Mobile Information Access

Manuel Schröder

Abstract: This articles deals with mobile software, its development and usage. In the beginning a tourist’s vision is introduced. The following parts examine the current state of technology to realize this vision and expose parts which still have to be developed. After this examination the two major software-platforms where introduced which enable application-developers to create software for mobile devices. Subsequent chapter deals with general considerations and limitations regarding this kind of software. To conlcude this work a first realization of that tourist's vision is presented and introduced.

1 Motivation

Mobile devices let us use modern technology wherever and in any situation we want. To accomplish this a lot of technology has to be provided. This technology can be divided in hard- and software but another distinction is possible: Let us think of service-providers and -users.

Services are suitable to a common environment and task which help us to deal in a specific situation. A broker needs other services than a tourist. For example a tourist wants to do sightseeing in a foreign city and he doesn't know where to go. What he can do is to join a guided tour but this isn't very individual. Maybe he can use his mobile device to supersede the tourist-guide and thus make his own way. From the map on the display of his electronic device he can choose a route through the ancient parts of the city. Indications of items of interest appear also on the display-map. Short videos inform our tourist about the unique and historical architectural features of the current site is seen. Whenever he passes a shopping district, translated advertisements of sales pop up on the display. So if he wants to buy some goods he can pay them electronically and let them deliver to the right gate for the flight home.[1]

Now what is the current state of technology to accomplish this scenario? We already have the proper mobile hardware. What lacks is the suitable software support. Content has to be created. And the most important part is to provide this content in the right context.

[MK04] defines context as follows: “*Context we understand the set of suitable environmental states and settings concerning a user, which are relevant for a situation sensitive application in the process of adapting the services and information offered to the user*”.

Context can be divided into several categories which are the following:

• Task Context:	What is the user doing?
• Social Context:	Social aspects of the user such as friends, relatives and colleagues.
• Spatio-Temporal:	Time, location and movement. Context
• Environmental Context:	User surroundings like things, services, light, people and information accessed by the user
• Personal Context:	Disabilities and weight (Physiological context) Mood and expertise (Mental context)

[1] [Sa96] p. 3

But technology has limitations. According to [Sa96] mobile elements are resource-poor relative to static elements. So considerations in weight, power, size and ergonomics are recommended. Furthermore there are limitations in computational resources like processor speed, memory size and disk capacity and also limitations in display and visualization. Mobility is inherently hazardous in reference of loss and damage. Another aspect is mobile connectivity which is highly variable in performance and reliablility. Last these devices rely on a finite energy source.

2 Mobile Technology

Today’s mobile technology consists of mobile devices and their counterparts wireless networks.

2.1 Mobile Devices

Among mobile devices you can find cell-phones, PDAs and Smartphones. For PDAs exist two major platforms which are PalmOS (Palm Inc.) and the Pocket P200x-series from Microsoft. The leading platform for smartphones is SymbianOS (Nokia, Ericsson and Motorola). The Smartphone 200x-series from Microsoft uses WindowsCE as its operating system.[2]

2.2 Mobile Networks

The wireless networks can be divided in LAN (Local Area Networks) and WAN (Wide Area Networks). IEEE 802.11, also known as WLAN, Bluetooth and Infrared belong to the LAN category. GSM (Global System for Mobile Communications), GPRS (General Packet Radio Services) and UTMS (Universal Mobile Telecommunication System) are examples for WAN.[3]

3 Mobile Software

As in the beginning mentioned, we already have the proper mobile hardware. What lacks is the right software.

What follows are general considerations concerning mobile-software. Afterwards the two most important software-platforms are introduced to the reader. A comparison of these two platforms conclude this chapter.

3.1 General considerations

Mobile software has to be developed for mobile devices. These devices are designed for a special market. Markets can be categorized in a vertical and horizontal market segment. Vertical market segments shows the different types of products for example communication, automobile devices, intercommunication toys and washing machines or fridges. The horizontal market segment divides the types of products into categories which belong to the same product familiy.

For example a cellular phone belongs to another vertical market segment than a board-computer in a car but to the same as an organizer. The cell phone and the organizer are communicating-devices but they differ in equipment and functionality.

To overcome these broad range of technical configurations we need abstraction. We accomplish this by using a software-runtime which shields the application-developer from the hardware and provides him a homogenous software environment and a rich set of APIs. Another aspect is a higher portability of code.

This software environment also supplies some tools for the creation (compilation and verification), tests and cross-debugging and deployment (the actual upload to the device) of software.

---

[2] [Ct04] p. 244
[3] [GK] slide 8ff

The development-process takes place on at least two machines (Figure 1): The development computer and a mobile device. On the first one the design, implementation, debugging with an emulator and testing will take place. The second one helps to finalize the debugging, testing and the deployment.[4]

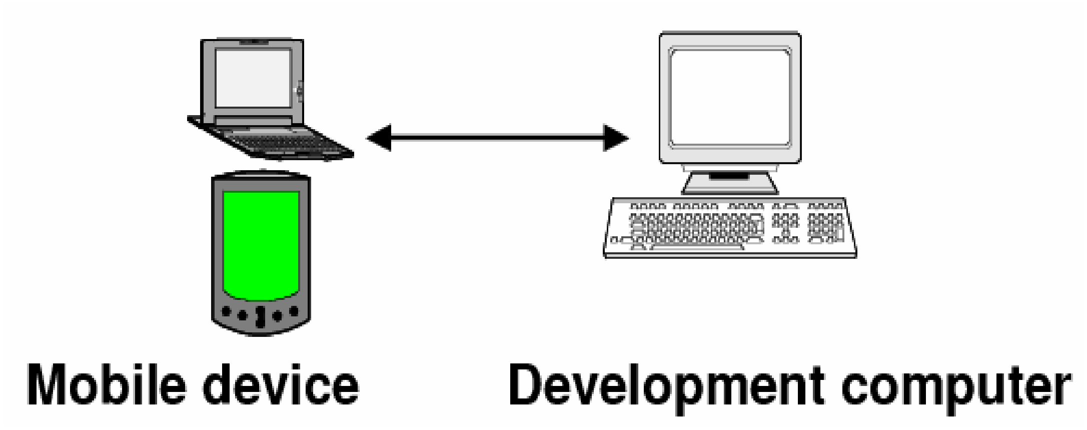


Figure 1 - Development Process

Portability is useful to develop software once and provide it on a rich set of hardware-platforms without adapting it to every single hardware configuration. Which not only takes a lot of time but causes a lot of costs.

But portability is only useful within a horizontal market segment. It makes no sense to expect from a program which runs on a washing machine that it runs also on a microwave oven and vice versa.

3.2 Mobile Development with Java

The Java-technology from SUN Microsystems comes in different editions. There are J2EE (Java 2 Enterprise Edition) for solid, complete and scalable internet business server solutions, J2SE (Java 2 Standard Edition) for the desktop computer market and the J2ME (Java 2 Micro Edition). This paper has the focus on the last one: J2ME.[5]

[4] [GK] slide 22
[5] [Sun00] p. 10

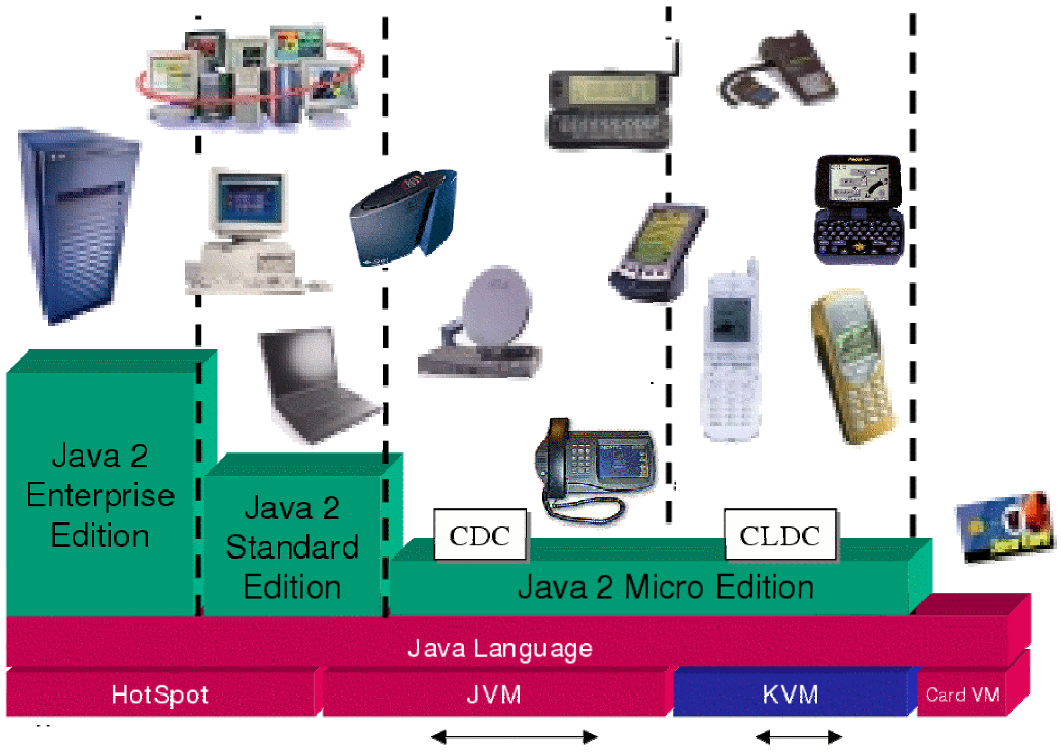


Figure 1- Java Editions

It's designed for consumer embedded devices, for service providers and content providers for small and resource-constrained devices. The features are the availabiltity of a highlevel object oriented programming language, safe network delivery and upward scalability with J2SE and J2EE. It adresses a range of devices from tiny commodities from pagers to set-top boxes. With the slogan „Running anywhere, any time, any place“ it also points out its portability of code.[6]

J2ME has two general configurations ([Sun00] p.16): CDC (Connected Device Configuration) and the CLDC (Connected, Limited Device Configuration). The latter stands for very simple user-interfaces, minimum memory budget (>128kB) and low bandwith, intermittent network connections. Examples for that configurations are cell phones, pagers and personal organizers. The first one – CDC – describes shared, fixed, connected information devices with a large range of user-interface capabilities and a memory budget from 2-16 MB. They also have a persistent, high bandwith network connection. Devices in this category are TV set-top boxes, Internet TVs, Internet enabled screenphones, high-end communicators and automobile entertainment- and/or navigation- systems.

In general you can say that the CLDC is a small subset of the CDC configuration which itself is a subset of the J2SE Edition with a little amount of special APIs. These special classes outside of the J2SE specification may not use the general java.* - namespace.

[6] [Sun00] p. 11

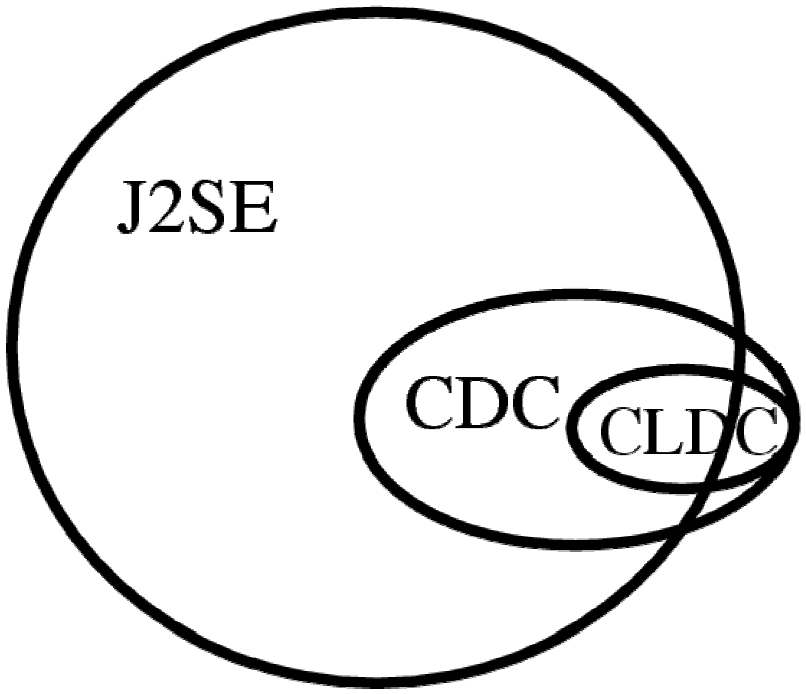


Figure 2 - Java API Dependencies

The idea of abstraction within the java-runtime is presented in Figure 3 from [Sun00]. First there's the Host Operating System, which is the core software-platform of the device. Above the OS the Java virtual machine (Java VM) abstracts the specialities of the current hardware-platform and creates the homogenous machine. Profile, on top of all, is a kind of contract between and application and a device family whereas a Configuration is a kind of contract between a profile implementer and a device's Java VM. A Configuration describes the available Java programming language features, all Java VM features and all basic Java libraries and APIs.

Figure 4 outlines the process of deployment. It all beginns with the source-file MyApp.java which is the input for Java-compiler javac to produce the compiled class-file MyApp.class. Verification takes place with the so called preverifier on the development computer, to produce a special format of a class file. After that step you can download the app to a mobile device. On that computer another verification takes place to ensure that this application conformes with the device's own configuration. When the app passes this final test it can be started with the Java VM (interpretation).

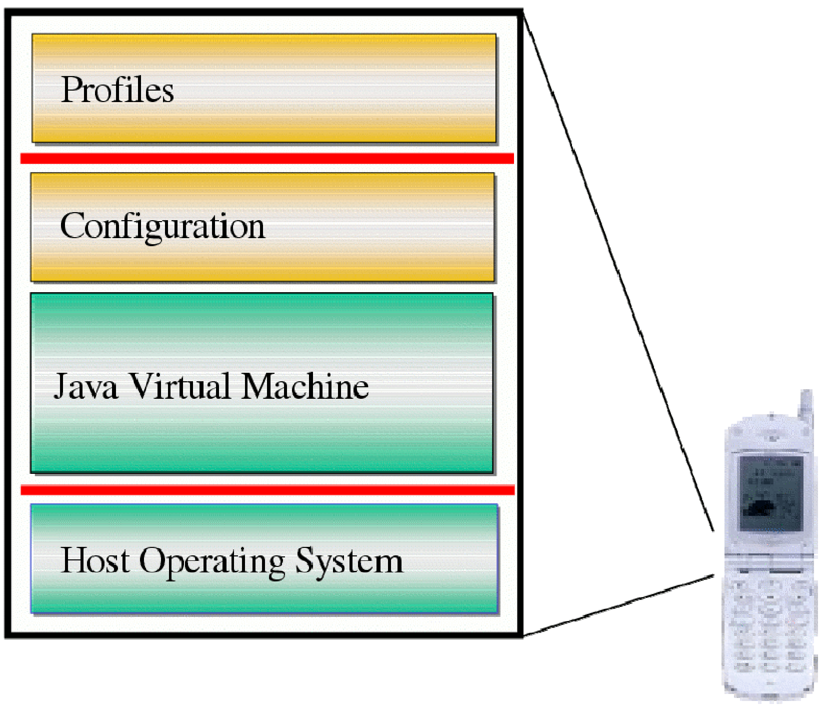


Figure 3 - Software architecture

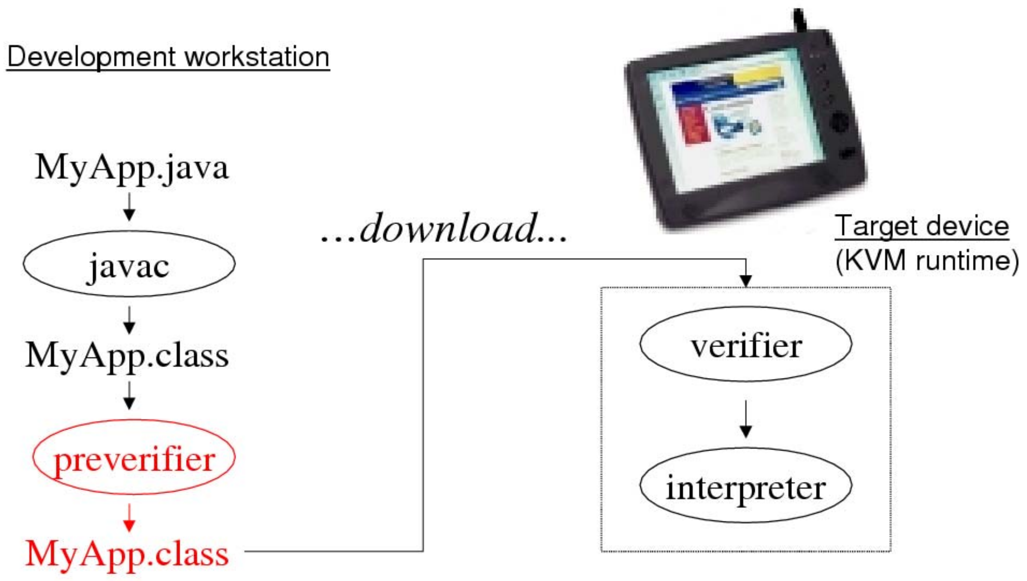


Figure 4 - Java Deployment

Just like an applet a J2ME-program consists of a class, which inherits from a special base-class. In the case of J2ME this base-class is MIDlet. In this class there are three methods in interest specified. They are called:

- public void startApp()

- public void pauseApp()
- public void destroyApp(boolean)

Now, to code a MIDlet-program it's necessary to overwrite this three methods in the derived class. A very simple MIDlet-program is this example, which prints the current date and time on the mobile's display:

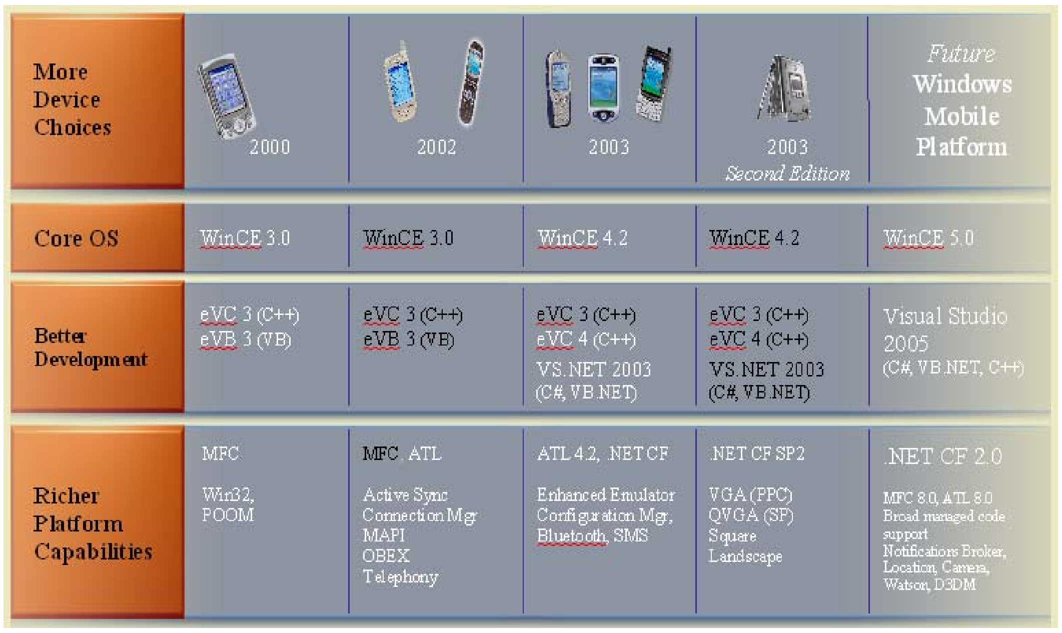
```
import java.util.Date;
import javax.microedition.lcdui.Alert;
import javax.microedition.lcdui.Display;
import javax.microedition.lcdui.Displayable;
import javax.microedition.midlet.MIDlet;

public class HelloJ2ME extends MIDlet {

 Alert timeAlert;

 public HelloJ2ME () {

       timeAlert = new Alert("Time");
       timeAlert.setString(new Date().toString());
 }

 public void startApp() {

       Display.getDisplay(this).setCurrent(timeAlert);
 }

 public void pauseApp() {}

 public void destroyApp(boolean unconditional) {}
}
```

Code 1 – Java-Example

3.3 Mobile Development with .NET Compact Framework[7]

The other of the two major software-platforms is the .NET Compact Framework (.NET CF) from Microsoft. This framework is only available on the WindowsCE operation system. The following figure shows Microsoft's roadmap of devices and their software:

[7] [MS05] and Microsoft Inc.



More Device Choices	2000	2002	2003	2003 *Second Edition*	*Future* Windows Mobile Platform
Core OS	WinCE 3.0	WinCE 3.0	WinCE 4.2	WinCE 4.2	WinCE 5.0
Better Development	eVC 3 (C++) eVB 3 (VB)	eVC 3 (C++) eVB 3 (VB)	eVC 3 (C++) eVC 4 (C++) VS.NET 2003 (C#, VB.NET)	eVC 3 (C++) eVC 4 (C++) VS.NET 2003 (C#, VB.NET)	Visual Studio 2005 (C#, VB.NET, C++)
Richer Platform Capabilities	MFC Win32, POOM	MFC, ATL Active Sync Connection Mgr MAPI OBEX Telephony	ATL 4.2, .NET CF Enhanced Emulator Configuration Mgr, Bluetooth, SMS	.NET CF SP2 VGA (PPC) QVGA (SP) Square Landscape	.NET CF 2.0 MFC 8.0, ATL 8.0 Broad managed code support Notifications Broker, Location, Camera, Watson, D3DM

Figure 5 - R oadmap

It starts with the first generation in 2000 and the version 3.0 of WindowsCE. At this time eVC (embedded Visual C++) and eVB (embedded Visual Basic) where the only available development plattforms. A little bit of alleviation was the included MFC (Microsoft Foundation Classes) which helped application developers to cope with the time consuming Win32-API programming. Two years later new APIs where introduced to control more hardware-features by software but the development plattforms were still the same. A new approach was VS.NET 2003 which not only introduced a new programming language (C#) but also a complete new programming environment. The .NET-technology consists like the Java-technology of a runtime and a rich set of APIs. It is high-level object oriented and promisses portability of code due the specification of IL (Immediate Language), an abstract code-set which can easily transformed to a specific real existing cpu-code whenever it's necessary. This has the advantage that code-interpretation, like in Java, isn't necessary anymore. .NET CF will become the most important development environment for the WindowsCE plattform. No new versions of the embedded Visual C++/Basic aren't announced for the fith-edition of Microsoft's embedded operating system.

.NET CF – programs where developed with the desktop version of the .NET technology on a development computer. Mobile developers have support with the Smart Device Extensions, which are part of the Visual Studio .NET 2003 software-package and extend the normal .NET desktop runtime. The produced code can uploaded to a WindowsCE-device and executed within the .NET CF Runtime. The .NET CF supports newer technologies like Webservices.

Figure 6 shows the software architecture of a WindowsCE device and where you can find the .NET CF within this architecture. It's above the APIs and substitudes MFC- and native Win32-programs.[8]

[8] URL: http://msdn.microsoft.com/library/default.aps?url=/library'en/us/dnanchor/html/NETCompactFrame.asp [08.03.2005]

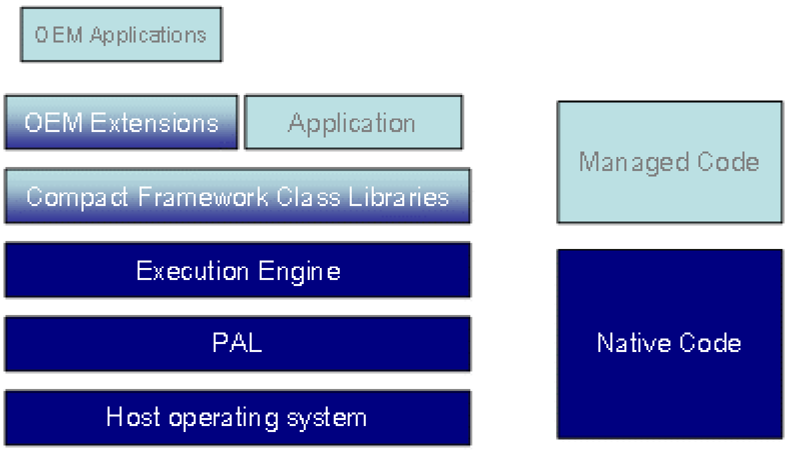


Figure 6 - .NET Compact Framework Class Architecture

The most important standard APIs shows Figure 7. Among them is System which is the general library for all .NET applications. In this dll you can find classes for collections, IO, reflection, threading and so on. Another basic library is System.Drawing which provides the programer basic 2D drawing support. System.Forms manages the graphical user-interface (GUI) of the WindowsCE-platform.

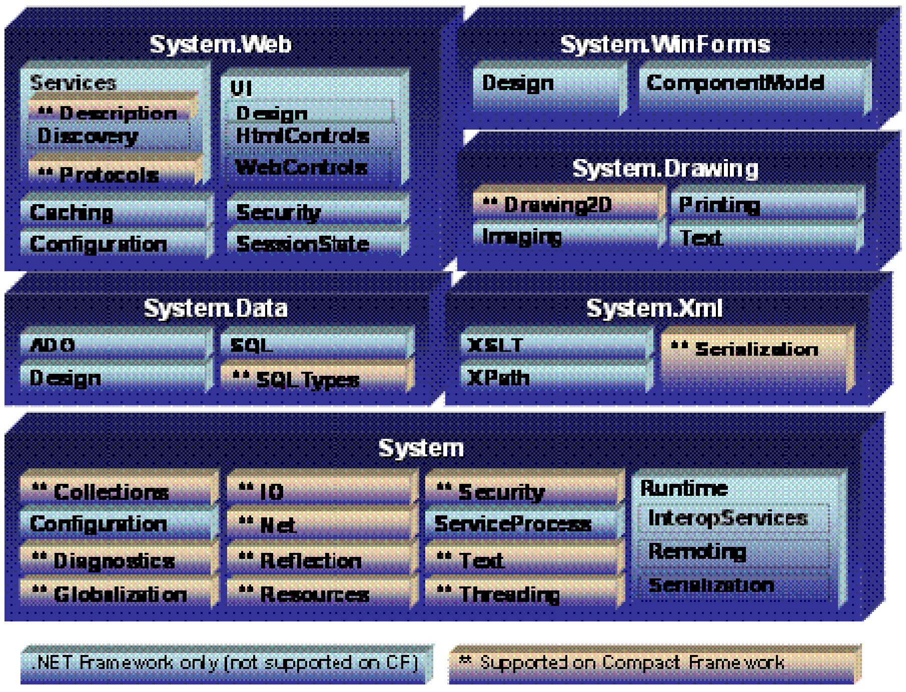


Figure 7 - .NET Libraries

In fall 2005 .NET CF will be upgraded to version 2.0. This new edition of that mobile framework promisses full support of interoperability and protocols. It will be compatible with .NET CF 1.0 and it will have enhanced performence. Some aspects of this performance enhancement are Unified JIT (Just in Time Compiling), improvements in string handling (ASCII and Unicode), improvements in XML support and the introduction of ADO.NET and SQL Mobile. Two libraries which should help to push Webservices. Another extension took place in the field of available gui-controls. .NET CF 2.0 supports more of these controls than its predecessor.

Deployment of .NET CF applications can be done with ActiveSync. This software provides the following tasks ([He03]):

- Connectivity for program installations. All the applications for the PocketPC are installed by running an installer on the host desktop.
- Filebrowsing on a PocketPC-device. In general mode the „My Documents Folder“ and in administrative mode the whole filesystem, including external storing cards.
- Backup- and Restore- functionality.
- Data/File Synchronization
    - Calender, contacts, tasks and Outlook-Inbox
    - Internet Explorer Favorites
    - Notes and Files
- Network/Internet connectivity for the docked PocketPC via the host computer.

A .NET CF – program looks just like normal .NET programs. No special classes are needed to code a program for the .NET CF as it is in J2ME. The following code shows a simple HelloWorld-application:

```
using System;
using System.Drawing;
using System.Collections;
using System.Windows.Forms;
using System.Data;

public class Form1 : System.Windows.Forms.Form
{

  private System.Windows.Forms.Button button1;
  private System.Windows.Forms.MainMenu mainMenu1;

  public Form1()
  {
   InitializeComponent();
  }

  protected override void Dispose( bool disposing )
  {
   base.Dispose( disposing );
  }
  private void InitializeComponent()
  {
   mainMenu1= new System.Windows.Forms.MainMenu();
   button1 = new System.Windows.Forms.Button();

   button1.Location = new System.Drawing.Point(72, 32);
   button1.Text = "press me";
   button1.Click += new System.EventHandler(button1_Click);

   Controls.Add(this.button1);
   Menu = this.mainMenu1;
   Text = "Form1";
  }

  static void Main()
  {
   Application.Run(new Form1());
  }

  private void button1_Click(object sender, System.EventArgs
e)
  {
   MessageBox.Show("Hello WindowsCE!!!");
  }
}
```

Code 2 - .NET Example

3.4 J2ME vs. .NET CF 1.0

These two software platforms have several similar aspects and help software-developers to enhance their productivity by providing a similar set of APIs and tools. They both abstract the underlying differences in hardware and enable portability of code. Now what are the differences between these platforms?

The .NET CF is powerful and resource expensive whereas the J2ME CLDC Configuration has fewer hardware requirements and hence is cheaper. It also is more pervasive than the .NET CF. The J2ME CLC Configuration has similar to the .NET CF in hardware requirements and power. J2ME CLDC focus both the consumer and the enterprise market whereas J2ME CDC and :NET CF addresses only the enterprise market. .NET CF supports more programming languages. By the time C# and Visual Basic .net. J2ME has only Java support in both configurations. Microsoft's runtime supports only the PocketPC and WindowsCE – platforms. J2ME CDC is supported by all major mobile platforms except PalmOS and J2ME CLDC is supported by all mobile platforms. Another aspect is the code compatibility. .NET CF is compatible through IL with the general .NET CLR (Common Language Runtime) whereas J2ME CDC is compatible to the J2SE. Instead of this J2ME CLDC has its own format and own virtual machine and it's therefore not compatible with J2SE and its CDC implementation. They differ also by the integrated set of APIs. :NET CF is a subset of its desktop variant and J2ME is a subset of J2SE plus optional packages. J2ME CLDC is partial compatible with the API-set of J2ME CLC and has also optional standard packages. Native APIs are not useable by the J2ME CLDC applications but due JNI (Java Native Interface) for J2ME CDC applications. .NET CF can use OS-APIs with the help of P/Invoke (Platform Invokation). Software is developed with Visual Studio .NET for the :NET CF whereas CodeWarrior supports the development of J2ME CDC as well as the command-line tools provided by SUN Microsystems. J2ME CLDC software can be programmed with all major Java-IDEs and of course the command-line tools. On both sides there are limitations in the security model. Only the J2ME CDC Configuration supplies the full Java Security Manager. The client installation of software can be done with ActiveSync or download via Internet Explorer on the .NET side and on Java with Sync or simple downloading. J2ME CLDC has a special OTA[9] specification for installation.[10]

Finally you can compare the specification process of that two platforms. :NET CF is specified by only one company whereas J2ME is specified by a community.

4 Design Considerations

Mobile software is different to its desktop pendant. You have to make some considerations about the right design with references to display constraints, connection problems and context.

4.1 Display constraints

There are several problems application developers have to cope with. For example tiny or small displays and low resolutions (e.g. 240x320 pixels).

Furthermore there are two different GUI-technics available on mobile devices:

- Single Window (PalmOS, PocketPC)
- Multiple Windows (WindowsCE)

For J2ME there's some additional support for programmers: J2ME Polish. This additional API consists of new routines which has a simpler GUI-Manager based on CSS-files (..). So design-changes can simply done by changing the CSS-files which are well established by website-programing.

Furthermore it has a device database in XML-Format as well as additional tools like a precompiler. This precompiler helps to minimize code-redundancy by statement like this:

```
//#if polish.midp2 || polish.api.mmapi
      // ok the audio-playback of the MMAPI can be used
      Player player = ...
//#endif
```

[9]OTA – Over The Air
[10] [GK] slides 46f

This device-dependend compiling is based on the device database provided by J2ME Polish.[11]

4.2 Connection problems

Another type of problems concerns connection. In the beginning different types of wireless network types were introduced. [GK] gives us a closer look of these technologies:

In the field of LAN (Local Area Network) the WLAN (also known as IEEE 802.11) has the greatest range of all. Its radius is about 300m and has a bandwith of 2/11/54 Mbit/s and uses frequencies like 2,4 GHz or 5 GHz. Bluetooth instead has a bandwith of 1Mbit and a radius of 10m using frequency 2,4 GHz. Infrared has a bandwith of 4Mbit and a radius of 10m.

WAN is the other category. The bandwiths in this field are 9.6kbps with GSM, 115kbps with GPRS and 2Mbps with the UMTS-technoloy.

The problems are:

- Heterogenous network technologies.
- There's no compatibility between these network types.
- Not always is the proper network connection reachable
- Low signal quality.

Following are suggestions to overcome these problems:

- Processing in offline mode.
- Buffering of data before transmitting
- If possible, reducing amount of data (e.g. Audio, Video) at low signal quality

4.4 Context

If we want to provide context based information we have to supply Context Middleware, which is provided by several sources and offers.

Some of these sources are the mobile user itself by providing data via the user-interface, other clients, software agents, sensors or even applications.

Context Middleware offers context representation and validation with the help of Context Templates and context storage and retrieval.

A Context Template is a pattern for context representation in a Context Domain. Whereas a Context Domain describes an environment.

A Context is now an instance of a Context Template within a Context Domain which now can be valid or invalid.

5 Application scenario: Dürer Weg

[11] URL: http://www.j2mepolish.org/ [15.03.2005]

Albrecht Dürer (May 21st, 1471 - April 6th, 1528) was a gifted german artist who was born in Nuremberg where he also had lived and worked until he died. He's one of the Nuremberg's most famous persons. In memory of him and his work the local cultural department has started a project for a multimedia tour through the ancient parts of the city. The tourists are guided by a small handheld computer. And all informations are given to them in form of digital content like audio and video.[12]

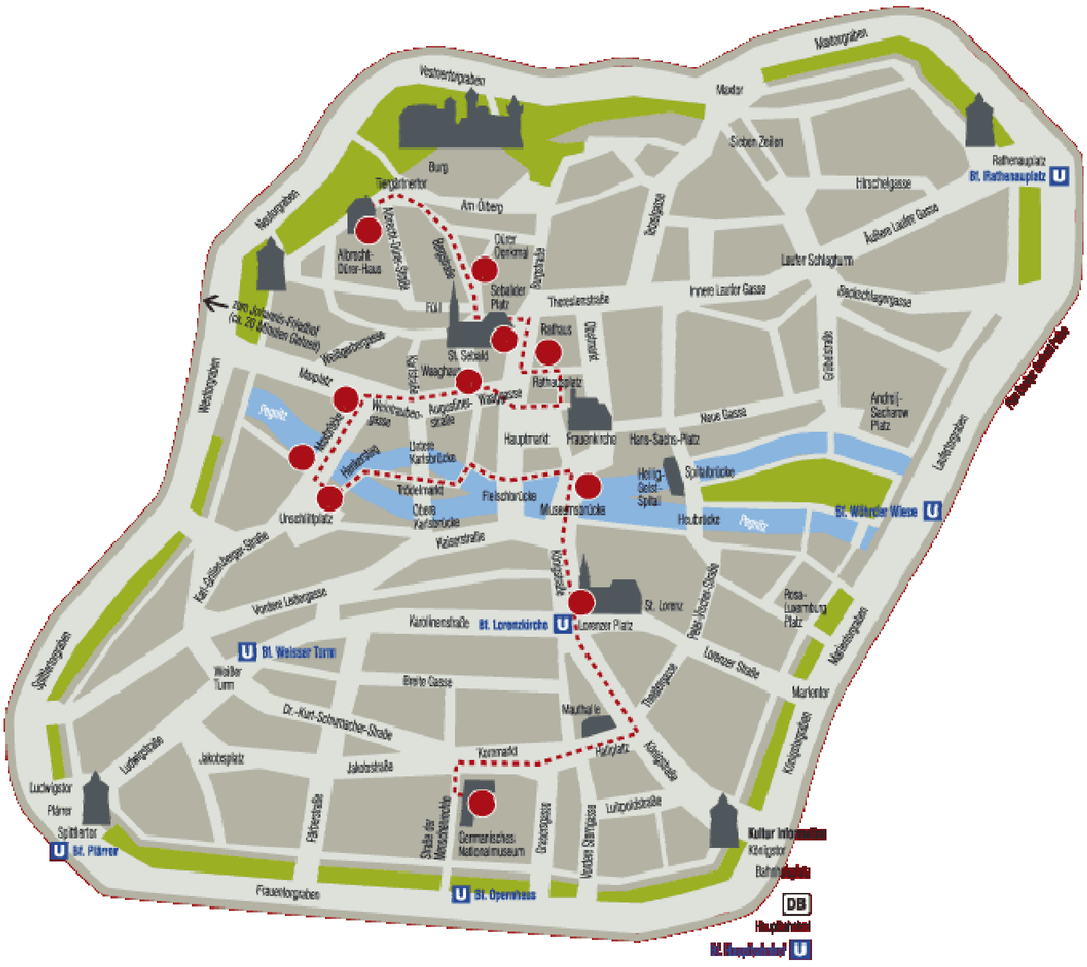


Figure 8 - A guided tour through ancient Nuremberg

Figure 8 shows a city map. There's only a route tourists can choose, but they can start whereever they want by deciding where to go next. This tour leads to eleven historical buildings, places and museums which are related to Albrecht Dürer as a person or his work. The dotted red lines show the tour and every red spot a place of interest.

Places of interest are e.g. The Germanische Nationalmuseum, Lorenz Platz with its huge cathedral or the Albrecht Dürer – House where he had lived.

If we compare this project to the tourist's vision in the beginning we see following points are already fullfilled:

- Sightseeing per guided walking tour

[12] URL: http://www.kubiss.de/kulturreferat/duerer/duererweg.htm [15.03.2005]

- Guide is a electronic mobile device
- Tourist can change a route from the map on the display
- Indications of items of interest on the virtual map
- Short videos describing the unique historical and architectural features of the current site is seen

What still lacks are the following:

- Passing a shopping district, translated advertisments of sales pop up on the display
- Bought goods are paid electronically and delivered to the correct gate for the flight home

In an abstract manner you can say that the context-based information is missing.

References

[GK] Gruber, F; Kurschel, W.: Building Mobile Applications. Comparing the Java and .NET approach. Uni Linz. URL: http://www.uni-linz.ac.at [08.03.2005]

[MK04] Mikalsen, M.; Kofod-Petersen, A.: Representing and Reasoning about Context in a Mobile Environment. Ulm. 2004. URL: http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-114/paper3.pdf [08.03.2005]

[MS05] Microsoft: Device Development with Visual Studio 2005 and future Windows Mobile Platforms. Microsoft. 2005.

[Pr05] Prengel, F: Neue Mobilitaet - .NET Framework 2.0 & SQL Server 2005 Mobile Edition. Microsoft. 2005.

[Sun00] Sun Microsystems : J2ME Building Blocks for Mobile Devices. Sun Microsystems. 2000.

[Sa96] Satyanarayanan, M.: Mobile Information Access. Varnegie Mellon University. Pittsburgh. URL: http://citeseer.ist.psu.edu/satyanarayanan96mobile.html [08.03.2005]