A development platform and execution environment for mobile applications

Guillermo Licea Sandoval, Ernesto Esliter Chávez, Juan Carlos Pérez Caballero

glicea@uabc.mx

Facultad de Ciencias Químicas e Ingeniería

Universidad Autónoma de Baja California

Calzada Tecnológico 14418, Mesa de Otay

Tijuana, Baja California, México

Abstract

Small computing devices are everywhere changing the way people communicate and interact, however applications for these devices are developed more or less with the same tools used for conventional computer applications. Developers of handheld computer applications can find few options for development. In this paper we present MADEE a development and execution environment for mobile information systems running on handheld computers. This environment allows the implementation of mobile information systems faster and easier than using conventional developing tools, including support for conventional computer applications.

Keywords: mobile information systems, handheld computer

1. Introduction

Through the years computing has evolved in a way that now when we talk about computer systems, we can not refer just to desktop computers, laptops or workstations, but to several custom programmable devices that supports many different kind of tasks.

Among this variety of non conventional computing devices, the most popular ones are handheld devices. A handheld device is a self contained, extremely portable device that supports communication and management of information. Features expected in a device to be considered handheld device are: [1]

- It must operate without cables, except temporarily (recharging or synchronizing information with a desktop computer)
- It must be easily used in one's hand, not resting on a table
- It must allow the addition of applications or support internet connectivity

There are four types of handheld devices: mobile phones (cell phones), pagers, handheld computers (also known as Personal Digital Assistants or PDAs) and communicators (a combination of the other three). From the point of view of the applications that these devices can execute, the most interesting are handheld computers.

A handheld computer is a device commonly larger and heavier than a mobile phone that includes address book, to do list and calendar functions, among others. Most handheld computers include:

- A touch screen that allows to point and select using fingers or a small stylus.
- Data input using a small keyboard or character recognition.

- Support for electronic mail, to do list, note taking and synchronization with desktop computers. Newer handheld computers include support for internet connection using internal or external hardware.
- Allow developers to write native applications.

The visible benefits of the use of handheld computers had promoted the increment of the variety and quantity of these handheld devices. Each year are sold 13 millions and the estimation for 2005 is 71 millions, according to IDC [2].

Storage capacity, computing power, size of display are some limitations of handheld computers when compared with conventional personal computers, but this devices are always at hand when needed changing the way people do their tasks in organizations. Some years ago handheld computers were used only by some people, but now is very common to see lawyers, accountants, medical doctors, teachers, among other [3].

Most popular handheld computers operating systems are Windows CE [4] and Palm OS [5]. Windows CE is a customized version of the Microsoft Windows operating system that fits in a handheld device. Palm OS was designed from scratch, considering the special features of a handheld device.

As is known, the software is always behind the hardware, so it is necessary to developed different kind of applications to take advantage of the use of handheld devices. There are some programming language extensions to C, C++ or Java specifically designed to support handheld computer software development considering typical features of the two most popular operating systems. In addition, some authors have been working in the development of frameworks or platforms to facilitate application development for mobile environments.

In the rest of this paper we describe most popular mobile application development tools for handheld computers, MADEE the development tool proposed by the authors, and the results obtained until now.

2. State of the art of the mobile application development tools

In this section we present the most popular application development tools found in the literature. All of these tools support the development of mobile applications for different kind of handheld devices, including handheld computers.

Munson and Dewan developed Sync [6] a Java framework that allows asynchronous collaboration between mobile users. Sync provides object replication and allows users to access and update local replicas, and maintaining objects synchronized.

Joseph, Tauber and Kaashoek developed Rover [7] as part of a MIT research project. Rover provides a framework for building mobile applications based on a flexible client-server architecture. Applications built with Rover can use a distributed objects system.

Pico, Murphy and Roman developed LIME [8], a middleware written in Java that supports mobile application development. LIME allows coordination of mobile units through a shared space.

Alba and Favela developed COMAL [9], a framework for the development of collaborative applications for handheld computers based on Palm OS. COMAL applications have three parts: server, desktop, and handheld.

Litiu and Prakash developed DACIA [10], a mobile component framework that supports the development of collaborative applications that adapt to the available resources and allows user mobility.

Roth and Unger developed QuickStep [11] a platform for the development of synchronous groupware applications running on handheld devices. QuickStep provides communication and collaboration primitives that allow to concentrate on application-specific details.

Myers developed Pebbles [12] a project that includes the implementation of several handheld computer applications that allow the use PCs and handhelds together. In these applications, handhelds augment PCs, instead of replacing them.

Grundy, Wang and Hosking [13] developed a set of server components that provide collaborative solutions for thin clients as chat, email, annotations, to do list, notifications, among others. Components provide HTML and WML user interfaces that can be integrated to a specific server.

Bergenti, Poggi and Somacher developed C/Webtop [14], a web-oriented synchronous collaborative platform that allows to run applications on different kind of mobile or fixed devices like laptops, handheld computers, and smart phones.

There are some other commercial programming tools like Metrowerks Code Warrior for Palm OS [15] and Microsoft Embedded Visual Tools for Windows CE [16].

The tools reviewed in this section are useful when implementing the specific features of mobile or mobile collaborative applications. Sync, COMAL, DACIA, QuickStep, C/Webtop and the work of Grundy et al. are tools oriented to the development of mobile collaborative applications, considering specific features of this kind of applications. Rover and LIME are tools designed considering the

features of generic mobile applications. Pebbles is a tool that supports the use of desktop and handheld computers together. All of these tools support the implementation of specific features of mobile applications, but do not consider more generic features of typical information systems like user interface support and objects management, among others.

Code Warrior and Embedded Visual Tools are development tools that support the implementation of more generic handheld applications, however they do not consider the specific features of mobile applications.

Considering this, we proposed the construction of an operation environment and application development tool (named MADEE from Mobile Application Development and Execution Environment) that supports and make easier the development of mobile information systems that run on handheld computers, allowing communication and information sharing among users in an organization. MADEE includes support for the specific features of handheld mobile applications, but it supports as well, the development of information systems for desktop computers, laptops, and workstations. In the rest of this paper we concentrate on handheld computers mobile information systems.

3. MADEE, a development platform and execution environment

The first step in the development of MADEE was to study different types of handheld computers and the programming tools available. After an analysis of the pros and cons of the two most popular operating systems, Windows CE and Palm OS, we decide to use in our project devices running Windows CE, specifically the ones called Pocket PC [17].

Windows CE is based on the Win32 Application Programming Interface (API), this offers significant advantages for application development over different platforms. There are a number of

programming tools for the Win32 API like Visual C++ or Visual Basic [18], in addition to other programming tools for languages like Java.

The programming language we selected for constructing MADEE and the applications developed with it was Java [19], because this programming language allows to write applications that can be executed on different operating systems and computer platforms, including handheld computers, desktop computers, laptop computers, and workstations. There are many Java Virtual Machines for Windows CE, KVM from Sun Microsystems, Waba from Wabasoft, J9 from IBM and Jeode from Insignia solutions [18]. We decide to use Jeode because it complains almost 100 % with Personal Java definition (Java version 1.1.8).

The second step in the development of MADEE was the development of a Pocket PC mobile information system using the Java programming language as the only development tool.

We build a mobile information system (called MAST from Mobile Application for Supporting Teacher activities) to support the activities of a high school teacher or university professor. Figure 1 shows some screenshots of the application running on a HP Pocket PC.

a) Opening a course b) Updating student's data c) Updating student's notes d) Checking course attendance

Fig. 1. Screenshots of MAST (first version) running on a Pocket PC

MAST includes all the management tasks related to the creation, retrieving, and updating of courses with the associated students registered, homework and lab assignments, students course attendance, etc. A teacher must load MAST information system and open a course while a network is available (reachable), then he/she moves to the classroom and uses MAST (where possibly there are no access to a network). Once the teacher can reach a network again, he/she can update the course data. An ideal scenario is when the teacher can reach a network all the time to load applications and open/update courses, but it is not always possible. At this moment we have been using MAST in two consecutive semesters, in three different courses obtaining good feedback from users.

As third step we used the knowledge and experience obtained during the development of MAST to collect and specify the requirements that were considered during the design and implementation of the first version of MADEE. In addition, to make a more complete design, we made an analysis of different software architectures [6 - 16] that can be adapted to support handheld computer mobile information systems development and execution. Figure 2 shows a general view of MADEE..

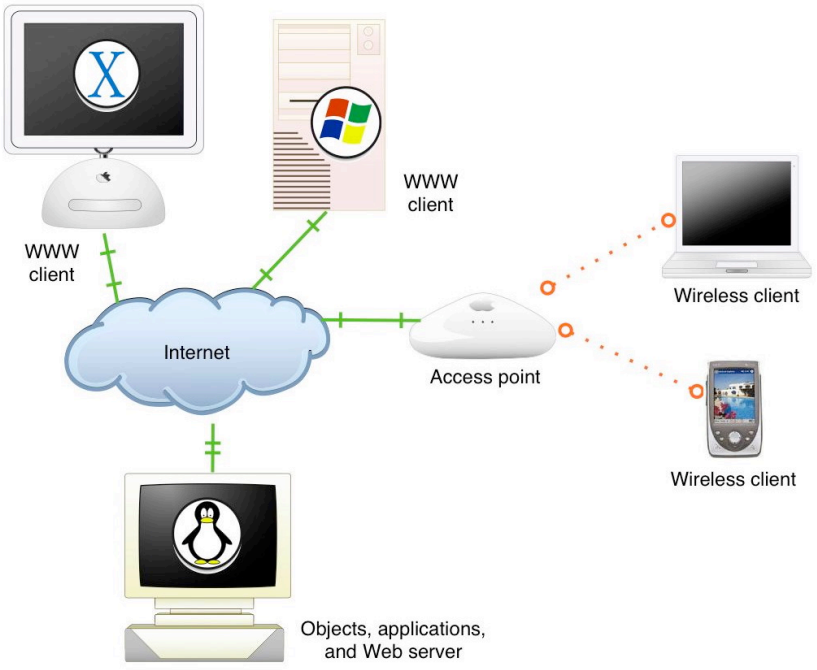


Fig. 2. General structure of MADEE

The access point is the base of a Wireless Local Area Network (WLAN). This device acts as a wireless hub, which allows connection between the different computers and devices in a Local Area Network. It is possible to connect up to 255 computers and devices, depending on the specific characteristics of the access point.

The structure of MADEE includes the use of personal computers (or workstations) that run a Web server and an object server.

3.1. MADEE's objects server

Object server stores objects (application data) grouped in object catalogs that are shared by different distributed applications. This server allows to access or update objects. Each time an object is updated, object server also updates web pages associated with the objects stored. These web pages can be accessed using the Web server.

Another function of the object server is to store and distribute application classes that are requested by wireless clients. Figure 3 illustrates the operation of the object server.

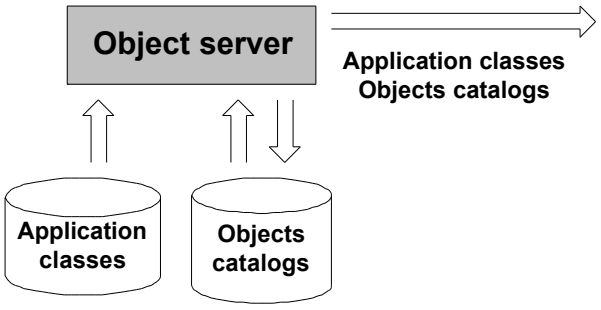


Fig. 3. Operation of MADEE object server

3.2. MADEE wireless clients

Wireless clients interact with the object server through the access point. These clients have an application launcher that allows obtaining application classes from the object server, as well as object catalogs according to the application needs. We use handheld computers as wireless clients, but it is possible to use laptop or palmtop computers. Figure 4 illustrates the operation of the application launcher executed on the wireless client.

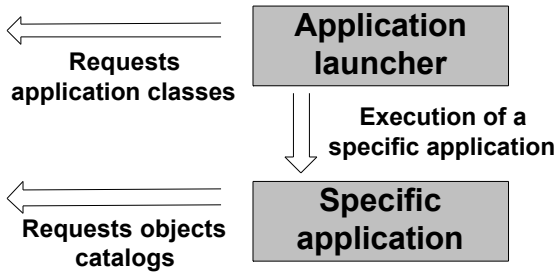


Fig. 4. Operation of MADEE wireless clients

Wireless and WWW clients interact with the object server executing mobile information systems that use MADEE Java classes. These Java classes hide low-level details of network connection to allow developers to concentrate on the functional characteristics of the application they are developing.

MADEE includes Java classes for the creation of specific applications or applets, catalogs, persistent objects, message passing between client and servers, among others. Figure 5 shows the relationship between these classes.

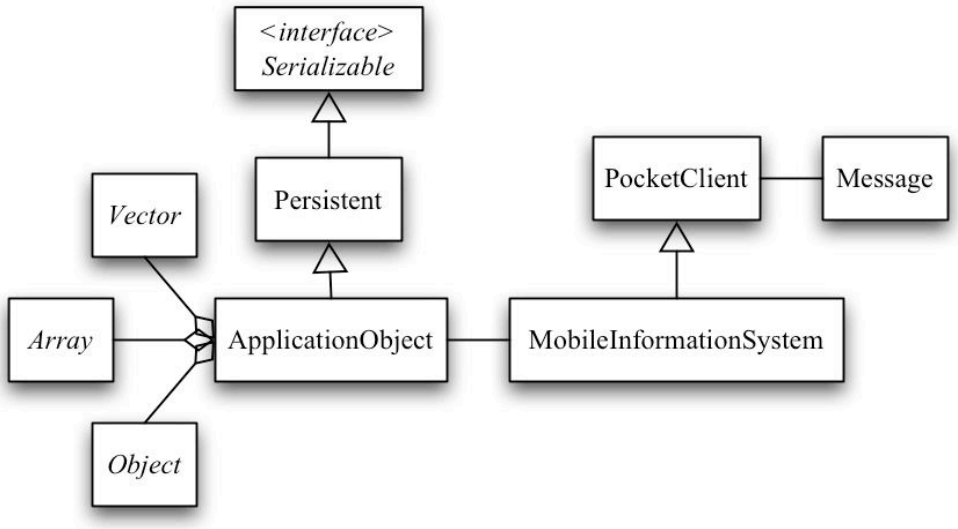


Fig. 5. A class diagram of MADEE's wireless client application (Mobile Information System)

3.3. Automatic generation of application user interface in MADEE

One task of application development that consumes a lot of time is user interface implementation. MADEE contains AGUILA [20], a class framework that includes classes that take advantage of Java's reflection property. Reflection is a java feature that allows an executing program to examine itself, obtaining information about the fields, constructors and methods included in the classes used by the objects created in the program. Made uses reflection property to know the internal structure of the objects for which input dialogs and views are generated.

MADEE's user interface class framework is based on automatic techniques, window managers and toolkits, event languages, and object-oriented programming [21]. Figure 6 shows the structure of this class framework.

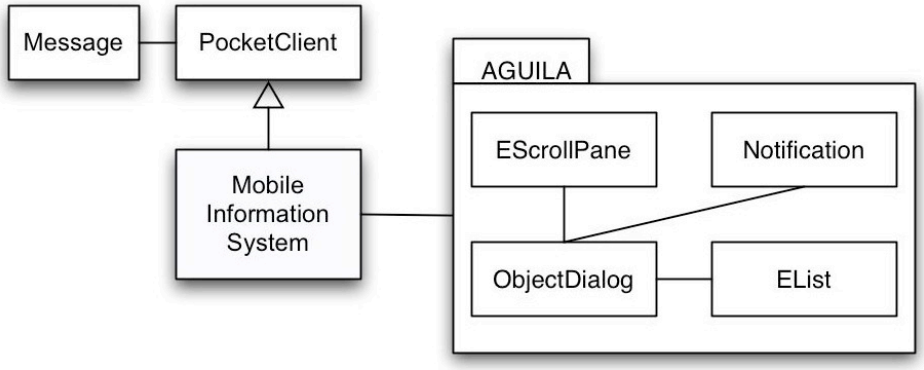


Fig. 6. Strucure of AGUILA class framework

AGUILA class framework includes 4 fundamental classes: *ObjectDialog*, *EScrollPane*, EList, and *Notification*. *EScrollPane* and *EList* extend two classes defined in the *java.awt* package, these extensions allow to generate input dialogs for complex objects like arrays and vectors. *ObjectDialog* is the main class of AGUILA, *ObjectDialog* generates input dialogs for user-defined objects and uses *EScrollPane* and *Elist* container objects. *Notification* is a support class for displaying different kind of messages and user notifications.

4. Results

As a test for MADEE, we rebuild MAST mobile information system. The original system required 28 Java classes for user interface, network connection and system functionality. The new version of MAST required just 11 Java classes, a considerable reduction of time and effort. This is because a programmer just need to write the classes representing the information that will be processed in the system, a programmer does not need to write any code for managing objects, persistence, information sharing, network connection, or dialog/views generation, these features were already considered in MADEE. Figure 7 shows the structure and relationship of the Java classes included in MAST.

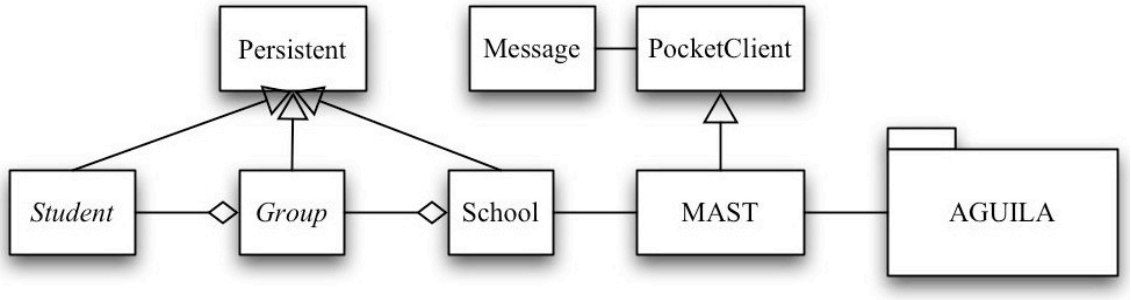


Fig. 7. Strucure of MAST mobile information system

PocketClient, *Message* and *Persistent* are classes that support connection and communication with MADEE's objects server, in addition to objects sharing and persistence. The Java code for *School*, *Group* and *Student* classes is presented next.

```
class School extends Persistent {
      private String schoolName;
      private String level;
      private int district;
      private String address;
      private Vector groups;

      // Constructors and methods
}
```

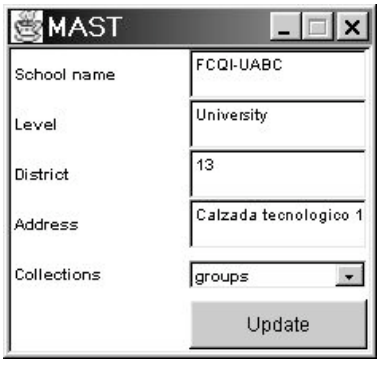
```
class Group extends Persistent {
      private String groupName;
      private int numberOfStudents;
      private Student students[ ];

      // Constructors and methods
}
```

```
class Student extends Persistent {
      private String id,;
      private String studentName;
      private int Semester;
      private int numberOfTasks;
      private String task[ ];
      private float note[ ];

      // Constructors and methods
}
```

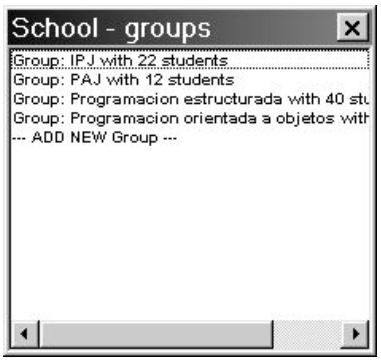
School, Group and Student classes extend *Persistent*, which allows to store and retrieve objects instances of these classes. It is not necessary to add special methods, but common methods must be written using the standard Java programming style. *MAST* is the main class of the mobile information system.

```
class MAST extends PocketClient {
   private School s;

   public MAST() {
      super("MAST");
      s = (School) School.open("uabc.school");  //1

      if (s == null)
         s = new School();
         add(new EScrollPane(s));                  // 2
         setVisible(true);
      }

   public void finish() {
      if (!s.save("uabc"))                              // 3
      new Notification("File can't be saved");
   }
}
```

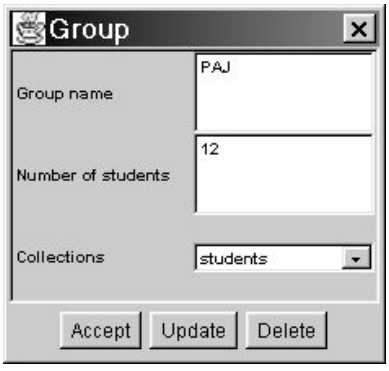
Lines 1 and 3 invoke *open* and *save* methods of class *Persistent* to retrieve and store a *School* objects. Line 2 adds an *EScrollPane* object to the application main window, after this, AGUILA classes take control of application's user interface to generate the dialogs and views needed for school information processing. Figure 8 shows MAST execution.



a) Dialog for a *School* object



b) Dialog for a *Group* vector



c) Dialog for a *Group* object

d) Dialog for a *Student* array e) Dialog for a Student object f) Dialog for a *String* array

Fig. 8. Dialogs generated automatically for a *School* object.

MAST mobile information system is a tool that facilitates the management activities in a teaching environment (typically a school). These management activities often distract teachers from their fundamental activities. In addition, MAST allows teachers to concentrate important information about their courses in a very small portable device that can be stored in a shirt pocket or a small briefcase, and that can be turned on/off very fast. This is an important advantage comparing to a laptop or desktop computer. A system like MAST running in a laptop computer can be used in a mobile environment, but would not be very practical. A system like MAST running in a desktop computer would be useful for the teacher for just some of the activities, but he/she can't take the computer to the classroom. In fact, we implemented a desktop version of MAST as a complement to the mobile version.

5. Conclusions and perspectives

There are several development tools for mobile applications proposed for several authors, however these tools have some disadvantages. Some tools support mobile or mobile collaborative features, but do not support more generic features of information systems. Other tools support more generic features of information systems, but do not support specific features of mobile applications.

In this paper we described MADEE, a development and execution environment for mobile information systems running on Pocket PCs and conventional computers.

Using MADEE, a user can develop specific medium size systems in a fraction of the time usually spent developing a mobile information system with a conventional programming language or programming tool.

As part of MADEE's development process, we implemented a useful mobile information system called MAST that supports teacher management activities. MAST have been used by several teachers in several courses and the feedback of users confirm the utility of the system.

At this point we have a functional version of MADEE that will evolve according to the feedback obtained as a result of testing information systems developed. We have been developing several mobile information systems as the one mentioned previously. At this moment, we are developing more mobile information systems for various scenarios (like schools, hospitals, supermarkets, and restaurants, among others), this includes real time, agent-based and workflow applications.

6. References

[1] Weiss, S. *Handheld usability*. John Wiley & Sons, 2002.

[2] IDC. IDC Remains Optimistic About Handheld Devices, Forecasts 71 Million Shipments by 2005. 20 of June 2001. http://www.idc.com.

[3] IDC. Handhelds Are Making Their Way to the Mainstream. 22 of February 2000. http://www.idc.com.

[4] Muench, C., Kath, R. *The Windows CE Technology tutorial: Windows powered solutions for the developer*. Addison-Wesley, 2000.

[5] Pogue, D. *Palm Pilot. The ultimate guide*. O'Reilly, 1999.

[6] Munson, J. P., Dewan, P. Sync: a Java framework for mobile collaborative applications. IEEE Computer, Vol. 30, No. 6. pp. 59-66. 1997

[7] Joseph, A. D., Tauber, J. A. and Kaashoek, M. F. Mobile computing with the Rover toolkit. IEEE Transactions on Computers Vol. 46, No. 3, pp. 337-352, 1997.

[8] Pico, G. P., Murphy, A. L. y Roman, G. *Developing mobile computing applications with LIME*. Proceedings of International Conference on Software Engineering 2000, pp. 766-769, 2000.

[9] Alba, M. y Favela, J. *Supporting handheld collaboration through COMAL*. Proceedings of 6th Internacional Workshop on Groupware, Isla Madeira, Portugal, pp. 52-59, 2000.

[10] Litiu, R., Prakash, A. *Developing adaptive groupware applications using a mobile component framework*. Procedings of CSCW 2000. Philadelphia, PA, USA. pp. 107-116. 2000.

[11] Roth, J., Unger, C. Using handheld devices in synchronous collaborative scenarios. Personal and ubiquitous computing, Springer-Verlag. Vol. 5, No. 4. pp.243-252. 2001.

[12] Myers, B. A. *Using handhelds and PCs together*. Communications of the ACM, Vol. 44, No. 11. pp. 34-41. 2001.

[13] Grundy, J., Wang, X. y Hosking, J. *Building multi-device, component-based, thin-client groupware: Issues and experiences*. Proceedings of the Third Australasian User Interfeces Conference, Melbourne, Australia, pp. 71-80, 2002.

[14] Bergenti, F., Poggi, A., Somacher, M. *A collaborative platform for fixed and mobile networks*. Communications of the ACM, Vol. 45, No. 11. pp. 39-44. 2002.

[15] Metrowerks. Codewarrior development studio for Palm. http://www.metrowerks.com.

[16] Microsoft. Embedded Visual Tools. http://msdn.microsoft.com.

[17] Hanttula, D. *Pocket PC Handbook*. Hewlett-Packard. 2001.

[18] Burkhardt, J., Henn, H., Hepper, S., Rintdorff, K., Schack, T. *Pervasive Computing*. Addison-Wesley. 2002.

[19] Arnold, K., Gosling, J., Holmes, D. *The Java Programming Language*, Third Edition. Addison-Wesley. 2000.

[20] Licea, G. *Automatic generation of user interfaces for lightweight applications for mobile devices (In Spanish)*. 2nd International conference on Informatics, Zacatecas, Mexico, 2003.

[21] Myers, B. A., Hudson, S. E., Pausch, R. *Past, present, and future of user interface software tools*. ACM Transactions on Computer-Human Interaction, Vol. 7, No. 1, pp. 3-28, 2000.