authentication screen for mobile devices, which requires a companion piece of software and a secret pass phrase to compute the response, b) a simplified and customized one for quick access to iMobile, and c) the iMobile HTTP gateway screen. Note that Absent only allows users to get back to their intranet; iMobile must then map the user to either a unique user id in its own system or use a corporate authentication service such as the Windows domain authentication (as shown in Figure 8) through Microsoft Active Directory.

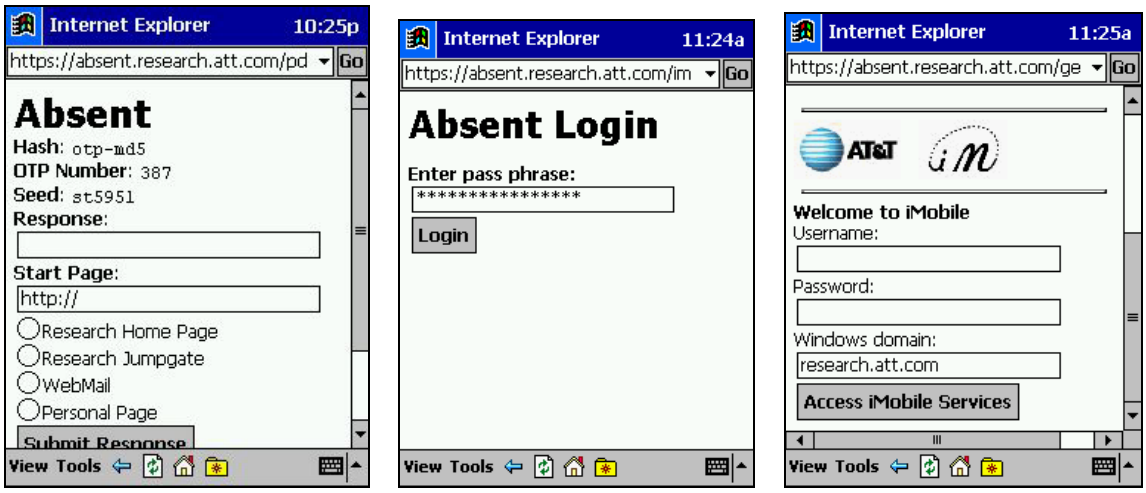

Figure 8. HTTP Gateway Access and iMobile Authentication

RADIUS scheme

iMobile also allows mobile users to dial up from an appropriate mobile device (such as a Visor Palm device with a GSM modem) to connect to a RADIUS server. In this case, iMobile will use the RADIUS user database instead of its own to perform authentication.

VPN (Virtual Private Network) scheme

Recently, VPN clients have extended their reach from laptops (such as Nortel's Contivity VPN client) to mobile devices (such as Certicom's Movian VPN client [36] that runs on iPAQ and Palm devices). A VPN client allows a device on the public Internet to have a secure tunnel back to the corporate intranet.

Regardless of the authentication scheme, access to several backend services (such as Microsoft Exchange Server or corporate database servers) may require additional authentication because the iMobile system is frequently outside the authoritative domains of certain backend services. To avoid multiple authentications, iMobile provides a Single Sign-On (SSO) option [18], a mechanism whereby a single user authentication permits a user to access all computers and systems where he has access permissions, without the need to enter multiple passwords. Single sign-on reduces human error and is therefore highly desirable. With secure wireless access, iMobile extends the traditional SSO capability to mobile users accessing enterprise resources from personal/handheld devices. iMobile, however, must guarantee the secrecy of user passwords after service provisioning, during which a mobile user provides account information through HTTPS.

4.5 Security for WAP Access

WAP can be used to access iMobile HTTP gateways from wireless devices such as mobile phones and PDAs. In the WAP model, the mobile device has embedded browser software that connects to a WAP Gateway (software infrastructure residing in the operator's network that optimizes the transmission of content for the wireless network) and makes requests for

information from web servers in the form of a URL. The content must be formatted suitably for the mobile phone's small screen and low bandwidth/high latency connection. Content is written in a markup language called WML (Wireless Markup Language) and is hosted on regular Web servers.

Accessing web contents from a WAP device actually involves two sessions: one between the mobile device and the WAP gateway and the other between the WAP gateway and the web server. Even though Wireless Transport Layer Security (WTLS) is used between the mobile device and the WAP gateway, there is still potentially a security gap between the WAP gateway and the Web server—unless both are hosted under the same authoritative domain or a private line (such as Frame Relay) is established to guarantee secrecy. Since this is not usually the case, we take measures to increase security.

Figure 9 shows how we deploy the iMobile HTTP/WAP gateway, which sits on our security perimeter. It interacts with the carrier's WAP gateway and iMobile's message queue. The iMobile gateway hosts WML files and it allows only HTTP traffic from the carrier's gateway, while the message queue allows only WAP service requests to be placed from the iMobile gateway. This separation shields the enterprise network from outside attacks aimed at the iMobile HTTP/WAP gateway; however, we will continue to limit sensitive information from being delivered through WAP until we can secure a secret channel between the carrier's WAP gateway and the iMobile gateway.

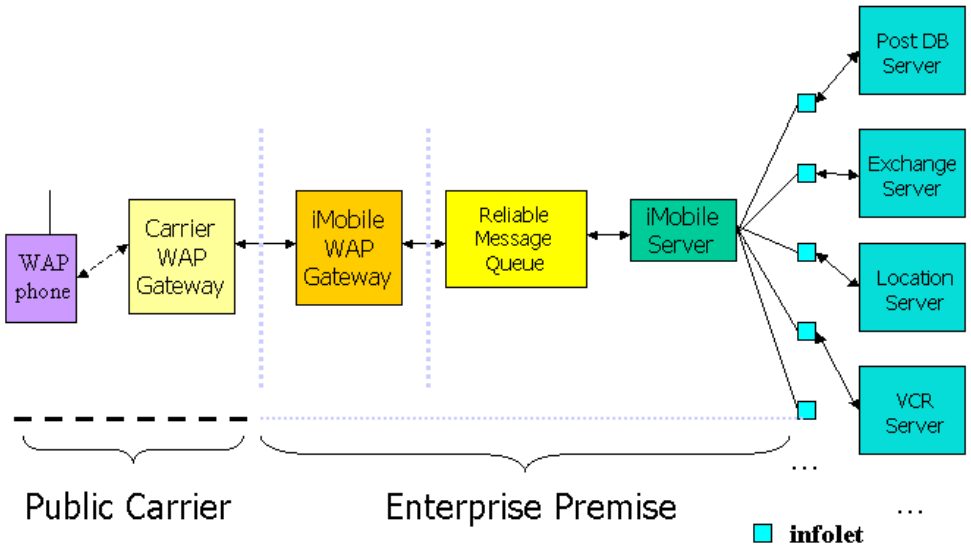

Figure 9. Secure Access to the WAP Gateway and Services inside the Firewall

5. Experiences and Performance

We have conducted iMobile service trials in Florham Park and several other companies and organizations. In the following, we report our experiences and performance evaluation of some selected services.

5.1 Exchange Services through iMobile

Microsoft Exchange 2000 supports remote access through the Web Distributed Authoring and Versioning (WebDAV) protocol [22]. WebDAV extends the HTTP 1.1 protocol to support remote collaborative authoring of network resources of any media type. Some methods added to WebDAV that use XML as the request and response format are: PROPFIND (property retrieval), PROPPATCH (set and remove properties), and MKCOL (make a new resource collection). The Exchange server also supports DAV Searching & Locating (DASL) [26], which employs HTTP 1.1 to form a lightweight search protocol to transport queries and result sets. It also allows clients to make use of server-side search facilities using the SEARCH method.

iMobile Exchange infolets use these HTTP extensions to query, search, and update inboxes, contacts, and calendars in the Exchange 2000 server. For example, the following PROPFIND method will return the Sender, Subject and Date Received in all the messages in the Exchange inbox of a user named Huale:

```
PROPFIND /exchange/huale/inbox HTTP/1.1
Content-Type: text/xml; charset="utf-8"
Content-Length:XXX
Depth: 1

<?xml version="1.0" encoding=\"utf-8\" ?>
<D:propfind xmlns:D="DAV:" xmlns:E="urn:schemas:httpmail:">
<D:prop>
<E:sender/>
<E:subject/>
<E:datereceived/>
</D:prop>
</D:propfind>
```

Once the XML response is returned, the Exchange infolet then invokes Xalan [14], an XSLT [15] stylesheet processor, to transcode the XML content to the appropriate MIME type for the receiving device. Figure 10 shows Microsoft Exchange inbox displayed on different form factors/devices.

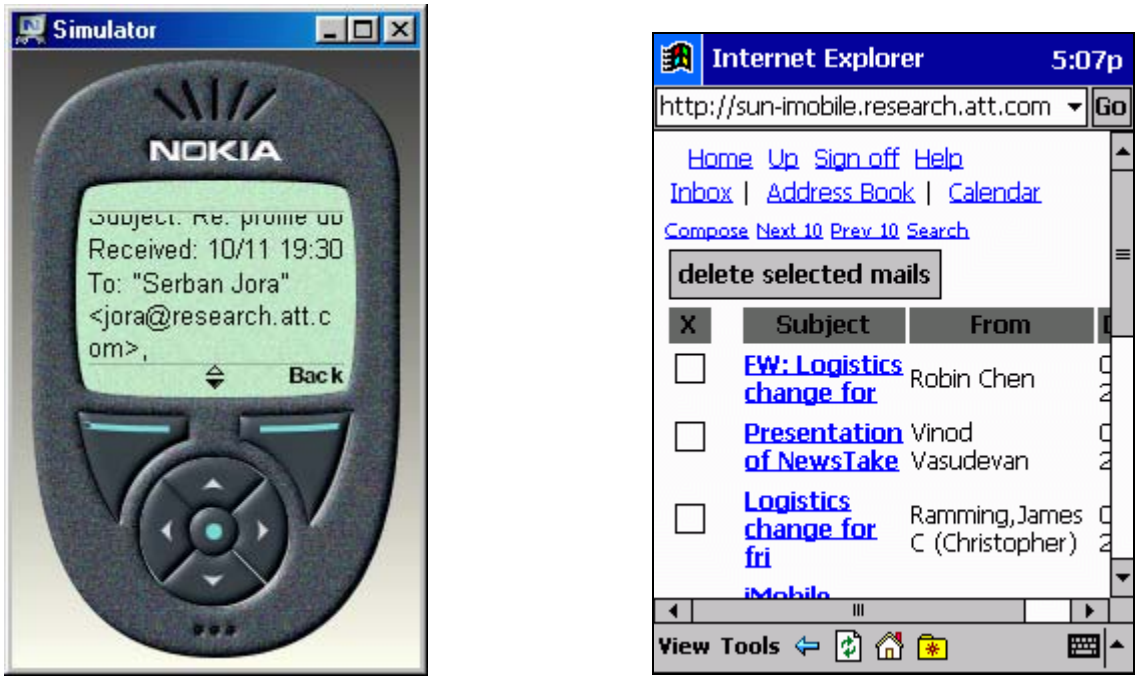

Figure 10. Messaging application (Microsoft Exchange 2000) delivered via iMobile to iPAQ and mobile phones.

5.2 Multimedia Services through iMobile

iMobile also provides personalized multimedia services [12]. It enables a mobile user to remotely record video programs, control cameras, and request the delivery of pre-recorded or live video content to his or her own mobile device. iMobile authenticates users who send service requests from various mobile devices, transcodes video content based on user and device profiles, and authorizes the delivery of content from the iVideo [13] media server to the proper client device. The media server adapts automatically to the fluctuations of the wireless channel conditions for reasonable viewing on the client device. The mobile service platform essentially manages the control path, while the media server handles the actual content delivery. We have successfully integrated iMobile and iVideo and conducted experiments on wireless LAN and Cellular Digital Packet Data (CDPD) networks.

Figure 11 shows remote control and video delivery to wirelessly connected mobile devices from an AIM client. The user first sets the channel on the VCR server to 33 (CNN), then requests a 20-second segment to be recorded in a file called cnn-news. He then verifies that his tv.ip property in the profile matches the IP address of his iPAQ and requests the delivery of pre-recorded video, live TV, or camera views (as shown on the right of Figure 11) to his iPAQ device.

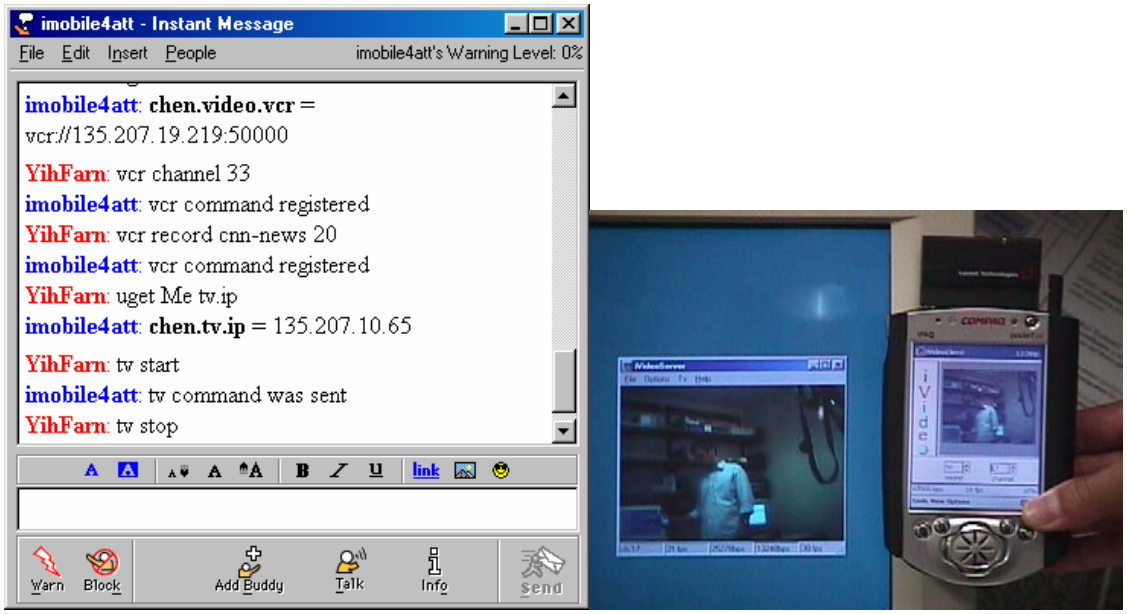

Figure 11. Remote Control of Video Recording and Delivery through an Instant Messenger

Currently, iMobile controls multimedia delivery from dedicated servers. This was under the assumption that the video servers are personal and will be under the control of a few people usually at a home or in a small business environment. Even though the access authorization of these video sources is controlled by iMobile, it is essentially still operating in a peer-to-peer mode between the mobile device and the video server. The move to a multipoint-to-multipoint distribution will greatly enhance the availability of the video sources to end clients. We plan to migrate the current architecture to support multicast groups and thus allow end users to switch between multicast groups. The MBONE architecture is better suited for this requirement [18]. iMobile will play a crucial role again in user authentication and content authorization.

5.3 Performance Experiments

Our initial load testing experiment includes three gateways and three servers running on various Red Hat Linux, SGI Irix, Sun Solaris and Microsoft Windows machines. We started 120 concurrent threads that generated load for the three gateways. Each thread executes 50 service requests with a random delay between requests of 0 to 20 seconds. The iMobile gateways and servers were able to finish all 6000 requests in 9 minutes. We first tested the round-trip delay of each request between the client and the iMobile server using the echo infolet, which simply responds with whatever the mobile user sends without invoking any external service. The delay was in the range of 69 ms (milliseconds) to 204 ms with an average of 105.35 ms. The following table gives more complete test results based on different kinds of service requests.

Queries	Num. Gateways	Num. Servers	Num. Threads	Num. Req. Per Thread	Req. Arrival Interval (sec)	Avg. Round Trip Delay (ms)
echo	3	3	120	50	0-20	105.35
post	3	3	30	20	0-20	715.03
quote	3	3	30	20	0-20	423.30
exchange	3	3	30	20	0-20	931.60

We also measured the response time of the different system components by instrumenting the gateways, iMobile servers, and an http test client. The gateways were instrumented to measure the response time from the time the request is sent to the JMS until a response is received from the JMS. The iMobile servers were instrumented to measure the execution time of the infolet. Finally, the http client was instrumented to measure the response time as seen by the client. The following table gives the average response times from a two-hour test run on a lightly loaded system. The table shows that the response time is dominated by the infolet execution time. This is to be expected because the infolet typically interacts with external services potentially over the Internet.

	Response Time (ms)		
infolet	Client	Gateway	Infolet
browse	390	326	251
quote	181	160	119
stocksymbol	169	149	119
find	297	260	211
weather	956	932	891

6. Related Work

The concepts of middleware service platforms and network edge services have received a lot of attention in the standards bodies, academia, and industry alike. In the following, we discuss work going on in these different sectors.

A number of related efforts are working on content transcoding, which is becoming increasingly important since users are accessing different information sources using devices with limited resources and using bandwidth-limited wireless networks. To perform transcoding, the system must be provided a device profile that describes the characteristics (size, format, etc.) of the receiving device. The W3C Device Independence working group is working on a structured and universal format CC/PP (Composite Capabilities/Preferences Profiles), which allows a client device to tell an origin server or proxy about its profile [28]. We are currently monitoring the progress of this protocol and may migrate to its format when it receives wide support from the industry. Examples of a project and a product that address transcoding are the Apache Cocoon project that allows the automatic generation of HTML, PDF, and WML (for WAP devices) files through the processing of statically or dynamically generated XML files using XSL and XSLT [29] and the Cisco Content Transformation Engine (Cisco CTE 1400) that provides enterprises with a high-performance, appliance-based solution that delivers back-end content to a variety of mobile devices.

There are several companies that focus on enabling enterprise mobile applications. They come in with the perspectives of software, networking, and database vendors. Most of the major mobile phone manufacturers provide mobile services platforms that allow corporate clients to access a limited set of services (e.g., email and calendar) through both their mobile phone and a web browser on a desktop. An example of such a platform is Nokia One Mobile Connectivity Service[37] that supports SMS, WAP, voice, and HTTP access for email, calendar, and corporate directories.

Some other related efforts include:

- Wireless Knowledge — www.wirelessknowledge.com: This company's Workstyle for Microsoft Exchange™ extends the capabilities of Exchange Server 5.5 and Exchange 2000 Server to mobile devices with wireless (or wired) connections. Workstyle for Lotus Domino™ extends the capabilities of Notes/Domino Server 4.6 and 5.0 to mobile devices with wireless (or wired) connections.
- Microsoft — www.microsoft.com: Microsoft Hailstorm (now renamed to My Services) intends to advance the Microsoft .NET strategy and will enable developers to build user-centric XML Web services that offer a new level of personalization for both consumers and business users. It helps to create XML-based Web services deliverable to a variety of PC and non-PC devices such as handhelds and Web appliances.
- Oracle 9i Application Server — www.oracle.com: The Wireless option of the enterprise edition of this server lets you extend Web applications and portals to wireless devices and standard phones using voice recognition. The Personalization Option lets you offer personalized recommendations to your Web site visitors.
- ActiveBuddy — www.activebuddy.com: ActiveBuddy allows instant messaging (IM) users to access the databases of a variety of corporate clients using IM commands. The company's technology ("bots") can be added to existing buddy lists, and lets users tap into partner databases for stock quotes, news, weather, sports and movie listings.
- *Berkeley's Iceberg Project*: The recent ICEBERG project [30] from UC-Berkeley shares the iMobile goals of any-to-any communication services and personal mobility services, but has so far concentrated mostly on voice, rather than data services.

While the iMobile Enterprise Service Platform is only in its infancy and may miss some enterprise functionalities (such as access to Lotus Notes) described in some of the above platforms, its flexible architecture has allowed us to rapidly add new services like location-aware services, personalized multimedia services and home network device controls. Furthermore, since the iMobile architecture is completely independent of any specific access devices, wireless networks, or operating systems, it can very easily accommodate any new emerging mobile devices and network solutions.

Finally, the IETF standardization effort on Open Pluggable Edge Services (OPES) [14] can be viewed as a potential future use for iMobile. The Internet is facilitating multiple forms of distributed applications, some of which employ application-level intermediaries. The OPES group is defining application-level protocols enabling such intermediaries to incorporate services that operate on messages transported by HTTP and RTP/RTSP. At the IP level, the participating intermediaries are endpoints that are addressed explicitly. The emergence of ideas like middleware service providers supported by a group like OPES shows an industry trend to create value-added services on the network edge. The security model for such services involves defining the administrator roles and privileges for the application client, application server, intermediary, and auxiliary server. The data integrity model defines what operations are permitted by the content owner(s), and what guarantees of content correctness can be made to the owner(s) and viewers when content-related services are performed. Our mobile service platform work can potentially be deployed on the network edge to provide value-added services described in the proposed OPES charter.

7. Summary and Future Work

The original iMobile service platform (standard edition) introduced three abstractions on top of a programmable proxy: devlets to interact with various access devices and protocols, infolets to access multiple information spaces, and applets to implement application and service logic. The proxy engine arbitrates the communications among devlets, applets, and infolets. It maintains user and device profiles, which are used to provide personalized services and to perform transcoding before returning contents to receiving devices.

The iMobile Enterprise Platform (enterprise edition) addresses the needs of enterprise applications by interacting with corporate authentication and authorization services. It provides a scalable, available, and flexible service platform by introducing the notion of reliable message queues accessed by multiple gateways and iMobile servers. The enhanced service profile database governs all iMobile administrative and service operations. The current and future work includes designing a specification language and enforcement mechanism for QoS policies including security, fault tolerance, timeliness, and prioritization. We are also working on increasing all the dependability aspects of iMobile EE, including exploring the use Java 2 EE technologies and platforms. Naturally, we are also developing new and improved devlets and infolets for new access devices and iMobile services.

Work is also underway to create the iMobile Micro Edition (ME) that would allow us to move a scaled-down iMobile server to increasingly powerful mobile devices. iMobile ME [38] will allow a device to export its environment resources and information as infolets accessible by all other devices and users around the world. It can also handle intermittent wireless connections seamlessly with message queues and network synchronization capabilities. By combining iMobile ME with iMobile Enterprise platform (EE), we hope to give mobile users the richest mobile service experience that AT&T can offer to its enterprise customers.

Going forward, the iMobile project will be commercialized under the name "AT&T Mobile Network," or AMN. For the latest project information on AMN, please visit

http://www.research.att.com/sw/tools/amn .

Acknowledgements

Herman Rao, Di-Fa Chang, and Ming-Feng Chen made significant contributions in the original iMobile standard edition. Chris Rath, Jim Rowloand, and Matt Green helped with the initial user provisioning website and added several new infolets. David Kormann and Avi Rubin helped us with the integration of Absent and iMobile on mobile devices. We collaborated with Muthu Muthukrishnan and Ted Johnson on the use of a CDPD location determination technology. Urs Muller and Beat Flepp helped us with an experimental integration of iMobile with the Microsoft Exchange 5.5 interface (which does not support WebDAV). Ashish Singh helped migrate the SMS gateway from SE to EE, Jun He helped develop a syslog-based logging mechanism for iMobile, and Kai Wei developed an early prototype of a resource monitor for iMobile components. George Otto directed and produced an iMobile video that showcases the iMobile technology. The video is downloadable from our website. We would also like to thank Fred Douglis and Kathy McKenna for their valuable comments, which helped us greatly in enhancing the quality of this paper.

References

[1] H. Rao, Y. Chen, D. Chang, and M. Chen, “iMobile: A Proxy-based Platform for Mobile Services”, The First ACM Workshop on Wireless Mobile Internet (WMI 2001), Rome, July 2001.

[2] Apache Software Foundation, Jakarta Tomcat, http://jakarta.apache.org/tomcat/.

[3] C. Gilmore, D. Kormann, and A. Rubin, “Secure Remote Access to an Internal Web Server”, Proc. ISOC Symposium on Network and Distributed System Security, February, 1999.

[4] Sun Microsystems, “Java Message Service API”, http://java.sun.com/products/jms/.

[5] IBM, “MQ Series”, http://www.ibm.com/software/ts/mqseries/.

[6] Fiorano Software, Inc., “FioranoMQ”, http://www.fiorano.com/.

[7] Informix, Inc., CloudScape, http://www.cloudscape.com/.

[8] Sun Microsystems, “JDBC API”, http://java.sun.com/products/jdbc/.

[9] Sun Microsystems, “Java Naming and Directory Interface”, http://java.sun.com/products/jndi/

[10] AT&T, PhoneWeb, http://phoneweb.research.att.com/.

[11] H. Rao, D. Chang, and Y. Lin, “iSMS: An Integration Platform for Short Message Service and IP Network”, IEEE Network, 15(2): 2001, 48–55.

[12] Y. Chen, H. Huang, R. Jana, S. John, S. Jora, A. Reibman, and B. Wei, “Personalized Multimedia Services Using a Mobile Service Platform”, in Proceedings of the IEEE Wireless Communications Networking Conference, Florida, March 17–21, 2002.

[13] S. John, R. Jana, V. Vaishampayan, and A. Reibman, “iVideo—A Video Proxy for the Mobile Internet”, Proceedings of IEEE 11th International Packet Video Workshop, Korea, May 2001.

[14] Xalan, XSLT stylesheet processor, http://xml.apache.org.

[15] XSLT, XSL Transformations, http://www.w3.org/TR/xslt.

[16] The Open Group, “Introduction to Single Sign On”, http://www.opengroup.org/security/sso/sso_intro.htm.

[17] The WAP Forum, Wireless Application Protocol, http://www.wapforum.org.

[18] K. Savetz, N. Randall, and Y. Lepage, “MBONE: Multicasting Tomorrow's Internet”, http://www.savetz.com/mbone/.

[19] Eliza, http://www-ai.ijs.si/eliza-cgi-bin/eliza_script.

[20] Remote Authentication Dial In User Service (RADIUS), http://www.ietf.org/rfc/rfc2138.txt.

[21] SonicMQ, http://www.sonicsoftware.com/.

[22] WEBDAV, http://www.ietf.org/rfc/rfc2518.txt.

[23] Newstakes, http://www.newstakes.com.

[24] ITU-T Recommendation H.263, “Video coding for low bit rate communication”.

[25] P. Chen, “The Entity-Relationship Model—Toward a Unified View of Data”, ACM Trans. on Database Systems, 1(1), 9–36, 1976.

[26] DAV Searching and Locating (DASL), http://www.webdav.org/dasl/protocol/draft-davis-dasl-protocol-00.html.

[27] Research In Motion, Ltd., “Blackberry Enterprise Edition for Microsoft Exchange”, http://www.blackberry.net/support/pdfs/HandheldTechnicalWP.pdf, 2001.

[28] M. Nilsson, J. Hjelm, and H. Ohto, “Composite Capabilities/Preference Profiles: Requirements and Architecture”, W3C working group, http://www.w3.org/Mobile/CCPP/, 2000.

[29] The Apache Group, The Apache Cocoon Project, http://xml.apache.org/cocoon/index.html, 2000.

[30] H. Wang, B. Raman, C. Chuah, R. Biswas, R. Gummadi, B. Hohlt, X. Hong, E. Kiciman, Z. Mao, J. Shih, L. Subramanian, B. Zhao, A. Joseph, and R. Katz, "ICEBERG: An Internet-core Network Architecture for Integrated Communications," IEEE Personal Communications, 7(4), August 2000.

[31] M. Hiltunen, R. Jana, and R. Chen, "Resource Allocation for an Enterprise Mobile Services Platform", Proceedings of the 17th International Symposium on Computer and Information Sciences (ISCIS XVII), Orlando, FL, October 2002.

[32] Oracle Corporation, "Oracle 9i Application Server Release 2 – An Overview of the J2EE and Web Services Features", http://technet.oracle.com/tech/java/oc4j/pdf/Oracle9iAS-R2-J2EE.pdf , 2002.

[33] Jabber Software Foundation, The Jabber Protocol, http://www.jabber.org .

[34] The SMS Forum, The Short Message Peer-to-Peer Protocol, http://smsforum.net .

[35] BEA Systems, Inc, BEA Web Logic Server, http://www.bea.com/products/weblogic/server .

[36] Certicom, Inc, Movian VPN Client, http://www.certicom.com .

[37] Nokia corporation, Nokia One – Mobile Connectivity Service, http://www.nokia.com/nokiaone.

[38] Y. Chen, H. Huang, B. Wei, M. Chen, and H. Rao, "iMobile ME – A Light-Weight Mobile Service Platform for Peer-to-Peer Mobile Computing (invited paper)", in Internet Technologies, Applications and Social Impact, IFIP TC6/WG6.4, Workshop on Internet Technologies, Applications and Social Impact (WITASI 2002), October 10-11, 2002, Wroclaw, Poland.

[39] Sun Microsystems, "Java Servlet Technology", http://java.sun.com/products/servlet/ .

[40] Alteon WebSystems, Inc. "Implementing High Availability Layer 4 Services Using VRRP and VRRP Extensions", White paper, http://www.alteonwebsystems.com, October 1999.

[41] Cisco Systems. "Enhancing Availability, Performance, and Security for BEA WebLogic Clusters Using Cisco CSS 11000 Series Content Services Switches", White paper.