12-31-2004

Mobile Information Systems in a 3G - WLAN World: Many Rich Clients Who Know Where They Are

David Parsons
Massey University

Mark Cranshaw
Southampton Institute, United Kingdom

Mobile Information Systems in a 3G – WLAN World: Many Rich Clients Who Know Where They Are

David Parsons[1]
Mark Cranshaw[2]

[1]Institute of Information and Mathematical Sciences
Massey University, Auckland, New Zealand
Email: d.p.parsons@massey.ac.nz

[2]School of Computing and Digital Communication
Southampton Institute, Southampton, United Kingdom
Email: mark.cranshaw@solent.ac.auk

Abstract

The worldwide introduction of third generation mobile phone networks, coupled with advanced handsets capable of hosting rich client applications, provides a new context for mass access information systems. This infrastructure provides the opportunity to move on from the client-server and intranet models of mobile computing and build more sophisticated mass-client architectures that distribute processes and data transparently between servers and mobile clients using personalisation, push technologies and location aware services. This paper discusses the technical platform for 3G and WLAN based mobile information systems and explores an example application that leverages some of the key features of this platform using Java APIs.

Keywords

Java, J2ME, mobile, 3G, WLAN, location, push, architecture

INTRODUCTION

There are many mobilised information systems in production, but they generally fall into one of two categories, the mobile intranet or the mass access mobile information system. Mobile intranet systems are those that can be categorised as mobilised versions of company intranets, whereby mobile devices are integrated into the internal business operation. In this context, we use the term mobile intranet to refer to systems that are accessible only internally within a company or organisation, rather than those that are confined to a geographical area, as described in (Grecas, Maniatis et al. 2003). In this type of mobile information system, the mobile devices are likely to be the larger and more powerful PDA type devices that can run software similar to that on the desktop. In such systems the number of mobile clients is relatively small and applications can be developed to target a particular type of device and a very specific working environment. A typical example of this kind of system is that used by the New Zealand Automobile Association (Vodafone 2004), whereby fewer than six hundred notebook devices act as mobile system clients and the software is specific to supporting roadside assistance. In contrast, there are what we might regard as mass access mobile information systems, where application services are made available to the public and the most common mobile device to access these services is currently the mobile phone. Most mobile phones are currently only capable of running applications that use a client server model, whereby a thin client with limited browser capability, typically using the Wireless Application Protocol (WAP), downloads data from an application server. In such cases the number of mobile clients can be very large and the system must be developed in a generic way such that it can work on a wide range of telephones and other mobile devices. An example of such a system currently available in the Australasian market is WordDial, which is a mobile portal (WordDial 2004). Its services are totally generic, in that any subscribing service provider can publish itself via the portal and any member of the public can access the site. As with all systems of this type, the client side presentation has to conform to the lowest common denominator, the basic WAP browser. Thus all processing must be done on the server side, which increases network traffic and reduces the opportunities for personalisation.

Future Mobilised Information Systems

Current developments in mass usage mobile devices will enable us to develop mobilised systems somewhere between these two application models. As the capabilities of the average mobile telephone increase, so the opportunities for building rich-client mass information systems will become greater. In a world where the client is not limited to WAP but can run applications using technologies such as Microsoft's .NET compact framework, or Java 2 Micro Edition, we can be more ambitious in the way that we build such systems. This world is rapidly arriving in the shape of 3G mobile phone networks, Wireless LANs and the more sophisticated handsets that use them. What, then, might potential future mass client information systems provide? There are a number of key themes, which together open up a number of possible application scenarios. The most obvious one is probably the ability to build rich interfaces that give a much better HCI experience than simple browsers can provide. However, perhaps the most powerful feature will be the ability to develop data centric applications that take advantage of the greater data storage and processing capabilities of mobile devices. Kobylarz (2004) describes *compound wireless services*, based on combinations of smaller component services that may be partially or fully implemented on a mobile device. Such services are device centric, utilising local application and data stores as well as wireless resources. Many mobilised systems will include an element of long lived publications (Burcea & Jacobsen, 2003) where data pushed from the server needs to be made locally available in a persistent store. The more data that can be stored locally, the less affected systems will be by weak connectivity and average data access time and data transfer requirements will be much less than in a thin client model. Location awareness is also a very important feature, enabling targeted services and content delivery filtered by location within a very small target area. Good examples of current location based services are those in the field of telematics for travel and transport, where the device size and cost of previous generation technology has been less of a barrier to adoption. Technology that allows vehicles to be tracked to enable dynamic map reading or tracking the progress of buses or other commercial vehicles is becoming commonplace, for example the Helsinki public transport system (Langstron 1998; Anon 2000). As the necessary equipment becomes smaller and cheaper, pervasive location-aware technology is going to enable all kinds of new applications on mobile phones and other mobile devices.

Features such as rich interfaces, mobile data and location based services can be part of a general approach to personalisation, whereby preferred data and services are filtered both on the server and client side to ensure that the information presented to the user is the most relevant and timely that can be found. (Chorafas 2001) states that personalisation of services is essential if the user is to commit to using mobilised systems. In fact a whole range of personalised M-commerce opportunities are possible, including wireless payment systems and meta malls (Gomes 2003). We are rapidly moving towards a world where enterprise information systems will by default include an element of mobility, using a range of technology including WLAN (wireless LAN) networks, Bluetooth and 3G / 4G mobile telephone networks. With the growing support infrastructure developing all the time, the potential applications for mobile information systems will be huge.

MOBILE PLATFORM GENERATIONS

What, then are the key features of the third generation (3G) mobile platform, in what ways does it offer more to mobile information systems than the second generation (2G) and how will it enable the types of rich client mass information system that we envisage? In essence, what we are seeing is a new wireless technology infrastructure that changes many of our preconceptions about the way that we use the telephone. Not least, this is because the devices themselves have seen a continuous increase in power and flexibility. This is not just due to Moore's Law (Moore 1965) and the resulting miniaturizing of processors but also to advances in a number of related technologies. The number of components on a chip may have changed out of all recognition since 1965, but so too have available random access memory and addressing, data storage technology, mobile communications infrastructure, battery technology and display technology. Together, all of these technological advances have converged to give us devices capable of mobile computing. The specification of a typical 3G handset is likely to comprise 32-64Mb of total memory and a processor running at about 400-500 MHz. Although memory is differently configured to larger computer systems, and batteries impose limitations on processors, it is not unreasonable to assert that such a telephone is already similar in power to a 1970s minicomputer or a mid 1980s PC. With such computing power available, there is clearly the potential to build client side applications that are much more sophisticated than simple WAP browsers.

Technological developments from second to fourth generation

Previous generations of mobile technology have simply enabled us to make standard telephone calls with a wireless device. However, the shift from 2G to 3G enables us to move far beyond devices that are mobile telephones to much more sophisticated mobile computing devices. Table 1 compares 2G and 3G characteristics, showing the key differences between the two platforms[*]

[*] It should be noted that the so-called 2.5G systems fall somewhere between these two specifications.

	2G	3G
Transmission	Airwaves	Packets
Bandwidth	30-200KHz	15-20 MHz
Speed	9.6 Kilobits per second (Kbps)	Up to 2 Megabits per second (at least 144 Kbps)
Connectivity	Dial up ‘talk time’	Always connected (mobile IP)
Hardware	Telephone handset	Mobile computing device

Table 1: 2G versus 3G characteristics

The change from dial up talk time to mobile Internet Protocol with much greater bandwidth changes the whole landscape of mobile software development. As devices get smaller and more powerful, and the available bandwidth gets bigger, more interaction is possible in mobile systems. Video and sound streaming may become commonplace. Interactive applications might involve client-server communications, peer-to-peer or a combination of both. System features such as Over-The-Air (OTA) provisioning can allow application downloads, updates and data synchronization, while still enabling concurrent voice calls. Looking into the next ten years, we can expect continuing development of mobile systems into the fourth generation (4G). The definition of 4G is somewhat vague, but is essentially 3G technology taken to its logical conclusion, with all ‘legacy’ aspects of the system removed and all services on a 4G network being purely digital. For example, all voice calls over a 4G network could use the Voice over Internet Protocol (VoIP). The key benefit of 4G architecture will be further scalability, since 4G effectively provides the kind of data access that is expected of a WiFi (802.11 WLAN) network but across the range of a mobile phone network. Although opinions differ as to how soon 4G systems can be widely implemented, some operators are considering leapfrogging 3G altogether and moving towards 4G (Economist 2003). Some comparisons between 3G and 4G are summarised in table 2.

3G	4G
Back compatible to 2G	Extend 3G capacity by one order of magnitude
Circuit and packet switched networks	Entirely packet switched networks
Combination of exiting and evolved equipment	All network elements are digital
Data rate up to 2 Mbps	Data rate up to 100 Mbps

Table 2: Comparison of 3G and 4G

Investment and return

Perhaps the most overarching motivation driving the mobile software industry forward is the investment that has already been made in 3G by the major mobile service providers, and the return that must come from this if they are to survive in the long term. In the U.K. for example the cost per head of adult population of the 20 year 3G licenses bought by the five successful bidders was US$734.61. In contrast, prices paid in Australasia seem relatively sober, with the six successful Australian bidders paying a total of $23 per head of adult population for 15 year licenses. New Zealand had four successful bidders, paying US$17 per head of adult population. The first of these, Telecom, is currently rolling out its 3G services based on Lucent’s EV-DO (Evolution Data Optimised) technology (cellular-news 2004).

Investment has been high but financial returns have been slow to materialise. Mobile data rates for current 3G services have been disappointing with speeds between 144Kbps and 384Kbps, significantly less than required for the effective provision of video streaming and video-conferencing services. This situation is, however, changing as wideband code division multiple access (W-CDMA) networks are steadily introduced (3G PressReleases 2004). This will require the conversion of existing 2G infrastructures and thus add to financial pressures. However, it is the only means of delivering the bandwidth needed for 3G services such as videoconferencing that is available to the providers (Ohr 2003).

There remain other business drivers that have led a number of companies to anticipate that they may be able to generate sufficient revenue to cover their investment and generate a return. There are ever more potential customers connected to the wireless internet in global markets that go way beyond what was possible with fixed line only systems. Many developing countries are installing wireless networks where no traditional cabling existed before. (Shaw 2002) states that the global number of mobile subscribers overtook the number of fixed line subscribers in 2002. Of course this statistic needs to be taken in context, because many mobile subscribers are also fixed line subscribers. However, it is indicative of a longer trend that will see widespread adoption of mobile technology where no fixed lines have previously been available. For example, one forecast estimates that there will soon be 100 million mobile users in Africa, or three times the number of fixed-line subscribers, and 200 million by 2010 (Foley 2002). It is therefore clear that mobile service providers will have access to new markets where there will never be competition from a fixed line infrastructure. Applications running in this type of global environment can benefit from economies of scale. Corporate vertical market applications can be deployed across the world, and the 3G infrastructure can provide new seamless service concepts to markets previously untapped. However, none of this is of any value unless subscribers are prepared to pay for the applications that are available; content is the defining essential.

The path from 3G to 4G technologies is unlikely to prove to be as direct as the mobile phone companies might wish. WLAN networks already offer an infrastructural alternative to 3G services in metropolitan areas, with bandwidth to compare with wired broadband networks. Some analysts believe that as business data requirements will be centred in the big cities, WLAN technology poses a significant threat to the viability of 3G networks in the business data application field (Woolcock et al 2001). Although some question-marks exist over the business case for the proliferation of WLAN access points, a number of wireless service providers currently offer commercial WiFi connection. T-Systems recently announced an initiative to link together 10,000 locations for wireless computer access bringing together 120 existing services and offering connection to WLAN hotspots in coffee bars airport and hotels across the world (Reuters 2004). WLAN technology also threatens the phone companies domination of the voice call market by supporting telephone calls over the internet using voice over the internet protocol (VoIP). Such services are already available in the US (Blake 2004).

The response of the mobile phone companies has been to work for the co-existence of 3G and WiFi as complementary technologies. Seamless roaming between 3G and WiFi networks has been possible for some time (Martin-Leon 2002) and Nokia is already working on the development of hybrid GSM/GPRS/WiFi phones from which the user will be able to make VoIP calls through WiFi hotspots (Blake 2004).

MOBILE SOFTWARE DEVELOPMENT PLATFORMS

For the developer planning to build applications on 3G telephones and other mobile devices, there are a number of available software platforms. Of course each type of device will expose its original equipment manufacturer (OEM) APIs which can be used for development, but since these are proprietary on different devices the same application may have to be recoded many times. In order to work at a higher level and develop cross platform software, it will usually be more effective to use either the .NET Compact Framework (for deployment on Windows Mobile for SmartPhone or Pocket PC Phone Edition operating systems) or the Java 2 Micro Edition (J2ME). .NET provides a rich set of tools and APIs, including the ASP.NET Mobile Controls for mobile web access. The Java offering is less integrated, comprising standard API specifications from the Java Community Process such as the Java Technology for the Wireless Industry specification (JCP 2003), with tool integration generally offered by third parties. However, the overall feature sets available for development from both the .NET and J2ME platforms are broadly similar.

Mobile phone operating systems and handsets

There are a multitude of 3G handsets available running a number of mobile phone operating systems, but we might usefully make a distinction between those phones that support Microsoft Windows and those that use other systems. In the Microsoft camp, manufacturers of Microsoft Smartphone handsets include Mitac, HTC and Samsung. It may be that some of the other major manufacturers will begin to support this operating system over time, but currently this is a relatively small set. In contrast, the Symbian operating system is supported by a much larger set of manufacturers, including Samsung, Nokia, Siemens, Motorola, Matsushita, Psion, Sony Ericsson, Fujitsu, Sendo and BenQ. There is also a move towards some dedicated Linux phones, such as the E28, Invair and Motorola phones. In addition, many other handsets are capable of running the Linux operating system. The strength of non-windows operating systems will no doubt influence the software development platform used by developers of mobile applications when choosing between Java or .NET, since the huge range of phones that run Symbian or Linux will support Java but only a few will be Smartphones that can host applications developed with the .NET compact framework. In this paper we focus on software development for Java phones, though much of the architectural discussion would apply to .NET development as much as Java.

JAVA PHONE APIs

Java phones use the APIs of the Java 2 Micro Edition (J2ME) platform, specifically Java Technology for the Wireless Industry (JTWI). JTWI is a standard API specification (from Java specification request number 185) that is the result of the Java Community Process (JCP) and is published on behalf of the Java community by Sun Microsystems. This specification provides a set of APIs for mobile phone application development, the most important of which is the Mobile Information Device Protocol (MIDP) version 2.0, one of two mandatory components. The other mandatory component is the Wireless Messaging API (WMA) version 1.1. In addition, there is a conditional (optional) component, the Mobile Media API (MMAPI) 1.1. Underlying these APIs is the minimum configuration specification, which is the Connected Limited Device Configuration (CLDC) version 1.0. Figure 1 shows the JTWI phone architecture, which comprises the various Java APIs running over a lower level platform of custom and native APIs. As well as the standard APIs of JTWI, mobile applications may also utilize other APIs for J2ME, including the standard specifications for web services, security and trust services, location, session initiated protocol, mobile 3D graphics and event tracking.

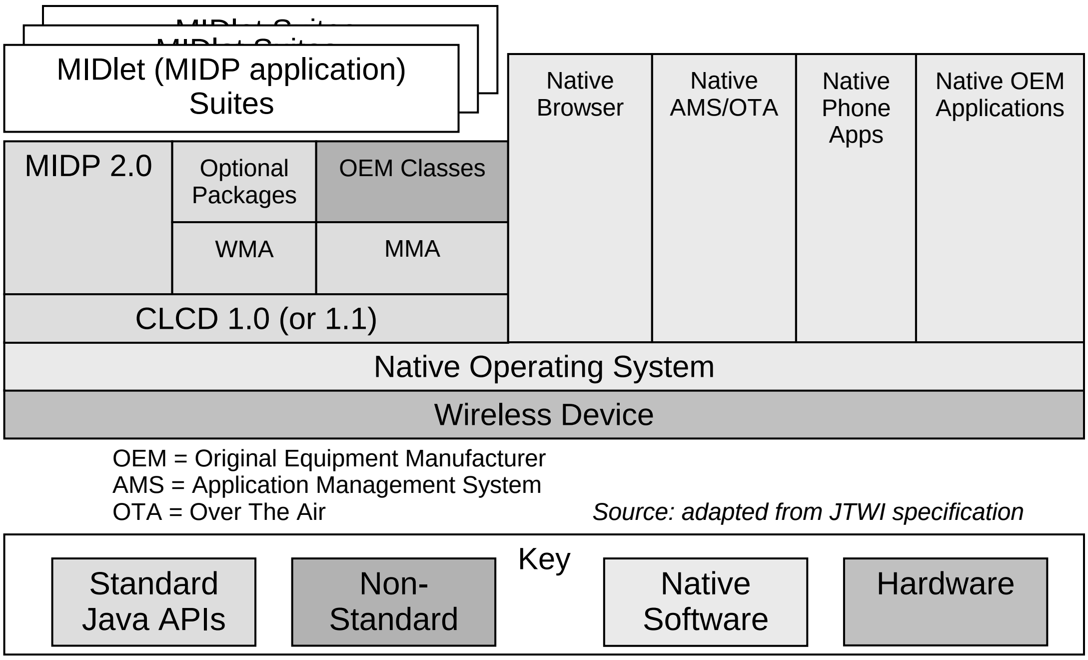

Figure 1: JTWI phone architecture

MIDP libraries

A J2ME profile requires the support of a specific J2ME configuration, and is used to provide functionality targeted to a particular family of devices, vertical market or industry. The Mobile Information Device Profile (MIDP) is one of two profiles (the other being the Information Module Profile) that work on top of the CLDC. It provides for interactive applications with graphical interfaces, and is the standard Java profile for mobile telephone development under the JTWI specification.

The minimum specification MIDP interface is something of a bounded universe. The minimum requirement is simply for a monochrome display of 96 by 54 pixels and either a standard QWERTY keypad or a standard ITU-T (International Telecommunication Union standard) Keypad, comprising only the digits 0-9 (and their associated alphabetic characters), * and #. This very limited input and output suggests that any successful software development for MIDP devices would need to provide value in features other than a rich interactive experience. Such value might come from areas such as push technology, location based services and data management.

Push technology

Push technology means the ability to push information to a device without that information being specifically requested, enabling live, transparent updates to applications and data. A fairly basic example of this is already provided with WAP push technology. MIDP push enables us to send a message to a running application, trigger an application update, provide user alerts, send data, notify listeners or start an application using the Application Management System (AMS). The original MIDP specification (version 1.0) only provided support for HTTP based connectivity, which is a client-server request-response mechanism. Using HTTP, it is not possible to push data since the initial message must always be a request to the server, i.e. it must be client initiated. To provide for a richer set of connectivity options, MIDP version 2.0 included support for push connections, which may be implemented using Transmission Control Protocol (TCP) sockets or User Datagram Protocol (UDP) datagrams. Such technologies allow servers to push data to clients, and also enable peer to peer two-way Java based communications. Due to some optional features of the specification, it is difficult to guarantee which particular connection types might be supported by a given mobile phone. An alternative way of supporting push is to use the Java Wireless Messaging API (WMA) component of JTWI, which includes a Short Message Service (SMS) and Cell Broadcast SMS (CBS) push option. Since SMS is now a standard feature of mobile phones, this type of connection is virtually guaranteed to be available on a 3G phone. The various architectural components are shown in Figure 2.

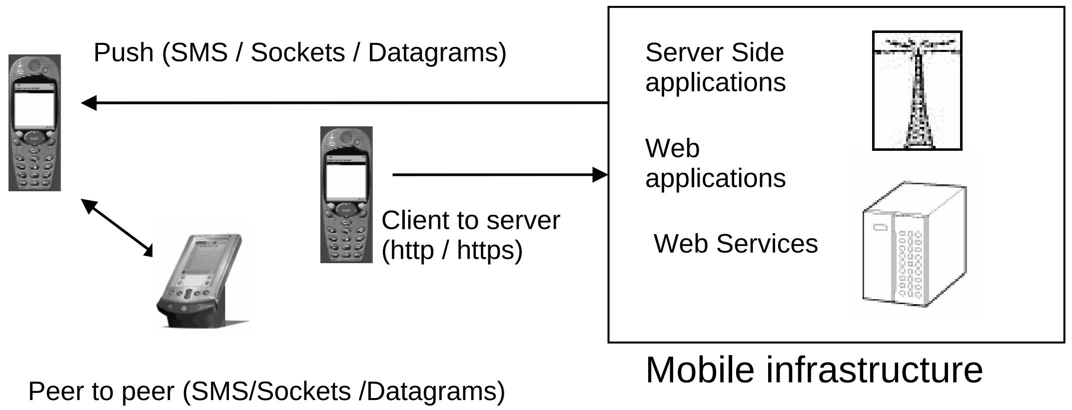

Figure 2: Components of a mobile application architecture

Regardless of the underlying communication mechanism being used, the MIDlet handles pushed connections using the push registry, comprising a simple API and two types of push events; inbound connections and timer alarms. Timer alarms enable an application to be started or updated on a timed basis. This does not require any external connection, so the source of the push event comes from the device itself. In contrast, inbound connections enable an application to be started or updated from an external source, which may be a peer (i.e. another mobile device) or a server side application. Since the wireless messaging API is also available as part of the Java 2 Standard Edition (J2SE), desktop or server based systems can be developed to provide the control framework for push-enabled application services.

The Java Location API

Perhaps the most powerful feature of a mobile information system is the ability to discover the location of a mobile device. The Java Location API specification (Java Specification Request 179) is intended to provide a generic interface for position of mobile devices that will apply to most positioning methods, such as GPS and E-OTD (Haiges 2003). In order to remain generic, the core API does not include features that are specific to one technology only. However, there may be extensions for specific features. The location API is intended as an optional package that will work with various J2ME profiles. The minimum platform is assumed to be CLDC 1.1, which is currently beyond the minimum JTWI specification. Since the target platform is low memory devices, the recommend maximum footprint for implementations of the API is 20K of ROM and 2K of RAM.

Mobile positioning technology covers a range of implementations, broadly sub-divisible into systems that use the network to position the device within it and those that use the features of the device itself. Network based options include approaches such as Cell Of Origin (COO), a very basic mechanism that simply identifies which cell a device is located in, and more complex approaches such as TDOA (Time Difference of Arrival) where the time difference between signals from the device to the network are measured. Device based features include the widely used Global Positioning System and Enhanced Observed Time Difference (EOTD). There are also hybrid systems such as assisted GPS (Haiges 2003).

Location objects in the Java API include not just the standard location information such as latitude, longitude and timestamp but also encompass attributes such as altitude, course and speed. With this kind of information, it is possible to model in three dimensions the relative positions of mobile devices, and calculate likely points of interaction (Ishbell 2003). Constantly updated central data can be used to track other mobile devices and historical location data can be stored in a database, making some prediction possible. The accuracy of the location data available will depend on its source, since it may be network based, or use GPS. Where a GPS facility is available, positioning can be accurate to between 10 and 30 metres. GPS provides a richer data set than can be gathered from the mobile telephone network, but a GPS receiver needs to get information from 4 of the 24 global satellites to fully determine its position and of course will not work indoors, so cannot always be relied upon. Therefore a combination of the two is likely to be used. Another optional J2ME package of the Connected Limited Device Configuration is the Bluetooth API, which analyses the discovery of local devices. This can also be used as part of a comprehensive location enabled system, providing that static Bluetooth terminals are integrated into the system. The location APIs, via the Landmark class, also support knowledge about addresses, including building, floor and room information. Applications can manipulate these landmarks in categories, and store them in a local persistent store (the LandmarkStore). A landmark can be any type of location such as a home, office, restaurant etc., enabling applications to target specific types of location based services. These services can be integrated within broader applications to support business processes such as identifying the nearest source of part supply for mobile technicians (Skoglund et al 2003.) One of the key drivers for systems to support location awareness is government legislation relating to emergency calls from mobile telephones. In the US, the federal Communications Commission (FCC) has introduced the Enhanced 911 (E911) laws, which require all mobile telephones to have location reporting facilities. The first phase of this process requires wireless carriers to be able to respond to requests by local Public Safety Answering Points (PSAP) for the telephone number of a wireless 911 caller and the location of the antenna that received the call. The second phase requires the carrier to provide far more precise location information, within 50 to 100 meters in most cases. The deployment of phase 2 should be completed by December 31, 2005. However, it should be noted that many carriers were unable to meet the original deadline for phase 1, in October 2001 (Ishbell 2003).

EXAMPLE APPLICATION TEST ENVIRONMENT

In order to develop and test mobile Java applications that combine features such as push technology and location awareness, an environment has been developed that works alongside the Sun J2ME Wireless Toolkit. The Sun toolkit provides support for the MIDP APIs, including push technology, but does not provide any support for working with the Java Location APIs. Whilst there are some location simulators freely available, such as the Sony Ericsson simulator, none of these currently support the standard Java APIs. Therefore an orthogonal Java module has been developed that provides mobile device simulation data that is exposed through an implementation of the location APIs. Data is fed into the simulator via a GUI that enables a virtual mobile device to be guided through a map of the upper campus at Massey University's Albany campus (Figure 3).

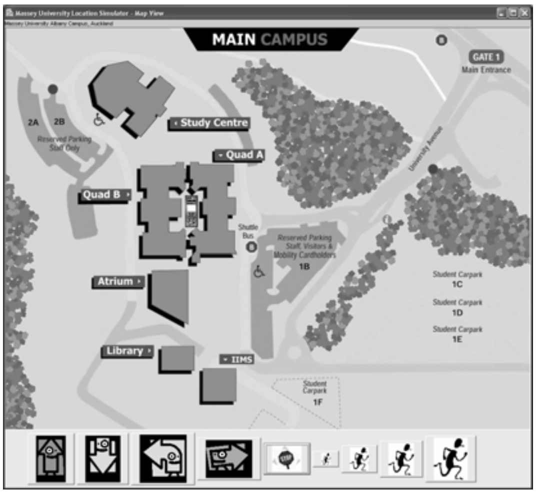

Figure 3: The Location API GUI

The virtual device is initially located at the centre of the quadrangle, at location 36 43 57 south, 174 42 04 east. In order to provide appropriate simulation data, 16 key points were mapped across the campus using a Global Positioning System (GPS) device. Our location experiments are based on knowledge of which building a mobile device is inside, or closest to. The connection between the running Wireless Toolkit and the Location API simulator is via Remote Method Invocation (RMI), with an implementation of the LocationProvider from the Java Location APIs acting as the remote client.

The Mobile Student application has been developed to demonstrate some key features of a mobile application that uses persistent data, push technology and location awareness. An example drawn from this system is shown here to demonstrate a mobile database of course timetable information: the days, times, and rooms of the lectures and labs and who is presenting them. Since this is the kind of information that can be subject to change, for example if a lecturer is unavailable or there are rooming problems, it is useful to enable dynamic updates to this data. The mobile persistent store maintains student-specific timetable data using the MIDP Record Management System (RMS). This enables the timetable to be accessed even when the device is disconnected. However, this data is not stored only on the mobile device but also on a central server, where the master timetable can be updated. To ensure data synchronization with all mobile devices, the push registry is used to update local data using a publish-subscribe type mechanism. This is implemented using a combination of CBS and HTTP connections. When there is a central change to the timetable, a cell broadcast message is sent to the push registry connections of all the registered mobile devices. On receiving this message, an agent on the mobile device filters the notification and, if the changes are relevant, triggers an HTTP download of the new data from a web server. This is transparent to the user of the mobile device, apart from some notification alerts. There is no requirement for the user to specifically check for updates to the timetable since these are all handled from the server side. The push sequence is shown in Figure 4.

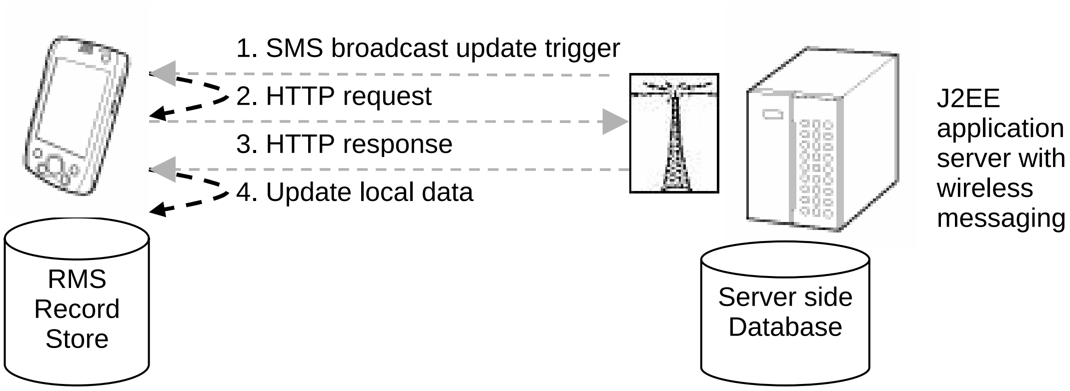

Figure 4: The push sequence

A screen dump (from the Sun J2ME Wireless Toolkit) showing the system in action appears in Figure 5. Note that the timetable information for the lecturer and room of Thursday morning's lecture is being updated dynamically by a server-initiated push message, followed by a download. The user sees the required alerts, informing them of the change to the data and asking for acceptance of network charges, but the actual data transfer is automatic if the airtime use is accepted.

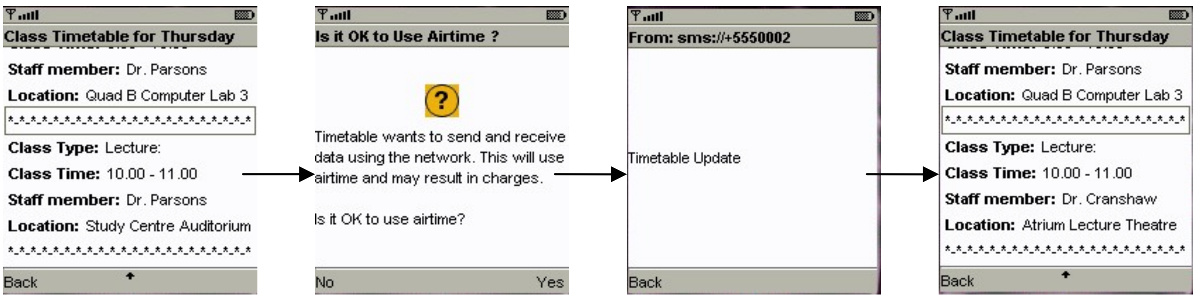

Figure 5: The Mobile Timetable application in action

The course timetable feature previously described does not require location awareness. However there are a number of scenarios where it would be useful to push data to the mobile client that was relevant to the current location. One such example might be to inform a passing user of new books arrived in the library. The MIDlet can be provided with location specific data by implementing the ProximityListener interface (from the Java Location APIs), and is then informed by the implementation of the LocationProvider (also from the Java Location APIs) when the device is in range of the library, once it has registered itself as a ProximityListener. This range, which is the radius around a single set of coordinates, is defined in metres by the application when the ProximityListener is registered. The same MIDlet can register itself as a ProximityListener for multiple locations, so that, for example it might register itself to listen for the six main building locations on campus.

When a proximity event is triggered, the MIDlet is provided with the appropriate Coordinates and Location objects, from which it can invoke the appropriate response. In the case of proximity to the library, it can make an HTTP request to the new books listing service. Alternatively, if the location was one of the buildings housing computer labs, the system might download information about the currently available lab machines.

Questions and futures

There are a number of questions about mobile software architectures that are raised by the Mobile Student application. Perhaps the most important one that we need to address is the location of data in a system. Where will we store the data in a distributed system that consists of many nodes of very different scales? How much data should be stored or replicated locally on a mobile device, and how much should reside only on the server? What are the best mechanisms for maintaining replicated data on mostly-connected devices, and how do we ensure both high availability of data and data integrity? If we integrate multiple nodes into a distributed, data centric application, we will have to handle many different types of message. How big, then, is a message to be? Should it contain data, or should it merely act as a trigger for data to be downloaded in a more traditional client server architecture? If we have a number of choices for the types of connection that we may use (HTTP, HTTPS, sockets, datagrams, SMS, CBS), which combination of connections would suit a given application? These are the kinds of questions that we will have to address in some detail if we are to build mobile applications that are both robust and performant. Considerable research has been undertaken in recent years into many of these questions, but generally in the context of proprietary systems. Now that the technology is in place within mass access mobile devices to support standard, interoperable frameworks, it would be useful to mine this previous research for the most effective architectural and design patterns and apply them to commercial systems.

Looking ahead, it is clear that building useful applications is crucial to the success of 3G. For such applications to be successful, usability through the transparent integration of servers and mobile devices is essential. To provide applications that are feature rich and usable within the confines of the current hardware, operating systems and APIs, the required architectures will be complex. It is unlikely that there will be one standard pattern for mobilized applications. Rather, each system will need a customised architecture that is based on the scale, operating context and non-functional requirements of its business context. Choosing the correct architecture for a given system and providing a successful feature set will mean a major challenge for developers, but one that could bring major rewards in bringing new mobile information systems to a fast growing market. One helpful contribution of the research community would be to identify the key patterns in the literature and test their applicability to the particular systems and platforms now available for mass access information systems.

REFERENCES

3G PressReleases (2004) Advances in 3G Services Demonstrated, URL:http://www.3g.co.uk/PR/June2004/7895.htm, Accessed 7th July 2004.

Anon (2000) Finnish National Architecture for Transport Telematics complete, *ITE Journal - Institute of Transportation Engineers* 70, 4.

AskOxford (2004) Email SMS and Online Chat, URL:http://www.askoxford.com/betterwriting/emoticons/?view=uk, Accessed 14 July 2004.

Blake (2004) VoIP mobile phone service rings in. *AustralianIT News*, July 6 2004, URL: http://australianit.news.com.au/articles/0,7204,10028142%5E15841%5E%5Enbv%5E,00.html, Accessed 15 July 2004

Burcea, I. and Jacobsen, H. (2003) L-ToPSS – Push-Oriented Location-Based Services, *Technologies for E-services*, Lecture Notes in Computer Science, 2819, 131-142.

cellular-news (2004) New Zealand getting 3G this year, *cellular-news*, URL:http://www.cellular-news.com/3G/new_zealand.shtml, Accessed 14 July 2004.

Chorafas, D. (2001) *Enterprise Architecture For New Generation Information Systems*, Boca Raton, FL, St. Lucie Press.

Economist (2003) Move over 3G: here comes 4G, *The Economist*, San Francisco, URL:http://www.economist.com/business/displayStory.cfm?story_id=1816742, Accessed 14 July 2004.

Foley, K. (2002). Africa goes mobile, *NUA Internet Surveys*, URL http://www.nua.ie/surveys/analysis/weekly_editorial/archives/issue1no216.html, Accessed 14 July 2004.

Gomes, P. (2003) WYSINWYG - What You See is Not What You Get, *Receiver*, 8, URL http://www.receiver.vodafone.com/archive/index.html, Accessed 14 July 2004.

Grecas, C., S. Maniatis, et al. (2003) GIP: An Infrastructure for Mobile Intranets Deployment, *Wireless Networks* 9, 321-330.

Haiges, S. (2003) The location API: simplify access to mobile positioning methods, *Java Developer's Journal*, 8, 10, 52-5.

Ishbell, S. (2003) Targeting GPS-Integrating J2ME, GPS, and the Wireless Web, *Java Developers Journal*, 7, 3.

JCP (2003) JSR-000185 Java Technology for the Wireless Industry 1.0 (Final Release), URL http://jcp.org/aboutJava/communityprocess/final/jsr185/index.html, Accessed 14 July 2004.

Kobylarz, T. (2004) Beyond 3G: Compound Wireless Services, *IEEE Computer*, 37, 9, 23-8.

Langstrom, L. (1998) Public Transport Telematics System Configuration in the City of Helsinki, City of Helsinki Traffic Planning Division, URL:http://www.hel.fi/ksv/entire/repDetSystem.htm, Accessed 14 July 2004.

Martin-Leon, S. (2002) *Leveraging Mobile High-Speed Data Access: Seamless Roaming between WiFi and 3G Networks*, Lucent Technologies White Paper

Moore, G. (1965) Cramming more components onto integrated circuits, *Electronics*, 38, 8.

Ohr (2003) 3G rollout plagued by uncertainty URL:http://www.commsdesign.com/showArticle.jhtml?articleID=16500565, Acessed 7 July 2004

Reuters (2004) T-Systems aims to make WiFi as easy as a phone call. July 2004. URL:http://www.reuters.com/newsArticle.jhtml?type=topNews&storyID=5595914, Accessed 6 July 2004

Shaw, R. (2003) Mobile Overtakes Fixed: What Happens Next? URL:http://www.itu.int/osg/spu/presentations/2003/prague-presentation-mobile-overtakes-fixed.pdf, Accessed 14 July 2004.

Skoglund, E., Hartlen, B., Erickson M. and McIntyre, A. (2003) Practical Integration of Java-Oriented Technologies and 2.5/3G Platforms, *Java Developers Journal*, 8, 8.

Vodafone (2004) Featured Case Study: NZAA. MobiliseYour Business, URL:http://www.vodafone.co.nz/business/10.6_mobilise_cs_nzaa.pdf, accessed 14 July 2004.

Woolcock, K. Murphy, S. Wyatt, P. Tyndall, M (2001) *The barbarians at the gate- Wireless LAN storms 3G citadel*, Nomura Equity Research, March 2001

WordDial (2004) Your Short Cut To More On The Mobile Web, URL:http://www.worddial.com/, accessed 14 July 2004.

COPYRIGHT