Beyond Mobile Telephony: Exploring Opportunities for Applications on the Mobile Phone Handset

Shwetak N. Patel and Gregory D. Abowd

College of Computing & GVU Center, 801 Atlantic Drive, Atlanta, GA 30332-0280, USA

{shwetak, abowd}@cc.gatech.edu

Abstract

As mobile phone handsets attain increasing capabilities, we see many more opportunities for novel applications development. These handsets are typically characterized as constrained computing platforms, due to limitations in computing, storage and interface capabilities. Specialized development environments, such as J2ME, allow for cross-platform development that respects these limitations. While it is important to respect these resource constraints, we also want to highlight some of the unique features of mobile phones, such as high quality audio, constant connectivity and comfortable form factor for use as device to interact with the physical world. In this paper, we discuss the J2ME development environment for current mobile phones and demonstrate applications we have developed that respect the resource constraints while simultaneously exploiting the unique features of these commercially available devices. This paper is intended to whet the appetite of potential developers and designers for this important emerging platform.

Category of paper relative to this special issue:

- Applications, particularly in non-traditional settings
- Novel functions for handheld devices
- Interaction techniques for small devices
- Wireless-phone computing

1. Introduction

The widespread availability and popularity of mobile phones presents us with new opportunities for constantly available services. Compared with their predecessors of even a few years ago, today's mobile phones come with significant computational capabilities, a nearly always-on network connection, and audio and display systems. Mobile phones are popular (for example, there are over 110 million mobile phone subscribers in the US) and the typical user has one nearby at almost all times. The mobile phone handset is a

viable wearable computing alternative, and we should consider this opportunity more seriously.

Applications for mobile phones have been limited to standard telephony or messaging services, PDA activities (calendar and contact management) and games. In this paper, we want to demonstrate that viewing a handset as a generic computing platform with some unique capabilities opens up the possibility for a variety of interesting applications.

These handsets do have resource constraints with respect to storage, computation, network bandwidth and physical I/O. As a large variety of handsets are introduced to the market, there is a need for a fairly consistent and flexible programming environment. Sun introduced the J2ME (Java 2 Micro Edition) development environment in 1999 to provide a cross-platform programming standard geared towards mobile and embedded devices with limited storage and processing power. More interestingly, the unique features of these handsets, specifically high quality audio, nearly constant network connectivity and comfortable form factor, open up many opportunities for creative application development. J2ME can be extended to exploit vendor-specific features of a handset.

In this paper, we present the J2ME development architecture for mobile devices, specifically mobile phone handsets. We explore the development of J2ME applications for these handsets, known as MIDlets, in terms of memory consumption, storage, graphics, audio, user input, networking and IO, and security. We survey the unique functionality of commercially available Java-enabled phones, emphasizing features that can be exploited through extensions to J2ME. We then demonstrate a variety of applications developed for a particular handset, the Motorola iDEN i95cl. These applications and interaction techniques demonstrate basic features of mobile and ubiquitous computing applications, namely automated capture, context-awareness and seamless interaction between the electronic and physical world. While each application is itself interesting, the main purpose of this paper is to encourage others to similarly explore handset applications more creatively.

2. J2ME Programming Environment for Mobile Phones

J2ME is a highly optimized runtime environment that looks similar to the well-known J2SE (Java 2 Standard Edition). However, J2ME does not contain many of the heavyweight classes found in J2SE. Rather those classes have been rewritten with a smaller footprint and with low power consumption in mind. The Connected, Limited Device Configuration (CLDC) and the Mobile Information Device Profile (MIDP) have emerged as J2ME standards for mobile phone applications development.

2.1 J2ME Architecture

The J2ME programming suite for mobile phones consists of three main layers: Java VM, Connected Limited Device Configuration (CLDC), and Mobile Information Device

Profile (MIDP). OEM specific java APIs provide additional functionality not available in the J2ME suite. Applications may be developed on top of any of these layers, but most MIDlets are typically built using the MIDP API.

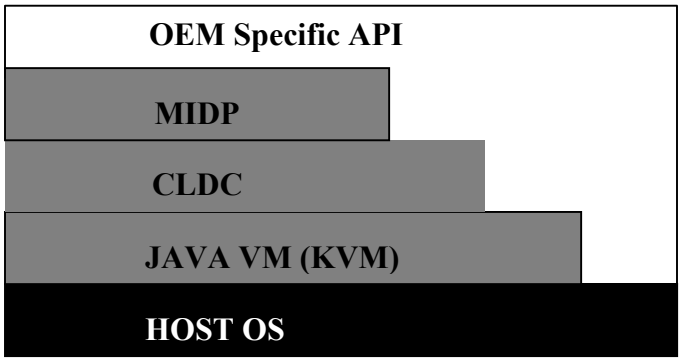

Figure1: J2ME Development Architecture Stack

- **Java VM**: The mobile Java VM, also know as KVM (Kilobyte Virtual Memory), usually resides on top of the host OS. The KVM is about 40–80 KB in size and is specifically targeted toward devices with limited power and memory. The KVM runs on 16 or 32-bit RISC/CISC microprocessors with as little as 160 KB of total available memory. The Java HotSpot VM is also available which is designed for devices with 32-bit RISC microprocessors and 512KB to 1MB of available memory. The Java VM is very device specific and requires the most porting across platforms.

- **CLDC**: CLDC is the configuration layer of the J2ME architecture. It provides the core lower level functionality for memory and resource constrained devices. CLDC consists of four main packages: java.io, java.lang, java.util, java.microedition.io, which facilitates the bare minimum IO and utilities.

- **MIDP**: MIDP provides the core application functionality for mobile devices, which includes network connectivity, data storage, and user interfaces. The set of APIs in MIDP forms the basis of the programming interface for mobile phone applications development. MIDP and CLDC constitute the complete Java programming and runtime environment for mobile phones. Devices implementing MIDP and CLDC must adhere to some minimum requirements. The device screen must have at least a 96X54 pixel screen size, at least 128 KB of RAM, and 32 KB of heap space.

- **OEM Specific API**: In addition to the CLDC and MIDP API, manufacturers offer additional APIs specific to their products. These APIs allow access to proprietary features and functionality available only on certain mobile phones. Such things may include access to the phone's audio, camera, vibrator, or lighting systems.

2.2 Addressing resource constraints for the mobile handset

Limited resources and a small display screen present many challenges when developing applications for mobile phones. We present various J2ME development constraints that may limit applications and present ways to workaround these limitations.

2.2.1 Memory

A typical high-end J2ME-enabled mobile phone like the Motorola iDEN i95cl has about 1.5 MB of data memory, 1.5 MB program memory, and 640 KB of heap memory. The limited program space requires very compact code for each application. Most handsets already enforce a maximum MIDlet size to ensure proper installation and execution of applications on their respective devices. For example, the Motorola iDEN i95cl has a maximum MIDlet size of 160 KB and the Nokia 3410 has a maximum size of 30 KB. Some handsets may not enforce a maximum limit, but MIDlets larger than the specified maximum size may not perform properly.

Currently there are two methods of trimming MIDlets. The first way is to manually reduce the size of resource files. This may involve reducing the quality or size of images used in the application. Another way is to run a MIDlet through an obfuscation tool. An obfuscator trims a MIDlet by removing and renaming methods and variables. ProGuard and JAX are two example obfuscators that can be used on MIDlets. An obfuscator can help ensure that all unnecessary code is eliminated from the MIDlet.

The limited amount of heap space may cause problems during runtime. Large static buffers and temporary stores may quickly use up most of the available heap space. So, programs must avoid having redundant stores of data in memory. In addition, limiting the number of object instances can help save heap space. One way to conserve heap space is to dereference objects to null as soon as the object or data structure is no longer needed so that the garbage collector can release the space.

2.2.2 Storage

MIDP specifies at least 8 KB of persistent data storage called the Record Management System (RMS). Some of the high-end mobile phones have over 1.5 MB of storage. J2ME uses a record-oriented database for the persistent data store. In addition, most

handhelds have a limit of 1024 addressable record stores, independent of total storage space. Since a series small reads and writes are slow, it is faster to read and write large chunks of data at a time. The reason for this is that there is a big overhead to initiate the record store and transferring large chunks ensure we only pay the penalty once. However, if the chunks are too big then there will be noticeable delays during the application runtime. The optimum record size specified for the i95cl is 512 bytes.

2.2.3 Graphics and Graphical User Interfaces

The latest mobile phones offer color display screens of about 120X150 pixels with 8 to 16 bit color graphics capabilities. The default MIDP UI package is the Limited Connected Device User Interface (LCDUI). This provides a limited number of UI components like lists, forms, choice groups and labels. LCDUI is used to develop graphical applications relatively quickly. The LCDUI Canvas class can also be extended to draw custom graphics and get access to raw key presses. Many handsets also support PNG and JPEG images that can be directly displayed in a MIDlet. However, images should be limited to only a few since they require lots of memory. Even though LCDUI provides a lightweight widget for quick development, it is very limited in the kinds of UIs that can be built. Motorola's Lightweight Window Toolkit (LWT) is an extension to MIDP to address these limitations. It provides a very flexible UI API similar to the J2SE Swing toolkit. Currently the LWT provides almost complete control over the layout of components and allows for easy custom component development. LWT allows various components to be placed within a container with absolute or relative locations. Each component also has an associated component listener for event handling.

One problem with directly drawing to the LCDUI or LWT canvases, especially with an animation, is flicker. MIDP does not directly support double buffering. So, one solution is to have the canvas thread paint the canvas off screen and then switch the contents at the end of the repaint call. There may still be a slight flicker but it is usually as a result of the LCD panel on the mobile phone.

2.2.4 Audio

Most handsets support direct MIDI playback provided that the file resides locally. A Java Specification Request (JSR) for a more sophisticated mobile media framework, called the Mobile Media API, has been submitted and is currently in development. The Mobile Media API is a lightweight version of the Java Media Framework so that it fits within the constraints of the CLDC. The MIDP 2.0 specifications include a subset of the Mobile Media API.

For now some handsets like the i95cl provide access to the audio system with a VoiceNote API. The VoiceNote API uses the phone's audio to provide a way for applications to record and playback audio files. The API does not directly facilitate moving forward or backward in the audio samples. However, manually traversing the

byte array containing the audio can allow some control over the audio if the encoding scheme is known and we use this technique in the first application discussed below.

2.2.5 User Input

MIDP provides low-level and high-level access to receive user input from a UI. The low-level control allows access to the raw key presses on the mobile phone such as the button on the outside of phones. The high-level control is always associated with a widget and actions are only reported when the user interacts with that specific widget. Both low-level and high-level controls may be used in the same MIDlet, but not at the same time.

Quick text entry is an issue with handsets. Many phones support smart text entry schemes like T9, which allows for faster text input by reducing the number of multiple key presses for characters and symbols. Most handsets automatically enable T9 when a text field or text box has focus of control, thus requiring almost not additional coding. For more flexibility, the T9 API allows for the direct handling of T9 engine events and allows the setting of custom text from within a MIDlet.

2.2.6 Networking and external interfaces

J2ME provides a variety of networking features and many handset manufacturers offer functionality beyond what is specified in MIDP. Some networking features are HTTP, HTTPS, TCP, SSL Secure Sockets, Server Sockets, and UDP. The number of concurrent sockets depends on the available resources available on the mobile phone. A typical high-end mobile will support 1 Server Socket, 7 TCP sockets, and 12 UDP sockets. J2ME also supports serial port access on the mobile phone for tethered devices like GPS receivers, barcode scanners, and other peripherals.

Since networking sockets are usually blocking, J2ME threads are used to process network activity while the normal application flow continues. There is not a limit on the number of threads that can be spawned during a MIDlet session, but the available memory and the limit on the number of concurrent sockets limits the number of threads that can be created.

Packet data cellular network connections offered by providers like Nextel and AT&T provide about 14.4 kbps of bandwidth. Cellular networks typically have high latency and are very “bursty” in nature, which may make developing network streaming applications difficult. Streaming applications need to buffer part of the content in local memory. Most network application benefit by sending large chunks of data at a time to avoid the latency delays between data segments.

Higher bandwidth network connections are already available by Nextel via their Packetstream Gold plan, which utilizes compression technology to achieve about 56 kbps. Sprint’s 3G network will provide up to 144 kbps of packet stream data.

2.2.7 Security

MIDP does not directly support security protocols. Most security is provided through the OEM APIs like SSL sockets and HTTPS. Many OEM APIs like Motorola offer lightweight cryptography such as message digest (MD5 and SHA-1), ciphers (DES, DESede, AES, and RC4), digital signatures (ECDSA) and key agreements (DH and ECDH). MIDP 2.0 only incorporates SSL sockets and HTTPS sockets in their new specifications.

3. Some novel applications

While the resource characteristics of handheld devices do constrain application development somewhat, it is the promise of their novel features that should drive more creative application development in the future. We now present a collection of J2ME applications developed at Georgia Tech for the Motorola iDEN i95cl. The i95cl has a large 120X168 pixel screen with 8 bits of color graphics. The phone has 1.5 MB of program and data space. The phone is serviced with a static routable IP address and the standard 14.4 kbps packetstream data service, provided by Nextel.

We have previously described some of the main features of mobile and ubiquitous computing applications: [Abowd *et al.* 2002]

- automated capture of live experiences for later access;
- context-awareness, leveraging automatically sensed information as implicit input to affect application behavior; and
- promoting a natural and seamless interaction between the physical and electronic worlds.

We will first present an example of an automated capture application to support near-term recall of conversations that benefits from the exceptional microphone capabilities of the handset. We next show two examples of context-aware behavior that leverage the constant connectivity of the device. The first example demonstrates a previously motivated context-aware messaging service and the second example provides unique context-aware capabilities for universal remote controls. Both of these examples exploit an indoor positioning system developed as part of the Aware Home Research Initiative. The last few applications demonstrate a novel, laser-assisted selection technique that establishes two-way connections between the handset and physical artifacts in the environment. The comfortable form factor of the i95cl makes it an ideal pointing device for natural, at-a-distance interaction with the physical world.

3.1 The Personal Audio Loop

Have you ever had difficulty resuming an interrupted conversation with another person because of some interruption? The Personal Audio Loop (PAL) is designed as a near-term audio memory device to help recover from interrupted conversations. PAL continuously records the last 15 minutes (or some other fixed time duration) of audio and provides a simple graphical interface and navigation strategy to allow an individual to skim the recorded conversation and "browse" the audio in hopes of recovering the content of an interrupted conversation. PAL is not an archiving service, a clear privacy concern. Rather, PAL is a near-term memory aid.

A requirement for PAL is reasonable quality audio recording at all times, and the mobile handset is the first commercial platform to provide this. We have built a number of PAL prototypes on various handhelds, with two problems. First, the quality of the microphone on the handhelds was often too poor to be able to understand the recorded audio. Second, recording only worked when the handheld was placed out in the open and nearby the people being recorded. The i95cl handset, like many other mobile phones and different from most PDAs, has an excellent microphone. We found that the i95cl's microphone is sensitive enough to pick up voices in a meeting room with the phone closed and in a shirt pocket. The PAL application uses the VoiceNote API to get access to the phone's VOCODER. In short, this handset is an ideal audio recording device, more suitable to the PAL application than any previous device we explored.

Ideally, the PAL application runs continually on the handset. The default mode is recording, but that mode is switched to either pause (to halt recording, for example, during a phone conversation) or playback to browse the recorded audio to recover from a lost conversation. Since the VoiceNote API does not allow direct control over the audio stream, we take the raw audio byte array and traverse it to playback at specific times within the audio stream. The phone has enough memory to support up to 1.5 hours of audio; we chose an arbitrary limit of 15 minutes for the recorded audio buffer.

In playback mode, quick and simple navigation through the audio stream is important. Figure 2 shows how PAL is controlled, both using the available side buttons of the handset as well as an optional graphical display for visual feedback. When a user needs to re-listen to part of a conversation, the up button is used to skip back a fixed amount of time in the recorded audio stream. The down button skips forward. This simple forward and backward jumping allows a user to skim a conversation to find a portion that reminds them of some salient feature of a previous conversation. It remains an empirical design question what the optimal forward and backward skip increments should be. For example, one popular digital video recorders on the market today provides a skip forward of 30 seconds and a skip backward of 7 seconds for browsing recorded television. Some researchers have explored audio skimming techniques [Arons, 1993], but not for the particular case of primarily backward skimming that would dominate a near-term reminder application like PAL.

PAL can be operated with the handset closed for complete heads-up operation. However, the optional graphical display provides a little more information to facilitate skimming. Operating on the metaphor of an analogue clock, the display shows the full 15-minute

audio buffer as a circle. There are two moving "hands" on the clock display. The first hand (in blue) moves during recording to show that audio is being recorded. As the recording hand sweeps across the clockface, it leaves a lighter blue trail. Simple features of the recorded audio (eg., amplitude) can be used to alter this recording trail (not shown in Figure 2). The second movable hand shows the playback position (in red) relative to the recording position. By looking at the display, the user can position the playback point relative to the current recording point. Hashmarks along the perimeter of the clock face indicate minute marks, to further facilitate positioning the playback pointer relative to the recording point.

Figure 2: PAL Application running on the right and control on the i95cl for the PAL application on the right.

3.2 A context-aware universal remote

Many researchers are exploring the challenges of handling context. The most popular form of context is location, so we are interested in seeing what kind of applications can be deployed on mobile phones that take advantage of knowing where the device is located. Much commercial effort is going into emergency services based on location,

such as e-911 in the U.S. The cellular telephony infrastructure provides a way to estimate the position of a handset based on signal strengths to a collection of base stations. Service providers do not make this positioning information available in the programming environment for the handhelds. Motorola provides a GPS chip for some of its phones (eg., the i88s handset) with a Location API extension to its J2ME environment, and other vendors will likely provide similar capabilities. For indoor environments, it is not clear that either of these solutions would work very well, for reasons of resolution and coverage. A desirable solution is to use existing indoor location services that communicate to the handset via its network connection. In the Aware Home Research Initiative at Georgia Tech, we have developed a room-level positioning system for tracking people using passive RF ID. Assuming we know the identity of the person holding the handset at any time, we can inform the handset where it is located based on the location of that individual.

A popular application for handheld devices is the universal remote control. Many people are already comfortable carrying their handset with them all the time and the comfortable form factor can be exploited for other in-home applications. We explored how to enable the mobile phone as a universal remote. As the number of remotely controllable devices in the home increases, we need a way to reduce the number remote controls and ease their use. One problem, addressed by Nichols *et al.* (2002), attempts to generate the interfaces for controlling a device automatically on a handheld platform.

One of the challenges for a universal remote is how to select the device to control. Our first solution was to leverage an indoor location service and the network connectivity to the phone to augment the universal remote application with knowledge of where it was being operated. Figure 3 shows the floor plan interface for our mobile phone universal remote control application. This interface allows zooming in and out between the overall floor plan and individual rooms. By default, the view selected is the room view for the current location of the user and device and inside this room view is a list of controllable devices that can be selected for further interaction. As the user changes rooms, the applications updates appropriately with new devices. This scheme helps reduce the number of items the user must search through to find the needed device. The user can control devices in other rooms by navigating the interface to the room containing that device. The user can also zoom out to different viewing levels, such as a single floor plan or the entire house. The application also takes advantage of the external buttons available on the phone. For example, the up/down buttons on the side of the i95cl (see Figure 2) can dim lights or change the HVAC temperature. Our particular application can also be used while outside the house to check the status of the lights or to turn on the HVAC system.

Figure 3: Screenshots of Universal Remote at different modes of control (floor level, room level, and device level)

3.3 Laser-assisted selection of physical objects

We are interested in how the constant companion of the mobile phone can facilitate our interactions with the physical world. We have considered using it as a selection device for initiating interactions with physical objects in the environment. Laser pointers, for example, are easy to use and can provide a natural feedback mechanism for pointing at objects from a distance. Several researchers have investigated how to detect what object a laser pointer is selecting. Most of those techniques have involved using computer vision to track the laser spot on a surface. We opted for a strategy that would not require the camera infrastructure, similar to the FindIt Flashlight (Paradiso et al, 2002). The user simply points a laser signal at an active tag (see Figure 4), which can respond when hit (see Patel & Abowd, 2003 for a more detailed account).

We integrated a laser pointer onto the Motorola handset and controlled it through the buttons on the phone (see Figure 4). By modulating the laser signal, we can send messages from the phone to the active tags. The active tags associated with specific devices in the environment detect these messages. The tags are connected to some network infrastructure, which provides a two-way interaction mechanism between environment and handset.

One demonstration of the 2-way laser-assisted selection was to integrate it with the universal remote control application described above. Previously, we showed how location information was used to filter the number of possible devices to select for subsequent control. Another alternative is to simply point at the device to control and have its interface delivered to the phone. We implemented this capability by attaching active tags to various home appliances. The phone would send its identity, that is, its static IP address, to an active tag, and the tag would respond to the phone via the phone's data network with information on how to control it.

We also considered using this 2-way selection technique outdoors. Active tags can be placed on road signs and billboards, using large tags connected to some wireless telephony service. When the billboard is spotted, the user selects it in order to receive additional information to the handset. At an airport, these active sensor tags can be placed in the logos of the airline companies. As you walk through the airport, you can simply identify one of these tags and the gate and flight information is directly sent to the handset.

Figure 4: On the left, a laser instrumentation of Motorola iDEN i95cl. In the middle, an example of a long-range active tag that responds and decodes. On the right, a prototype of an active tag embedded within a light switch and wall plate.

A particular instance of this outdoor interaction is the Listen 2 Me Digital Poster application we developed for radio station posters placed in the environment. Figure 5 shows an example of interaction with the local Georgia Tech student FM radio station, WREK. The digital posters advertising the radio station have active laser tags built into them. After a user selects one of these posters with a laser instrumented handheld, an audio sample from the radio station is streamed to the handheld and played back on the handheld. About 5 seconds of the audio is buffered before playback begins, because of the high latency of the network. The audio sample stream comes back either through the voice network or the data network. The mode of transaction is determined when the laser first selects the poster. If the message sent to the poster contains an IP address, then the audio returns through the data network and is played back directly on the mobile phone. If the message sent to the poster contains the handsets phone number, the audio returns through a phone call placed to the mobile phone. The phone call solution may be the most attractive solution for mobile phones that do not have the appropriate audio decoders. Mobile phones with advanced audio systems (supporting MP3 playback, for example) may request higher quality audio samples.

Figure 5: Radio Station Playback Screenshots

Conclusions

The purpose of this paper was to explore and further motivate the use of a mobile phone handset as a generic computing platform. There are important resource constraints to consider and emerging development environments, such as J2ME are mature enough to enable a larger community to program applications for the rich variety of commercial handsets in light of these constraints. In addition, most handsets have special features that can be exploited for a variety of personal and mobile applications. In this paper, we exploited the advanced audio capabilities, near-ubiquitous network connectivity and comfortable form factor of a particular handset, the Motorola i95cl, to demonstrate how more creative applications can be produced. We demonstrated applications for automated capture of audio, context-aware capabilities and natural interaction with the physical environment.

Though we have not subjected these applications to any empirical user study, it is clear that many technological and usability challenges remain. How do we provide simple switching between the many applications that could run on a handheld, some of which we suggest should be running nearly all the time (e.g., the near-term reminder service of PAL) and some of which we all expect to work as usual (standard telephony and messaging)? Power consumption remains an issue. The PAL application is as power hungry as normal talk time on the Motorola handset, currently limiting it to 2 hours continuous use. The laser selection technique is powered by a separate identical battery that fits within the handset. Despite these remaining concerns, we believe applications like the ones presented in this paper offer a compelling glance at a viable general-purpose computing platform with many of the advantages of instantaneous interaction that has driven the wearable and ubiquitous computing research communities.

Acknowledgments

The authors would like to thank Khai Truong and Mike Holloway for initial inspiration of the Personal Audio Loop, as well as access to a variety of handheld implementation of PAL to base our work upon. A special thanks goes to Motorola Inc., in particular Joe

Dvorak of iDEN Advanced Technologies, for the continued support of the Aware Home Research Initiative; and the donation of the i95cl handsets and service from Nextel required for this work.

References

1. Abowd, G.D., E. D. Mynatt and T. Rodden. (2002) The human experience. *IEEE Pervasive Computing*. Inaugural issue focusing on reaching Weiser's vision. Volume 1, Number 1, pp. 48-57.
2. Arons, B. (1993) SpeechSkimmer: Interactively Skimming Recorded. *Proceedings of the ACM Symposium on User Interface Software and Technology (UIST '93)*, p.187-196.
3. Ma, H. and J. A. Paradiso. (2002) The FindIT Flashlight: Responsive Tagging Based on Optically Triggered Microprocessor Wakeup. *Proceedings of Ubicomp 2002*, Springer-Verlag Lecture Notes in Computer Science, Volume 2498, pp. 160-167, 2002.
4. Motorola Global Telecom Solution Sector. (2002) "*i95cl Multi-Communication Device J2ME Developers' Guide"*. Motorola, 2002.
5. Nichols, J., B. A. Myers, M. Higgins, J. Hughes, T. K. Harris, R. Rosenfeld and M. Pignol. (2002) Generating remote control interfaces for complex appliances. *Proceedings of the 15th annual ACM symposium on User interface software and technology (UIST 2002)*, ACM Press, Paris, France, pp 161—170.
6. Patel, S.N. and G.D. Abowd. (2003) A 2-way Laser-assisted Selection Scheme for Handhelds in a Physical Environment. Technical Note submitted for review to *Ubicomp 2003.*