Res. Lett. Inf. Math. Sci., 2004, Vol. **6**, pp 47-58
Available online at http://iims.massey.ac.nz/research/letters/

The mobile future

D. PARSONS

Institute of Information & Mathematical Sciences
Massey University at Albany, Auckland, New Zealand

d.p.parsons@massey.ac.nz

The computing power of a third generation mobile telephone is already similar to that of the first personal computers from the mid 1980s or a third generation minicomputer from the early 1970s. In addition, wireless connection speeds can be substantially faster than the fixed-line modem in most home computers. The fact that the mobile telephone is the standard tool, leisure device or fashion accessory (depending on your perspective) carried by millions worldwide means that the potential for these devices as a pervasive computing platform is huge. In the mobile future, service providers will be selling pervasive applications rather than handsets or bandwidth, and there will be many opportunities to develop new and innovative mobilised systems. This paper looks what the mobile future might hold, and explores some of the standards and tools that can be used for mobile application development. It asks some questions about what sort of applications might be developed and what kind of distributed architectures might be required to make them work effectively and efficiently. It concludes with a discussion of an application developed using the Java Technology for the Wireless Industry APIs that are being adopted by the major mobile providers.

1 Introduction

In this paper we begin by asking whether the future is predictable, as a precursor to looking at some current predictions for mobile technologies, some of which are at the moment highly speculative. We then examine the current state of the mobile technology industry, in particular the features of third generation (3G) mobile telephones and their supporting infrastructure. The following section looks at issues around software development for 3G mobile devices, including available APIs and platforms. In particular, we focus on the Java Technology for the Wireless Industry (JTWI) specification, which will be an important software platform for the coming generation of mobile telephones. Finally, we use an example application to highlight some important features of JTWI and look at a sample mobile architecture.

Can we predict the future?

‘*Those who have knowledge, don't predict. Those who do predict, don't have knowledge*’ - Lao Tzu

If there is one thing that we can discern in the history of prediction, it is that it almost always results in the predictor looking foolish, typically due to either excessive optimism or excessive pessimism. It is almost too easy to find erroneous predictions in the fields of science and technology, such as the famous comment by Lord Kelvin in 1897 that ‘*radio has no future*’ [1]. Quotes like this teach us that it is perhaps unwise to discount the usefulness of a technology simply because its current applications seem limited. For example, the Vice-President of Engineering at US West said in 1999, on high speed wireless data connections, that ‘*You could use it in your car going 60 miles an hour, but I don't think too many people are going to be doing that.*’ [2]. Already, such attitudes seem very short sighted, as transport telematics applications become ever more sophisticated. In-car applications such as wireless video on demand, for example, will surely come as soon as the infrastructure is installed to support them.

This paper is based on an invited presentation originally given at the ‘Software Development Conference 2004’, Software Education Associates, Wellington and Sydney, March 8th-12th, 2004, http://www.softed.co.nz/sdc2004/.

Can we predict the success of mobile technologies?

The recent history of the mobile industry suggests that it is difficult to predict which particular technologies will be successful and which will disappoint. For example, Short Message Service (SMS) texting has been a much greater success (at least outside of the United States) than the industry expected, with tens of billions of messages being sent worldwide every month. There are probably a number of reasons for this success, and low cost is probably an important factor, but it may be that a sense of ownership is also a key characteristic. '*It is a method of communication that can be endlessly exploited and developed by individuals or small groups to create a kind of personal language.*' [3] In contrast to SMS, the Wireless Application Protocol (WAP) which was expected to be a highly popular way of mobilising the World Wide Web has been an almost universal failure*. Again, price may be a factor, but it seems that usability is probably the key problem. '*WAP seems to be so mysterious and impenetrable that users don't learn much even after substantial use*' [4]

Perhaps, before making predictions, we might consider some more successful attempts to predict the future and learn from them. In the science fiction genre, there are many books that attempt to predict but only a few that transcend the detail and become meaningful visions. One example is George Orwell's *1984* which, although it cannot be said to have accurately predicted the actual year 1984 in detail, nevertheless gave the first picture of a society managed by electronic surveillance. The interesting thing about this book is that it was not attempting to predict the year 1984 at all. The title was derived by simply swapping two digits from its year of authorship (1948) and in a sense Orwell was writing about his own time, not the future. The greatest science-fiction is not about the future at all, rather, it is about today seen in a different light. In other words, maybe you *can* predict the future by looking at today. In the technology area, an excellent example of this is Gordon Moore's 1965 article that predicted that the number of components that could be provided on a single integrated circuit for minimum cost would increase from 50 (in 1965) to 65,000 in 1975, and that this trend in processor power could well continue into the future. His general observation was that '*The complexity for minimum component costs has increased at a rate of roughly a factor of two per year. Certainly over the short term this rate can be expected to continue, if not to increase*'. [5] This prediction came to be known as 'Moore's Law'. Although Moore's estimate for 1975 was slightly optimistic, the really remarkable thing is the length of time that his observation has held true, so that by the turn of the century the Pentium 4 chip had 42 million components.

From our examples we might consider two possible indicators of the future. First, that once a technology is in place new uses will be found for it, even if the original developers of that technology are unable to predict them. Second, that the cost and value of ownership are key components of the equation that leads to success or failure. The technology may often come before a need has been identified, but the business or personal need is the key driver for adoption and success. Whatever happens in the arena of mobile technology, one thing is for sure, that some aspects of the future will be utterly bizarre and unpredictable, as Coupland's Reverse Time Capsule demonstrates. [6]

2 Where we might be going

We are moving towards a world where connectivity between mobile devices and networks will be both automatic and persistent, and there will be no bounds imposed on how much information we can exchange. With such an infrastructure in place, the possibilities for mobile pervasive computing will be huge. The mobile devices that will be commonly available in the next few years will be highly sophisticated, with features that are currently only available for the select few, such as location based services, and possibly some of those that are still currently in the realm of fantasy, such as linking human thoughts directly to the Internet.

* WAP has, however, provided some useful services and continues to mature, so has mainly been a failure against expectations rather than an absolute failure to meet requirements.

Location, location, location

Perhaps the most significant aspect of mobile software development over the next few years will be location based services, which will become much more readily available, enabling delivered content to be 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. 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. Pervasive location-aware technology is going to enable all kinds of new applications on mobile phones and other mobile devices. Looking further ahead, increasing miniaturisation will allow tiny independent devices, 'smart dust', to provide continuous real-time monitoring of a moving environment such as river or pollutant flows [7]. Scuka [8] relates a real world example of location based services in action from Japan, where an elderly man who went missing from a shopping centre was successfully tracked by the PHS wireless LAN location enabled service, which discovered him to have boarded a bullet train. Because he was carrying a location enabled device, it was possible for the police to retrieve him safely at the next station.

M-commerce, M-future

As devices become more sophisticated, we will move towards having everything that we carry in a single machine. No longer will we need to carry a separate phone, wallet, PDA etc., all that we need can be encapsulated into a single device. Wireless payment systems will allow transactions to take place between the phone and point of sale systems. Coupled with the e-wallet comes all the activities designed to remove all your money from it. There is already substantial development of 'Meta malls' through combinations of mobile technologies, including perhaps local Bluetooth connections as well as 3G networks, using location awareness to provide information to shoppers. *'Imagine a future Oxford Street in London, or a Times Square in New York, with consumers in the midst of a corporate and brand filled datacloud of information and products – infotainment guiding us to shops and tempting us with offers or products and tokens being sent directly into our mobile devices.'* [9]

New applications will be able to utilize context sensitivity, with integrated systems measuring temperature, time, nearby resources, other local devices and the carrier's biometrics. Real world games are one example of how this multitude of information can be used to provide a partly virtual, partly real experience. To further integrate mobile devices with their users, mobile devices will become increasingly more wearable, moving on from the relatively simple device pockets and fully washable wiring to smart materials / intelligent fibres, building the 'FAN' (Fabric Area Network). Integrated wearable systems could combine a range of information, for example *'Sensors in a tie could emit a warning signal if the colours did not match your suit'.* [10] Looking even further ahead, we may be moving towards the Organic Internet. Research into nanotechnology and molecular biology may lead to human neuron cells being stored on silicon chips, opening up all kinds of futuristic possibilities such as artificial telepathy and parallel cerebral networks. Who knows, perhaps mobile devices will rival the fictional TV character Joe 90: *"For nine-year-old Joe can do anything, thanks to a fabulous electronic device which can transfer the brain patterns of those who are the greatest experts in their field. When he receives these brain patterns, Joe, with the aid of a special pair of glasses which have built-in electrodes, becomes a man in thought and deed..."* [11]

3 What is happening now?

'*Today is the seed time, now are the hours of work, and tomorrow comes the harvest and the playtime*'
- W.E.B. Du Bois

Although telephone technology has changed incredibly from the original hand-charged phones with manual exchanges to current digital mobiles, the way that we use these devices has hardly changed at all. "*In the world of telephony, almost nothing has changed in more than 50 years. We've been making phone calls in exactly the same manner - pick up the phone, dial a number, say hello, talk, listen, say goodbye,* *and hang up."* [12] What is different, then, about the new generation of 3G telephones and why do they represent a paradigm shift in mobile communications? 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 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. In 1989 it was predicted that by 2000 "*A hand-held calculator sized computer will house a reasonably sized local library that can be searched for any material in less than a minute.*" [13] In practice, of course, the hand held computer of 2000 already offered much more, including wireless connectivity, and progress has continued apace. With remarkable prescience, Moore's original paper included a cartoon showing a computer salesman of the future selling a 'handy home computer' that he held in the palm of his hand. Already we have this kind of technology in handheld computers such as the PocketPC and the Palm. Soon, we will have this power in the humble mobile telephone. If we look at the specification of a typical 3G handset, it will have maybe 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. There are many types of mobile device evolving now that use various feature of mobile technology. These include the Aplix Java enabled wrist watch, Windows Mobile[tm] Smartphone and the Dallas iButton. Such devices already demonstrate the range of possibilities for pervasive computing. As technology continues to develop, we can expect to see both more sophisticated mobile phones and a range of other mobile devices that together will provide a rich wireless environment.

Technology drivers

The technology of the mobile telephone has taken a great leap forward between the second (2G) and third generations (3G), so much so that we might regard it as a paradigm shift. Previous generations of mobile technology have simply enabled us to make standard telephone calls with a wireless device, a technology that has been around since 1946, or even 1879 if Hughes' early experiments with portable equipment are regarded as telephony [14]. 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 specifications, showing the huge change between the two platforms*

Table 1: 2G versus 3G characteristics

	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

The change from dial up talk time to mobile IP 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. Interactive applications may involve client-server communications, peer-to-peer or a combination of both. System features such as

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

Over-The-Air (OTA) provisioning allow application downloads, updates and data synchronization, while still enabling concurrent voice calls.

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. Many countries have seen huge amounts of money paid for bandwidth licenses. For example, in the United Kingdom five mobile service providers paid a total of 22,477,400,000 (GBP) for UK bandwidth, and that does not include any kind of infrastructure investment, which is likely to cost another ten billion pounds or so. For multinational companies who have bid for bandwidth across the world, the total investment is staggering. The returns, potentially, may also be significant. There are a number of 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. One statistic states that in 2002 the global number of mobile subscribers overtook the number of fixed line subscribers [15]. Of course this statistic needs to be taken with a pinch of salt, 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 [16]. 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. Those with content will set the pace*' Sandy Climan (MD Entertainment Media Ventures). [17]

Technology, health and social issues

Whatever the plans of the mobile industry, there are, of course, still many problems to solve in the mobilisation of commerce. One issue that has had to be addressed is that of applying common standards, whether in the 3G infrastructure or the operating systems and development platforms of mobile phones. Although there is much less fragmentation in the technologies used by the service providers than there was with earlier generations of mobile systems, and the Symbian operating system has united a number of handset manufacturers, there will still be various incompatibilities to deal with. There are also a number of technological and physical constraints to development, including finite bandwidth, limits to economic signal coverage imposed by geography and demographics and available hardware (such as limitations on battery life).

Security is, of course, a major concern, and a number of technologies have to be put in place to ensure it. Existing certificate based authentication is one important component at the software level but is just one piece of a larger and more complex jigsaw. Privacy is again an important issue, particularly with reference to location based services. The ability to track the whereabouts of individuals can be used for very constructive purposes, but it also has its potentially negative side. Another key cause for concern is electromagnetic field (EMF) exposure. There have been many questions raised in recent years over the possible long term effects of radiation from handsets and masts, and not all of the concerns raised have yet been allayed. Perhaps the most immediately concerning aspect of mobilized software systems is the trade off between being empowered and being overpowered. Unless mobile information can be effectively managed and filtered there is a danger of computing being just too pervasive. "*I can just see it. You'll be driving home from work and the phone will ring. 'Your refrigerator is on the line', the car will say; 'it wants you to pick up some milk on your way home'. To which my response will be: 'Tell the refrigerator I'm in a meeting'.*" [18]

4 Writing tomorrow's mobile software

'*We march backwards into the future*' - Marshall McLuhan

For the developer planning development on 3G telephones, 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 [19], 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 phones

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.

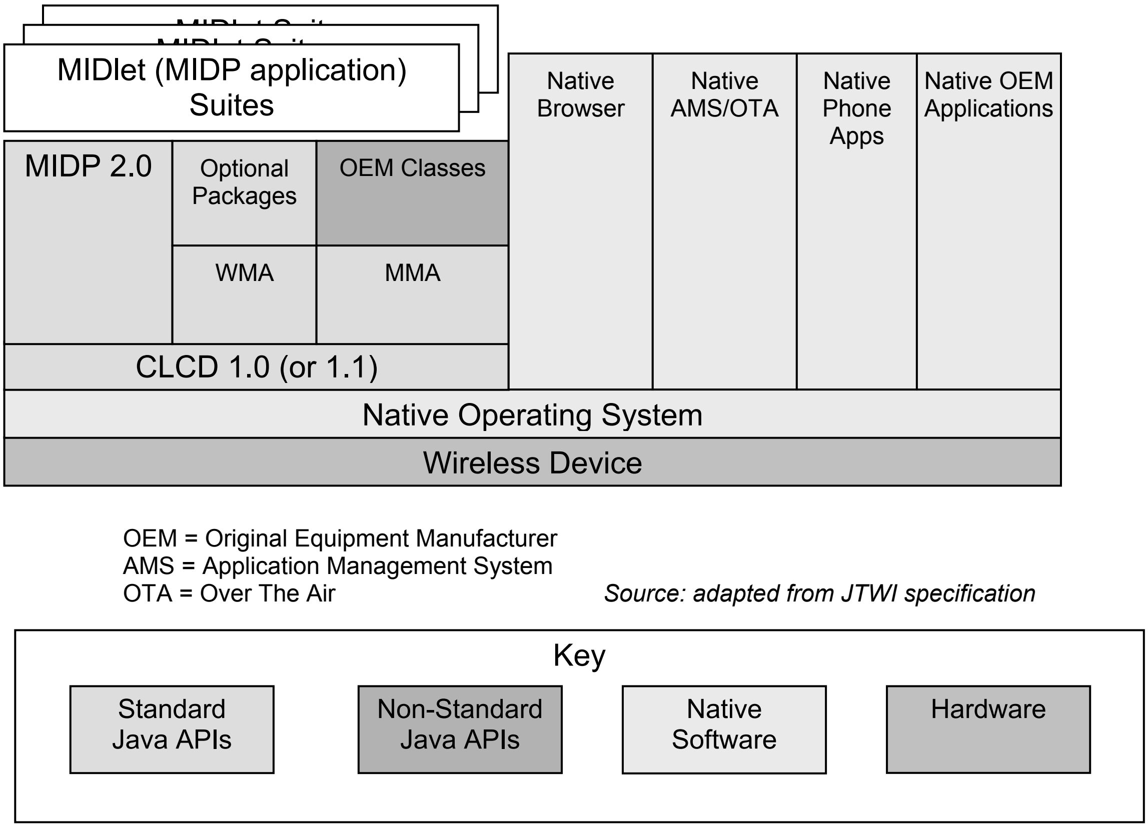

Figure 1: JTWI phone architecture

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. Of these perhaps the most interesting from an application development point of view are the location APIs. Locations include not just the standard location information such as latitude, longitude and timestamps, but also encompass attributes such as altitude, course and speed. The location APIs support knowledge about addresses, including building floor and room information. Applications can also manipulate landmarks in categories. 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.

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.

There are seven packages containing the additional classes and interfaces of the MIDP: *javax.microedition.lcdui* and *javax.microedition.lcdui.game* (providing user interface features at two levels of portability), *javax.microedition.media* and *javax.microedition.media.control* (for sound support) *javax.microedition.pks* (for certificate based authentication), *javax.microedition.rms* (record management system for persistence) and *javax.microedition.midlet*, which provides the framework for application classes deployed into a MIDP environment. In addition, there are extra classes in the *javax.microedition.io* and *java.util* packages.

Applications using the MIDP and CLDC APIs use the MIDlet application model. To be a MIDLet a class must extend the javax.microedition.midlet.MIDLet abstract class and must only use functionality consistent with the MIDP specification. The set of APIs available to a MIDP developer will be the combination of the classes in the CLDC and MIDP.

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 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. The various architectural components are shown in Figure 2.

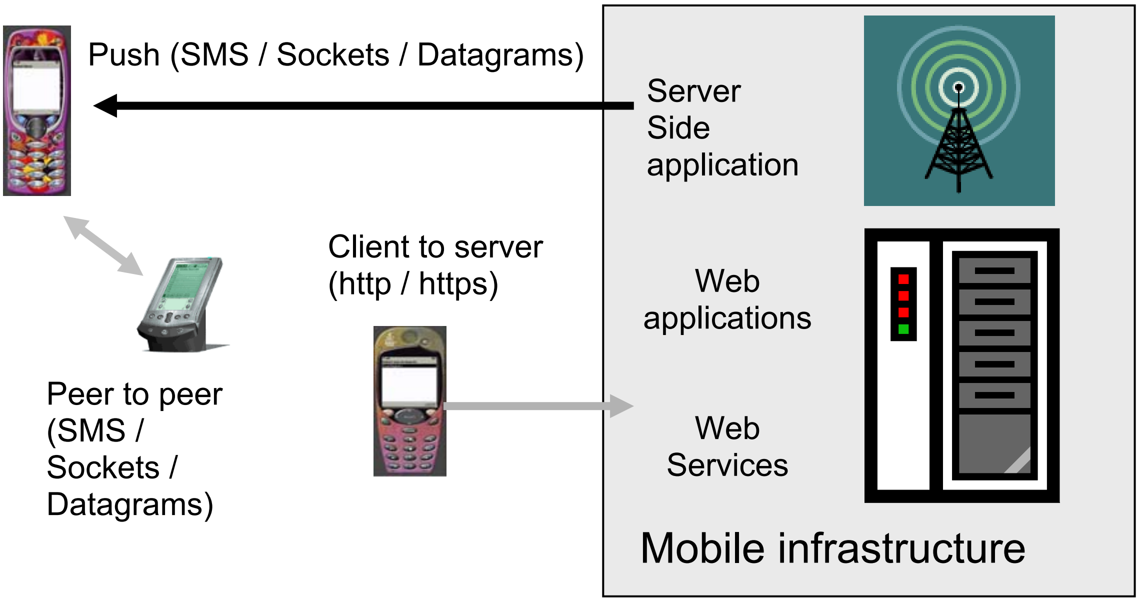

Figure 2: Components of a mobile application architecture

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) 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.

Regardless of the underlying communication mechanism being used, the MIDlet handles pushed connections using the push registry. Figure 3 shows the main features of the MIDP 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.

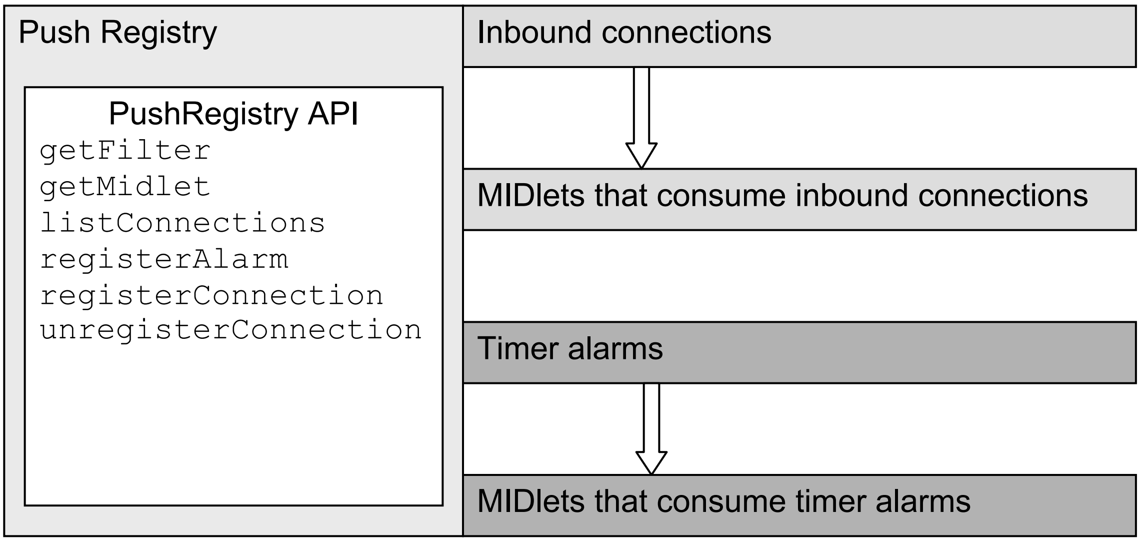

Figure 3: The MIDP push registry

5 An example application

The Mobile Conference Delegate application has been developed to demonstrate some key features of a mobile application that uses JTWI, specifically persistent data and push technology. The idea is that all delegates to a conference could download the application to their phones and register their mobile phone numbers with the server side part of the application.* Such an application could have many features, such as location finders for the conference venue, to help delegates find their way round the conference using the location APIs. However, the application that has been developed focuses on demonstrating a mobile database of conference schedule information: the days, times and titles of the conference sessions and who is presenting them. Since this is the kind of information that can be subject to change, for example if a speaker is delayed or there are rooming problems, it is useful to enable dynamic updates to this data. The mobile persistent store maintains conference schedule data using the MIDP Record Management System (RMS). This enables the schedule 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 schedule 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 SMS and HTTP connections. When there is a central change to the conference schedule, an SMS message is broadcast to the push registry connections of all the registered mobile devices. On receiving this message,

* For phones with fixed IP addresses, these could be used instead of telephone numbers.

the mobile device 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.

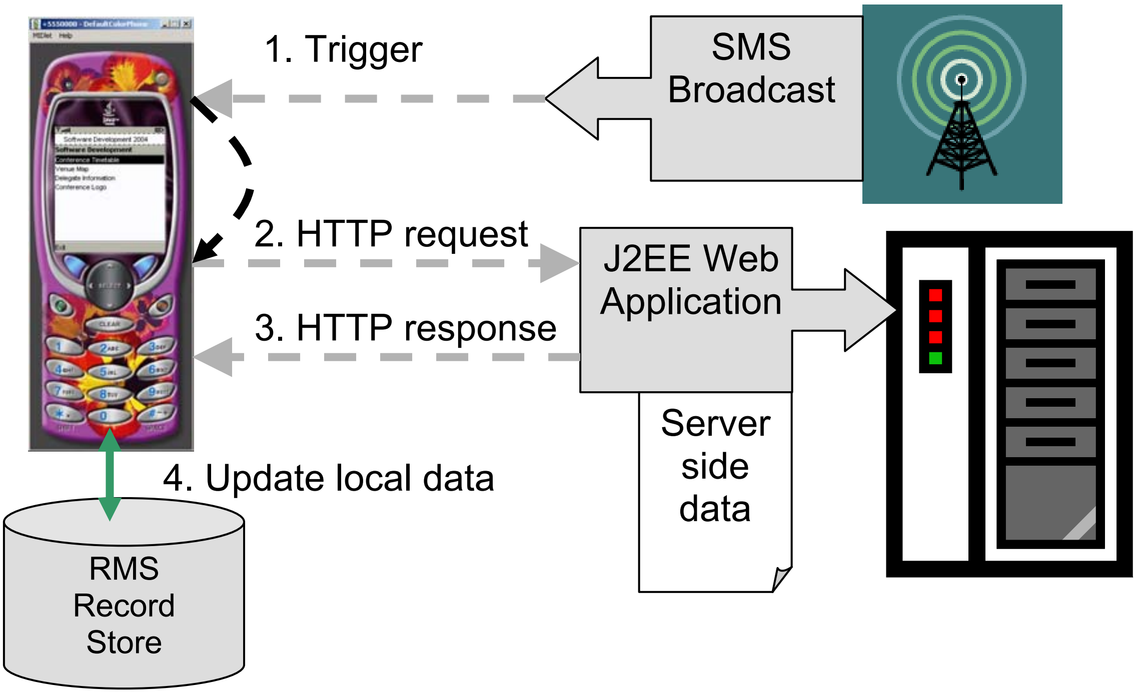

Figure 4: The push sequence

A screen dump showing the system in action appears in Figure 5. Note that the schedule information for day 1 of the conference 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.

Figure 5: The mobile delegate application in action

Questions and futures

There are a number of questions about mobile software architectures that are raised by the mobile delegate 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 systems 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), 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.

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 applications to a fast growing market.

6 References

[1] Thomson, W., in Adams, D. (1999). *Predicting the Future*. Retrieved March 16th, 2004, from http://www.bbc.co.uk/dna/h2g2/A216433

[2] Leuck, W., in Adams, D. (1999). *Predicting the Future*. Retrieved March 16th, 2004, from http://www.bbc.co.uk/dna/h2g2/A216433

[3] AskOxford. (2004). *Email SMS and Online Chat*. Retrieved March 16th, 2004, from http://www.askoxford.com/betterwriting/emoticons/?view=uk

[4] Sigurdson. J. (2001). *WAP OFF - Origin, Failure and Future.* Retrieved March 16th, 2004, from http://www.telecomvisions.com/articles/pdf/wap-off.pdf

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

[6] Coupland, D. (1998). Reverse Time Capsule. *Wired Magazine*. Retrieved March 16th, 2004 from http://www.wired.com/wired/scenarios/capsule.html

[7] Hughes, N. (2003). Mobiles and the environment – here and now! *Receiver*, 8
Retrieved March 16th, 2004, from http://www.receiver.vodafone.com/archive/index.html

[8] Scuka, D. (2003). Mobile Internet for the silver market. *Receiver*, 8 Retrieved March 16th, 2004, from http://www.receiver.vodafone.com/archive/index.html

[9] Gomes, P. (2003) WYSINWYG – What You See is Not What You Get. *Receiver*, 8
Retrieved March 16th, 2004, from http://www.receiver.vodafone.com/archive/index.html

[10] Glaser. P. (2001). The data aura - The outstanding possibilities of future UMTS services. *Receiver*, 4. Retrieved March 16th, 2004, from http://www.receiver.vodafone.com/archive/index.html

[11] ITC. (1968). *Joe 90 – Promotional Brochure*. Retrieved March 16th, 2004, from http://www.bigrat.co.uk/contents/premise.html

[12] Hirschhorn. K. (2000). An Orange Vision for a Wirefree Future. *Receiver*, 1. Retrieved March 16th, 2004, from http://www.receiver.vodafone.com/archive/index.html

[13] Gelsinger, P., Gargini, P., Parker, G. & Yu, A. (1989). Microprocessors Circa 2000. *IEEE Spectrum*, October 1989.

[14] Farley, T. (1999). *Mobile Telephone History*. Retrieved March 16th, 2004 from http://www.privateline.com/PCS/history.htm

[15] Shaw, R. (2003). *Mobile Overtakes Fixed: What Happens Next?* Retrieved March 16th, 2004, from http://www.itu.int/osg/spu/presentations/2003/prague-presentation-mobile-overtakes-fixed.pdf

[16] Foley, K. (2002). Africa goes mobile. *NUA Internet Surveys*. Retrieved March 16th, 2004 from http://www.nua.ie/surveys/analysis/weekly_editorial/archives/issue1no216.html

[17] Climans, S., in (author uncredited, 2001). *An Introduction to Third Generation Mobile Comms*. Retrieved March 16th, 2004, from http://www.three-g.net/3g_intro.pdf

[18] Thackara, J. (2002). Pervasive computing. *Receiver*, 5. Retrieved March 16th, 2004, from http://www.receiver.vodafone.com/archive/index.html

[19] Sun Microsystems. (2003). *JSR-000185 Java Technology for the Wireless Industry 1.0 (Final Release)*. Retrieved March 12th, 2004, from http://jcp.org/aboutJava/communityprocess/final/jsr185/index.html.