International Journal of Information Technology and Knowledge Management
January June 2009, Volume 2, No. 1, pp. 55-59

WINDOWS MOBILE OS POCKET PC BASED HANDS FREE INTERACTION THROUGH BLUETOOTH FOR ELEMATICS SERVICES IN AUTOMOBILES

N. Abid Ali Khan[*], G. Pavithra[**] & N. Karthikeyani[***]

According to Transportation Safety Group at the National Safety Council, the drivers using cell phone during the trips account for nearly 80 percent of car crashes. This paper addresses the issues of wireless networking standards impact on OEMs and highlights the necessity of a standalone mobile Pocket PC dedicated for automobiles. The application is developed with Microsoft .NET Compact Framework 3.5. This is a hands-free service which enables to recognize voice commands. Once the Bluetooth™ enhanced mobile phone is synchronized with the Pocket PC, the user can handle the calls and SMS messages using speech based commands. For prototyping ASUS A626 Pocket PC is used. The second half of the paper addresses an enhanced IT service to integrate with playing MP3 audio files, stored in USB. This paper gives focus design aspects of Telematics web based designs for automotive market.

Keywords: Protocol, Microsoft Windows Mobile, .NET Compact framework, Emulator, OEM, ROI.

1. Introduction

With the Pocket PC's user friendly application "Remote Presenter", it can be easily utilized to remotely link and control presentations on a computer via Bluetooth. Windows Mobile for Pocket PC features built-in Office Mobile applications for viewing and editing documents (such as Word, Excel and PowerPoint), a PC-like tap-to-select interface and a more robust set of customization options. This device does not have the ability to send or receive any calls. The aim of this paper is to explain the application that is developed and deployed on Pocket PC to make it behave as a hands free interactor for automobile users to make or receive calls with the help of Bluetooth™ enabled mobile phones. The purpose of designing Bluetooth™ based Hands Free Interaction is to enable the automobile users to access their Bluetooth™ enabled mobile phones using a series of voice commands, without taking one's hands off the wheel. The application developed includes the features such as compatibility with most Bluetooth Mobile phones, Hands-Free functionality with voice recognition, Automatic Message Reader, Playing MP3 files from the USB, connected to the Pocket PC.

1.1 Platform Selection

.NET platform is selected for developing applications in the Windows Mobile Pocket PC. Windows Mobile 6 is a platform for mobile devices, based on extended features of Windows CE 5.0. This is used in wide variety of third party hardware, such as Personal Digital Assistants (PDAs) and Smart phones. Microsoft Visual Studio 2008 and the Windows Mobile 6 SDK, make it possible to create software for the Windows Mobile platform in both native (Visual C++) and managed (Visual C#, Visual Basic .NET) code. The .NET Compact Framework inherits the full .NET Framework architecture of the CLR and managed code execution. The Microsoft .NET Compact Framework is an integral component on Windows Mobile and Windows Embedded CE devices that enables the developer to build and run managed applications and use Web services. The .NET Compact Framework includes an optimized common language runtime (CLR) and a subset of the .NET Framework class libraries, which supports features such as Windows Communication Foundation (WCF) and Windows Forms. It also contains classes that are designed exclusively for the .NET Compact Framework. The .NET Compact Framework supports Visual Basic and Visual C# development. It does not currently support C++ development.

The application that is discussed in this paper is developed with .NET Compact Framework 3.5 in ASP.NET using Visual Studio 2008 IDE. ASP.NET is a server-side solution. With ASP.NET, applications can be written in C# or Visual Basic .NET that reside on a Web Server, and perform complex processing, including creating user interface controls, and accessing databases. ASP.NET isolates the device characteristics from the application, making it straightforward to run one application on many difference device-types. From the above mentioned programming languages, Visual C# is chosen for developing the application. Visual C# is a "managed" development

[*] System Analyst- Automotive and Aerospace Centre of Excellence, Satyam Computer Services, Chennai-97, Tamil Nadu, INDIA, E-mail: abid_khan@satyam.com

[**] Department of Mathematics, Anna University, Tamil Nadu, INDIA, E-mail: paviannauniv@gmail.com

[***] Department of Computer Science, Madras University, Tamil Nadu, INDIA, E-mail: karthii26@gmail.com

language. It supports the .NET Compact Framework - a library of classes that perform a lot of frequently used programming tasks, to greatly simplify application development. The development tools for C# include a fully what-you-see-is-what-you-get user interface designer. Buttons and other controls can be dragged and dropped directly onto the application's window (called a "form" in managed programming), and then it is double-clicked to access the underlying code. This approach makes creating an application's user interface extremely fast and easy.

Visual Studio 2008 supports .NET Compact Framework 3.5 but does not have the support for deploying Windows Mobile 6 Professional device on it. The SDK is installed along with Visual Studio 2008 to develop the application in Windows Mobile 6 Professional device.

2. APPLICATION DESIGN

The overall functionality of the application development is represented with the help of use-case diagram in Figure 1.

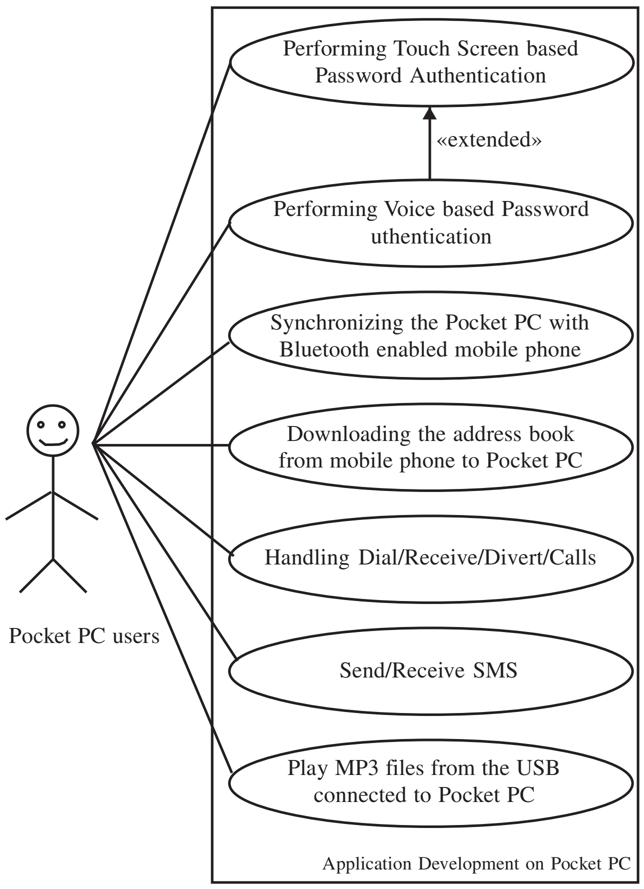

Performing Touch Screen based Password Authentication
«extended»
Performing Voice based Password uthentication
Synchronizing the Pocket PC with Bluetooth enabled mobile phone
Downloading the address book from mobile phone to Pocket PC
Handling Dial/Receive/Divert/Calls
Send/Receive SMS
Play MP3 files from the USB connected to Pocket PC
Pocket PC users
Application Development on Pocket PC

Figure 1: Use-Case Diagram of the Application Development on Pocket PC

The process flow of the entire application is shown as in Figure 2. The user has the privilege to Dial/Receive or Divert the call based on his requirement.

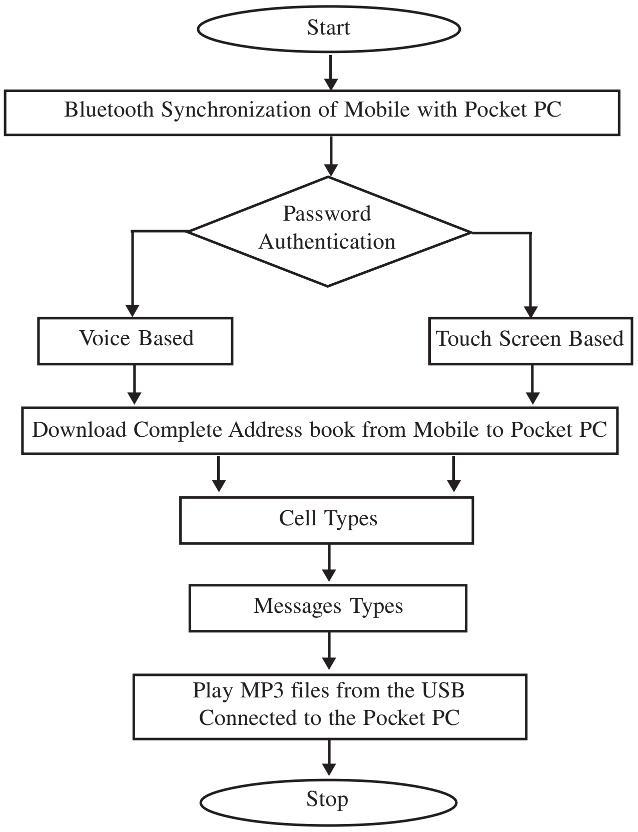

Start
Bluetooth Synchronization of Mobile with Pocket PC
Password Authentication
Voice Based
Touch Screen Based
Download Complete Address book from Mobile to Pocket PC
Cell Types
Messages Types
Play MP3 files from the USB Connected to the Pocket PC
Stop

Figure 2: Process Flow of the Application Development

For this purpose the call types is classified into three ways. Similarly the user can also send or receive the SMS. All this can be done either through touch screen or voice based commands. The process flow for call type and messaging is shown in Figure 3 and Figure 4.

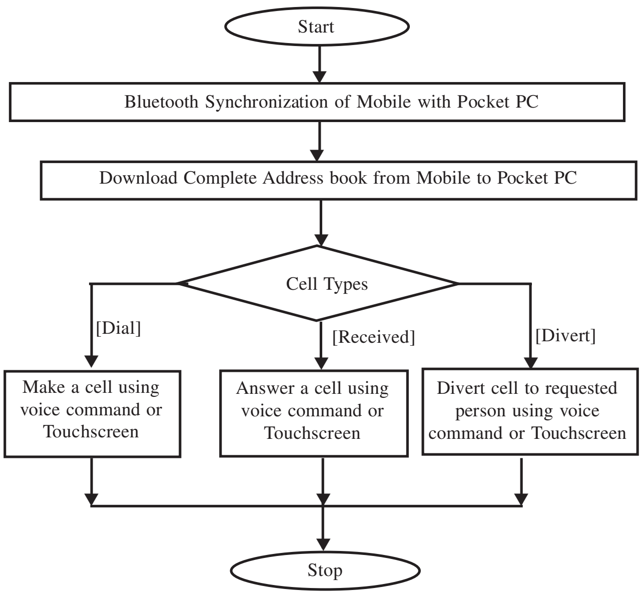

Start
Bluetooth Synchronization of Mobile with Pocket PC
Download Complete Address book from Mobile to Pocket PC
Cell Types
[Dial]
[Received]
[Divert]
Make a cell using voice command or Touchscreen
Answer a cell using voice command or Touchscreen
Divert cell to requested person using voice command or Touchscreen
Stop

Figure 3: Process Flow for Call Types

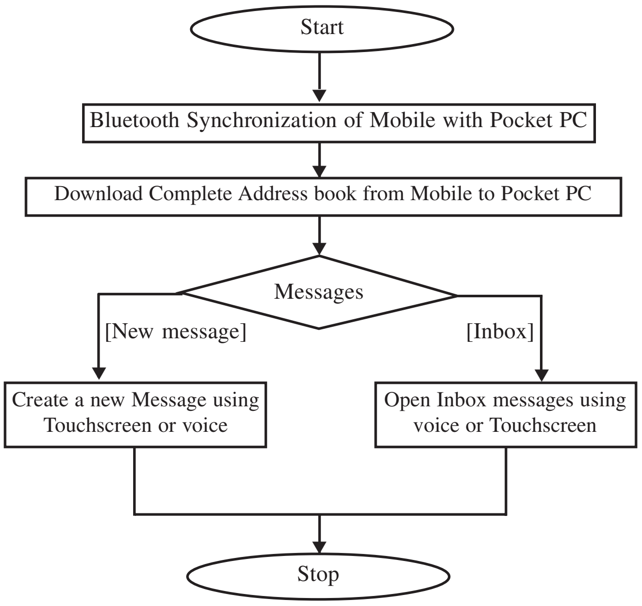
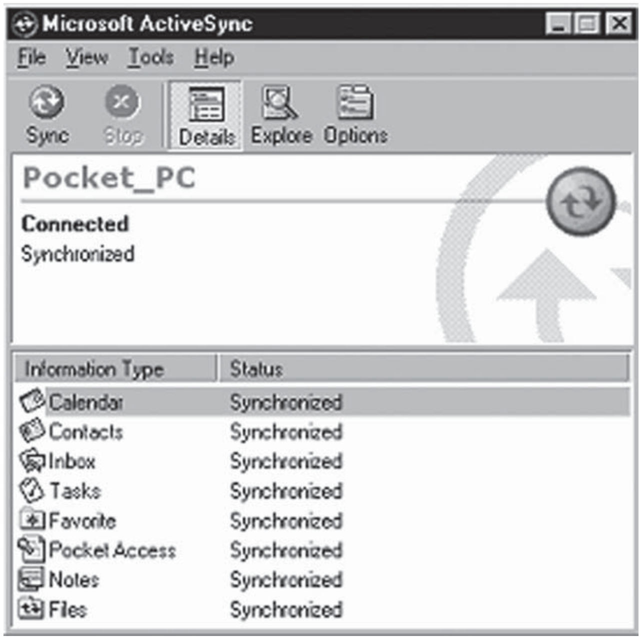

Start
Bluetooth Synchronization of Mobile with Pocket PC
Download Complete Address book from Mobile to Pocket PC
Messages
[New message]
[Inbox]
Create a new Message using Touchscreen or voice
Open Inbox messages using voice or Touchscreen
Stop

Figure 4: Process Flow for Messaging

3. POCKET PC CONNECTION WITH THE DESKTOP

The Pocket PC is connected to the desktop using ActiveSync 4.5 software. Active Sync runs on the PC and detects the presence of the Pocket PC. Once it is connected it will automatically synchronize the information from Desktop to Pocket PC. ActiveSync acts as the gateway between Windows-based PC and Windows Mobile-based device, enabling the transfer of Outlook information, Office documents, pictures, music, videos and applications to device. Once the connectivity synchronization is established, it allows users to play with files as shown in Figure 5.



Pocket_PC
Connected
Synchronized

Information Type	Status
Calendar	Synchronized
Contacts	Synchronized
Inbox	Synchronized
Tasks	Synchronized
Favorite	Synchronized
Pocket Access	Synchronized
Notes	Synchronized
Files	Synchronized

Figure 5: Connectivity Status

4. POCKET PC APPLICATION DEVELOPMENT USING WINDOWS MOBILE 6

Windows Mobile 6 extends the familiarity of the Windows desktop to Windows Mobile powered devices. Windows Mobile is based on Windows Embedded CE and supports the .NET Compact Framework. By using the Windows Mobile platform, it is possible to build innovative applications for mobile devices. This platform offers features such as data connectivity that allows seamless data transfer with enhanced security; rich API support such as Bluetooth™ and the Pocket Outlook Object Model (POOM); an extensive range of programming models that includes native code, managed code, and mobile web development; and device resources such as multithreading. Development time and cost is thus reduced by taking advantage of a familiar Windows development environment, a consistent programming model.

4.1 Creating the Application

Figure 6 shows the project workspace for the application specified.

On a Pocket PC, configuring a form as either a main program window or a dialog box is controlled by a single form property: the MinimizeBox property. When this property is set to true, a form operates as a main program window and displays a minimize box (a circle containing an "X") in the right side of the title bar. When MinimizeBox is set to false, a form operates as a dialog box and displays the 'OK' button (a circle containing "OK"). The standard on a Pocket PC is that a program continues to run when the user clicks on the minimize box. For certain types of tools and utilities, it can make sense for a program to shut down when the user closes the program's main form. Normally one can derive forms directly from the Form class that is defined in the System.Windows.Forms namespace as

Figure 6: Windows Form Design for Device Applications

specified in Listing 4.1.1. This is the default derivation to add a new form to the project.

```
using System;
using System.Drawing;
using System.Collections;
using System.ComponentModel;
using System.Windows.Forms;
namespace MainPlusOthers
{
   /// <summary>
   /// Summary description for FormDemo.
   /// </summary>
   public class FormDemo : System.Windows.Forms.Form
   {::
   : }
   /// Further Forms that can be integrated.
}
```

Listing 4.1.1: Defines a Sample Form

4.2 Building the Application

The following Listing 4.1.3 describes the OnPaint function which displays a message at run time.

```
protected override void OnPaint(PaintEventArgs e)
{
   // Create string to draw.
   string drawString = "Hello PocketPC";
   // Create font and brush.
   Font drawFont = new Font("Arial", 10, FontStyle.Regular);
SolidBrush drawBrush = new SolidBrush(Color.Black);
// Create point for upper-left corner of drawing.
   float x = 10.0F;
   float y = 10.0F;
// Draw string to screen.
   e.Graphics.DrawString(drawString, drawFont, drawBrush,
   x, y);
}
```

Listing 4.1.3: Defines a OnPaint Function

4.3 Debugging the Application

Very few programs work perfectly first time, which is why debugging is such an important part of the application development lifecycle. Visual Studio provides powerful debugging features for windows mobile application. Applications can be debugged both on the emulator, and on a real physical device. Emulator usage is a good way to test the application on differently-localized devices. The form is deployed in the Windows Mobile 6 Professional Emulator. The Device Emulator is a tool that mimics the behavior of a hardware platform for Windows Mobile. It provides a virtual hardware platform that can be used to test applications on multiple virtual devices. The Device Emulator runs code compiled for ARM microprocessors, and it provides a high degree of fidelity with an actual, target, consumer device. The Device Emulator supports the following features such as Configurable screen resolution, Flexible display orientation, Host-key combinations that support special functionality, serial port mappings, Storage card emulation along with networking support. In addition, this supports multiple development environments.

4.4 CAB Files for Delivering Windows Mobile Application

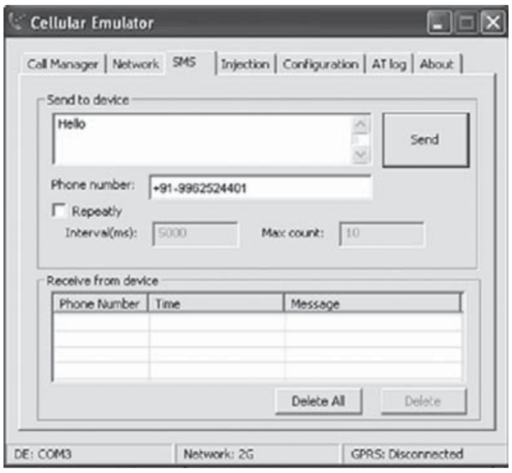
The output of the build process is an executable that will run on a Windows Mobile powered device. Before the application can be installed on the device, it is necessary to bundle the executable into a cabinet (.cab) file. A '.cab' file is created by creating a Smart Device CAB Project, which requires no additional piece of code. It can be directly added to the existing Visual Studio Solution for creating it. Using the browser, '.cab' file is copied onto a device. In addition, ActiveSync is bundled with a utility called Application Manager. Using this utility, the application contained in a '.cab' file can be installed onto a Windows Mobile powered device, and then a desktop computer-based installer application is written that invokes the Application Manager to install applications on a cradled device.

5. On Going Research and Results

The application that is developed on a Pocket PC is fully based on Touch screen feature of the device. For testing this application ASUS A626 is taken as a reference PDA. The ongoing research is to make the application work for voice based commands. This is achieved by importing the Speech Interfaces to the code to make the application running on Pocket PC. The Voice to text conversion and vice-versa will be handled by the Speech interfaces imported to the code. At this stage of development, sending and receiving SMS and managing the call between the Pocket PC and the mobile phone is designed and implemented using the emulators. Cellular Emulator is taken for testing the application. It allows the developer to simulate a 2G or 3G network on an emulator. One can also send SMS from cellular emulator to Windows Mobile 6 device emulator and vice-versa. It also supports making phone calls. The emulations between two emulators are shown in Figure 7 and Figure 8.

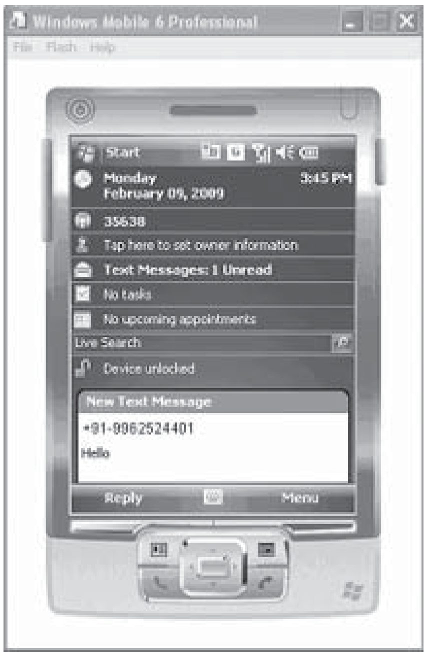
6. Conclusions

In this paper the preliminary need of Hands free communication service for Telematics is highlighted and



Send to device: Hello
Phone number: +91-9962524401
Repeatly: ☐
Interval(ms): 5000
Max count: 10
DE: COM3
Network: 2G
GPRS: Disconnected

Figure 7: Cellular Emulator



Monday
February 09, 2009
3:45 PM
35638
Tap here to set owner information
Text Messages: 1 Unread
No tasks
No upcoming appointments
Live Search
Device unlocked
New Text Message
+91-9962524401
Hello

Figure 8: Windows Mobile 6 Professional Emulator

the Research results implemented on ASUS A626 PDA were projected. The importance of application design, implementation and debugging for PDAs using Microsoft .Net Compact Framework is described. The application developed on the ASUS A626 is using the standard Bluetooth stack as a communication channel, which normally any cell phone service provider will offer at an affordable price. Irrespective of the type of the cellular services like 2G or 3G, the solutions that are provided over these standard interfaces will hold as a key business proposals with good Return on Investments for OEM specific IT service oriented companies for a long-run. The importance of Pocket PC based designs using such standard interface will address the mobile protocol convergence for future Telematics market.

7. ACKNOWLEDGMENTS

We would like to thank the faculty advisors Mr. Thirusangu of Anna University and Mrs. Velmayil of Madras University for allowing us to pursue Research at Satyam Computer Services. We would extend our thanks to Mr.S.Kannan of Anna University for his guidance. The authors would like to sincerely thank Mr. Venkatesh Chandrashekaran, Head of Aerospace and Automotive Centre of Excellence R&D lab and Research team for their support and cooperation.

References

[1] P. Bhagwat, Bluetooth: Technology for Short-Range Wireless Apps Internet Computing, *IEEE*, **5**, (3), (May/June 2001), 96–103.

[2] J. Valldorf W. Gessner, Telematics Digital Convergence- How to Cope with Emerging Standards and Protocols, Edition 4, Springer Berlin Heidelberg Publishers, (2004).

[3] Paul Yao: David Durant, .NET Compact Framework Programming with C#", Additional-Wesley Professional, (May 2004).

[4] John Viega and Gary McGraw. Building Secure Software. Addison-Wesley Pub. Co., (September 2001).

[5] Wei-Meng Lee, .NET Compact Framework Pocket Guide", Pocket References, O'Reilly, (2004).

[6] Adam Freeman and Alan Jones, Programming .NET Security, O'Reilly, (June 2003).

[7] Brian A. LaMacchia, Sebastian Lange, Matthew Lyons, Rudi Martin, Kevin T. Price. .NET Framework Security. Addison-Wesley, (April 2002).

[8] Dominick Baier, Security Bug *in* .NET Forms Authentication.

http://sourceforge.net/mailarchive/forum.php?thread_id=5671607&forum_id=24754

[9] J. D. Meier, Alex Mackman, Michael Dunner, and Srinath Vasireddy. Building Secure ASP.NET Applications*:* Authentication, Authorization, and Secure Communication.

http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnnetsec/html/SecNetch13.asp

[10] Erik Meijer and John Gough. Technical Overview of the Common Language Runtime.

http://research.microsoft.com/~emeijer/Papers/CLR.pdf

[11] Microsoft Corporation. Security Briefs: Strong Names and Security in the .NET Framework. *http://msdn.microsoft.com/netframework/?pull=/library/en-us/dnnetsec/html/strongNames.asp*

[12] .NET Framework Developer's Guide. Permissions.

http://msdn.microsoft.com/library/default.asp?url=/library/en-us/cpguide/html/cpconpermissions.asp

[13] Denis Pilipchuk. Java vs. .NET Security.

http://www.onjava.com/pub/a/onjava/2003/11/26/javavsdotnet.html

[14] *http://www.pocketpctalk.com/*