

Symbian Application Development White Paper

Sonera MediaLab

www.medialab.sonera.fi
info@medialab.sonera.fi

January 27, 2003

Symbian Application Development

1 INTRODUCTION

A couple of years ago most mobile phone manufacturers used their own operating systems in their products. Mobile phones were quite simple and did not require complex operating systems or software development platforms. Nowadays many new mobile phones are able to run downloaded applications, not just the ones preinstalled on the device by the manufacturer. The new phones are more and more complex and require an advanced operating system to provide a reliable and versatile platform for third party software.

To create such an operating system, some leading companies in the wireless industry established a private independent company, Symbian, in 1998 [1]. The company is owned by Ericsson, Nokia, Matsushita (Panasonic), Motorola, Psion, Siemens, and Sony Ericsson. Symbian is nowadays a supplier of an advanced, open, standard operating system for data enabled phones called Symbian OS [2]. Symbian OS is based on Psion's EPOC operating system.

Sun Microsystems' Java2 Micro Edition (J2ME) with Mobile Information Device Profile (MIDP) is currently the most used platform for developing hardware and operating system independent applications for modern mobile phones [3]. However, J2ME MIDP has many limitations that make some types of applications difficult or even impossible to implement. These applications usually need access to the device's hardware, operating system or other built-in software or data, or require maximum performance. Examples of these applications include games, some communications applications and applications that make use of special hardware such as a built-in digital camera.

The solution to this problem is to develop native applications for the device's operating system. Symbian OS is a widely supported and open operating system, which makes it a good choice for a platform for native applications. As the Symbian installation system (SIS) supports protocol independent over-the-air (OTA) installation, the delivery of native applications is generally not a problem, although the mobile network used can present some limitations in downloading very large installation files. Native applications running on Symbian OS have much better access to device hardware and software than MIDP applications. Performance is also significantly better in many cases. In Fig. 1, a typical software architecture diagram of a mobile device is presented. As can be seen, Java and native applications differ in that native applications have direct access to the device's operating system and system software, while Java applications have only limited access through several Java APIs.

There are also other platforms and operating systems designed for mobile devices, such as Palm OS [4] and Microsoft Pocket PC [5]. These platforms are used mostly on personal digital assistants (PDA) and similar devices, but not on mobile phones. Microsoft offers two operating systems for mobile phones, Pocket PC Phone Edition and Smartphone OS. The first products that use them entered the market in 2002.

This paper covers application development for Symbian OS and the Nokia Series 60 software platform [6] in particular. Other operating systems and Java technology are not within the scope of this document.

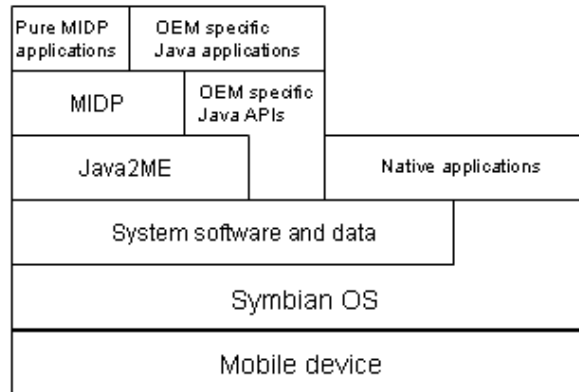


Figure 1. Typical mobile device software architecture.

2 SYMBIAN OS TECHNOLOGY

In this section we will first have a look at the requirements for mobile phone operating systems and then focus on Symbian OS technology.

2.1 Requirements for mobile phone operating systems

Symbian OS is an operating system designed especially for data enabled mobile phones. According to Symbian Ltd., the operating system was needed because scaling down PC operating systems or expanding existing light-weight operating systems would have led to too many fundamental compromises. Symbian OS is designed to fulfill some special requirements of mobile phones:

- devices are small and mobile
- the target is a consumer mass-market
- devices can be used when connected to the wireless network, locally to other devices or when not connected to any network
- manufacturers must be able to use it on very different kinds of products on different hardware designs, user interfaces and networks
- it must be open for third-party development of additional applications and services
- handling user data must be reliable even in case of unreliable communication, lack of resources like memory, storage or battery power
- all device resources, especially power consumption and memory, must be used efficiently
- wide support for industry standards.

2.2 Symbian OS devices and platforms

Symbian OS is designed to be flexible and can be used on devices with different user interfaces, hardware designs and network interfaces. The hardware design and communications capabilities of the device are mostly hidden behind the operating system, but the different user interfaces can be easily noticed. [7]

Version 6 of Symbian OS is divided into three categories: Quartz, Crystal and Pearl, depending on the type of the device and its user interface. These categories are called Device Family Reference Designs (DFRD) and they specify some guidelines for device hardware, software design and user interface. Quartz is used on pen operated PDAs, Crystal on communicators with a QWERTY keyboard, and Pearl on phones with a numeric keypad. These separate reference families do not exist anymore in Symbian OS version 7, which gives device manufacturers a possibility to make their own choices regarding design and user interface.

Nokia's Series 80 devices, such as the Nokia Communicator, are good examples of devices based on the Symbian OS 6 Crystal family [8]. These devices feature a full QWERTY keyboard and a large screen (640x200) that makes it possible to create more complex user interfaces. This type of user interface is suitable for editing and viewing data on the mobile device.

The most familiar mobile phone user interface is the conventional phone design with a numeric keypad, a relatively small screen, some soft keys, and possibly a joystick or jog dial. For example Nokia uses its Series 60 platform to create this kind of user interface on the Nokia 7650, 3650 and N-Gage phones [6]. Nokia's current Series 60 devices are based on Symbian OS 6.1 Pearl.

An interesting solution is to implement the user interface using a touch screen. Pen operated devices are well suited for browsing the Internet and using graphical applications, and services like multimedia messages. Sony Ericsson uses a touch screen user interface called UIQ in its P800 mobile phone [9]. P800 is the first device based on Symbian OS 7 and therefore it does not implement any DFRD.

Mobile phones that use Symbian OS are at the time writing this white paper [10]:

- Nokia Communicator 9210, 9210i, 9210c, 9290
- Nokia 7650
- Sony Ericsson P800
- Nokia 3650 (early 2003)
- Nokia N-Gage (spring 2003)

2.2.1 Nokia Series 60 Platform

The Nokia Series 60 Platform is a software platform for phones with Symbian OS and advanced data capabilities [6]. Series 60 includes features like user interface, telephony and personal information management applications, support for downloading new content based on Symbian, Java and XHTML technologies, support for messaging via SMS, MMS and e-mail, support for connections via GSM, GPRS, IrDA and Bluetooth, and synchronizing via SyncML. The Series 60 Platform is available for external OEM licensing and is currently licensed by Nokia, Siemens, Matsushita, Samsung and Sendo [6]. Software developers can create downloadable Java or C++ applications that are available for any Series 60 based mobile phone.

2.3 Application delivery

Symbian OS has its own application installation system and a special software management engine is included in the OS. This gives the user the possibility to view and uninstall already installed components. Installation packages (.sis files) are used for application delivery. Software developer kits include tools for application developers to create the installation package files consisting of the application components and data. There are several ways to install a SIS package on a device.

2.3.1 Installing from PC

Applications can be installed on a device from a PC using a serial cable, infrared or Bluetooth connection. This requires special installation software on the PC which is usually provided by the device manufacturer and delivered with the device.

2.3.2 Installing over the air

Applications can also be installed over –the air (OTA) using a mobile network [11]. There are a couple of ways to implement OTA installation. The installation package can be linked for example to a WAP page that can be viewed with the device's browser. The user can then use the

link to start the download. Downloading can be done protocol independently, but in most cases either the WAP or HTTP protocol is used. After the download has been completed, the device's software management engine completes the installation. WAP push can also be used for transferring the installation package to the device. It can be activated for example by an SMS message or from a web page.

2.4 Interoperability

Symbian supports most significant industry standards concerning mobile communications [12]. Technologies like SMS, MMS, Bluetooth, TCP/IP, GSM, GPRS, WCDMA, Java, WAP, SyncML, and more are supported, and the interfaces for using these technologies are available to application developers. This makes it easier and faster to develop applications that communicate with other applications, devices or systems.

3 DEVELOPMENT TOOLS

Because Symbian is a relatively new operating system, there are not many choices for development platforms or tools. Basically, manufacturers provide only a software developer's kit (SDK) with a set of tools, emulators, APIs, libraries and documentation for their own products. These SDKs usually either require or recommend the use of a common integrated development environment (IDE). Unfortunately, not all of the functionality of the IDEs is available for developers. For example, graphical user interfaces must often be coded manually, instead of using the graphical development tools available in the PC world, which makes application development a great deal slower. However, the situation is getting better, as there have been signs of co-operation between Symbian Ltd, mobile device manufacturers and development tools vendors. In Fig. 2, a generic development environment architecture is illustrated and some of the currently available development SDKs, tools, IDEs, and APIs are described in the following sections.

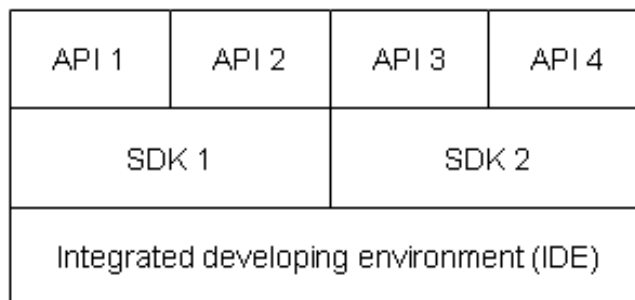


Figure 2. A development environment architecture.

3.1 Nokia Communicator Series SDK for Symbian OS

The Nokia Communicator Series SDK for Symbian OS is based on the Crystal v6.0 SDK. The SDK is compatible with the Nokia 9210, 9210i, and 9290 Communicators. The SDK requires the Microsoft Windows operating system and Microsoft Visual C++ 6.0 [13]. The SDK is available from Forum Nokia [14].

3.2 Nokia Series 60 SDK for Symbian OS

The Nokia Series 60 SDK is for the new Nokia 7650, 3650 and other upcoming Series 60 models based on Symbian OS 6.1 [15]. The SDK includes several APIs, examples, tools, and documentation for developing native applications for Series 60 devices. The tools include utilities for creating installation packages and a Series 60 emulator for PCs to make testing easier. The SDK's current version, 0.9, has many undocumented features and APIs. The Nokia

Series 60 SDK requires the Microsoft Windows operating system and Microsoft Visual C++ 6.0 [13]. The SDK is available from Forum Nokia which is a good place to get additional information from Nokia and other third party developers [14].

3.3 Symbian OS 7 UIQ SDK for Sony Ericsson P800

The Symbian OS 7 UIQ SDK for Sony Ericsson P800 is meant for application development for Sony Ericsson P800/P802 phones [16]. The SDK contains library files, documentation, sample code, tools and utilities for building applications in C++ and for creating installation files. An UIQ emulator is also included for testing. CodeWarrior Development Studio for Symbian OS is the recommended IDE and it is required to run the included emulator [17]. The SDK is available from Ericsson Mobility World [18].

3.4 Microsoft Visual Studio (IDE)

Microsoft Visual Studio is one of the most common IDEs on the Windows platform [13]. It is originally designed for Windows application development, but can also be used to develop applications for other platforms. Microsoft Visual C++ 6.0 can be used with Nokia's SDKs for Symbian OS.

3.5 Borland C++ Mobile Edition (IDE)

Borland C++ Mobile Edition is an extension for Borland C++ Builder 6 [19]. It makes it possible to develop applications for Series 60 and Symbian OS platforms using Borland's IDE. Borland C++ Mobile Edition has been available from late November 2002 and it will be supported by Nokia's Series 60 SDK [15].

3.6 Metrowerks CodeWarrior Wireless Development Kit for Symbian OS (IDE)

Metrowerks CodeWarrior Development Tools for Symbian OS is a complete IDE that enables C++ application development for Series 60, UIQ and other Symbian OS user interface packages [17]. The Symbian OS 7 UIQ SDK for Sony Ericsson P800 can be integrated with this IDE [16].

3.7 Symbian OS API

The Symbian OS API offers many programming interfaces for developers to access different subsystems of the device. These APIs can be used on any device with a proper version of the Symbian OS operating system. There are lots of similarities between different versions of the Symbian OS APIs, which makes it possible to develop the same application for the different OS versions only with minor changes. The following Symbian OS 6.1 APIs are included in the Nokia Series 60 SDK: Application Engines, Application Framework, Application Services, Base, Bluetooth, Comms Infrastructure, Graphics, Infrared, Multi Media Server, Messaging, Networking, Serial Comms, Telephony, WAP Stack.

3.8 Other APIs

There are also additional APIs that are dependent on the software and hardware platform. The most important of these APIs are the user interface APIs. Nokia has own its APIs for the Series 60 devices and Symbian's UIQ API can be used on phones with the UIQ user interface, such as Sony Ericsson's P800. There are also APIs that enable the use of some special features of the device. For example, Nokia's Camera API for the 7650 and 3650 phones gives developers the possibility to create their own applications that use the digital camera component integrated in the phone.

4 APPLICATION DEVELOPMENT CONSIDERATIONS

The fact that Symbian OS is designed for small and mobile devices with limited capabilities introduces some restrictions that must be taken into account in application development.

4.1 Processor

Most current devices have fairly powerful RISC processors with a clock frequency in the range of 100–200 MHz, which is usually adequate for most applications. However, the developer should keep in mind the fact that processors do not usually have any kind of floating point unit (FPU). This makes floating point math extremely slow and using integer or fixed-point math instead is preferred whenever possible.

4.2 Memory

Limited memory resources probably cause the most significant restrictions for mobile devices. There are usually three types of memory in the devices: read-only memory for the OS and system software, slow storage read-write memory for user applications and data, and fast execution read-write memory for running applications. The amount of memory for each of them is usually only a few megabytes (for example, the Nokia 7650 has 3.6 MB for system, 4 MB for storing user data and applications, and 4 MB of execution memory). Some new devices like the Sony Ericsson P800 and the Nokia 3650 also allow the storage memory to be extended with external memory cards. Because of the limited memory, the application should be as small as possible and use as little memory as possible at runtime. If the application has to write to the storage memory, the amount of data should be as low as possible. File formats that support compression are preferred. The developer must be careful when using dynamic memory allocation to prevent memory leaks which can rapidly lead to a lack of memory resources. This can have serious consequences such as device lock-ups, crashes, and loss of user data.

4.3 Data Communication Bandwidth

When developing communications applications, some restrictions of the current GSM and GPRS technologies must be taken into account. Throughput is an important factor when an application transfers a lot of data, such as multimedia content. Response time plays a significant role in applications like streaming, chatting, or multiplayer games. The delay is usually much longer (a few hundreds of milliseconds), and reliability is also poorer on mobile networks than on traditional fixed line networks. The connection duration is also an issue: circuit switched data (CSD) and high-speed circuit switched data (HSCSD) connections require a GSM data call, but GPRS connections can be used anytime without a data call when GPRS service is available and there is no GSM (voice or data) call open. Many Symbian phones use Bluetooth technology for short-range wireless applications [20]. Bluetooth can be used for faster and more reliable communication between two mobile phones, between a mobile phone and a PC, or between a mobile phone and a Bluetooth access point.

4.4 Security

Security is an important issue as native applications basically have unlimited access to the device. A native application can for example directly access the device's network interface, storage memory, phone interface, messaging, etc. This opens many possibilities for the application developer, but these possibilities can also be used improperly. Unfortunately, there is no easy way to determine if some particular software is secure or not. It is possible to use digital signatures and certificates to provide the user with some confidence that the application being installed is from a reliable provider. However, certifications and signatures do not make the application secure to use, they only ensure that the application really is from the provider it claims to be. Despite the facts above, standard, secure data communications technologies are

well supported in Symbian OS. These technologies give developers tools to create secure communications applications.

5 TEST APPLICATION

To test the ease of Symbian application development, we created a small program that utilizes interactive real-time graphics and consists of three different parts. The program demonstrates the possibilities and performance of native applications on the Nokia 7650 (See Fig. 3). Many game applications use very similar techniques to this test application and therefore it gives a good idea of the Nokia 7650's performance as a gaming device.

The application was developed using Microsoft Visual C++ 6.0 and Nokia Series 60 SDK for Symbian OS v0.9. The project was first created using Series 60 AppWizard v1.9 in Visual Studio and the application was tested on the emulator supplied with Series 60 SDK, as well as on a Nokia 7650.

The graphics algorithms used in the test application had already been implemented on the Windows and Pocket PC platforms, and porting them to the Symbian platform was an easy task. Creating a suitable application framework for a real-time, interactive graphics application was the most challenging task. Incomplete documentation caused many problems in the development and some issues had to be discussed in Forum Nokia to get the answers. In general, application development was quite slow. That was partially because the application was created to get some general experience on coding for the Symbian platform and therefore many new techniques had to be learnt not all of which were even used in the application.



Figure 3. Test application screen shots.

6 SUMMARY AND CONCLUSIONS

As explained in this white paper, developing native applications for mobile phones with Symbian OS is the only solution when there is a need to access device's hardware or software or when maximum performance is needed. Native code is always faster to execute than code run on a virtual machine like Java and there are less restrictions. Faster execution time is very important with real-time applications such as games and other multimedia applications. Our test application for the Nokia 7650 proves that it is possible to create fast, full-screen, real-time

graphics on mobile devices when the application is developed directly for the device's hardware and OS. There are also some disadvantages compared to application development with technologies like Java. In theory, J2ME MIDP applications can be run on any device with J2ME MIDP support, but native applications must be compiled separately for every device with a different platform. However, in many cases also MIDP applications need to be compiled separately. It is very hard to develop an application that fits all different J2ME MIDP devices, because of the different screen sizes, UIs, keyboard layouts, etc.

It is true that developing native Symbian OS applications is somewhat more difficult than developing J2ME MIDP applications, for example. One significant reason is that there are currently not many good development tools, as many manufacturers' own SDKs, development tools, and documentation are still under construction. This causes difficulties and slowness in development. Fortunately, this will change in the future because of the co-operation between the device manufacturers and development tools vendors. Also, some technical issues might make application development for Symbian OS feel difficult at first. In general, "the Symbian way" of doing things is somewhat different than Java or standard C++ development. For example, in Symbian programming, there is no Java-like garbage collection and error handling is totally different compared to standard C++. Fortunately, there are already many developers who share their knowledge on forums like Forum Nokia.

Symbian OS devices, including the first devices using Symbian OS 7, are steadily coming to the market. Hopefully, this will also bring new development tools to ease the development process and bring about better applications for mobile phones. It will also be interesting to see what kind of role Microsoft's Windows CE based Smartphone OS will play on the mobile OS market and how easy application development for it is. The first device, the Orange SPV, is already available in the UK and France.

The future of the Nokia Series 60 platform is also an interesting question. There are already some licensees for it, but so far only Nokia itself has released a product using the Series 60 platform.

REFERENCES

- [1] Symbian Ltd, <http://www.symbian.com/> [Accessed December 3, 2002]
- [2] Symbian OS, <http://www.symbian.com/about/symb-os.html> [Accessed December 3, 2002]
- [3] J2ME Mobile Information Device Profile (MIDP), <http://wireless.java.sun.com/midp/> [Accessed December 3, 2002]
- [4] Palm OS, <http://www.palmsource.com/palmos/> [Accessed December 3, 2002]
- [5] Microsoft Pocket PC, <http://www.microsoft.com/mobile/pocketpc/default.asp> [Accessed December 3, 2002]
- [6] Nokia Series 60 Platform, <http://www.nokia.com/nokia/0,5184,2816,00.html> [Accessed December 3, 2002]
- [7] Symbian OS User Interfaces, <http://www.symbian.com/technology/devices.html> [Accessed December 3, 2002]
- [8] Nokia Series 80, http://www.forum.nokia.com/main/1,35452,1_0_55_40,00.html [Accessed January 9 2003]
- [9] Sony Ericsson, <http://www.sonyericsson.com> [Accessed December 3, 2002]

- [10] Symbian OS phones, <http://www.symbian.com/technology/symbos-phones.html>
[Accessed December 3, 2002]
- [11] OMA Download Version 1.0,
http://www.openmobilealliance.org/omacopyrightNEW.asp?doc=OMA-DL-V1_0-20021104-d.zip [Accessed December 12, 2002]
- [12] Symbian standards, <http://www.symbian.com/technology/standards.html>
[Accessed December 3, 2002]
- [13] Microsoft Visual Studio, <http://msdn.microsoft.com/vstudio/> [Accessed December 9, 2002]
- [14] Forum Nokia – Symbian, http://www.forum.nokia.com/main/1,35452,1_32,00.html
[Accessed December 9, 2002]
- [15] Forum Nokia – Series 60 Platform,
http://www.forum.nokia.com/main/1,35452,1_36,00.html [Accessed December 9, 2002]
- [16] Symbian OS 7 UIQ SDK for Sony Ericsson P800,
http://www.ericsson.com/mobilityworld/sub/open/technologies/epoc/tools/7_uiq
[Accessed December 9, 2002]
- [17] CodeWarrior Wireless Development Kit for Symbian OS,
<http://www.metrowerks.com/MW/Develop/Wireless/Symbian/SymbianWirelessDevKit.htm> [Accessed December 9, 2002]
- [18] Ericsson Mobility World, <http://www.ericsson.com/mobilityworld/> [Accessed December 9, 2002]
- [19] Borland C++ Mobile Edition, <http://www.borland.com/cbuilder/mobile/> [Accessed December 9, 2002]
- [20] The Official Bluetooth Website, <http://www.bluetooth.com/> [Accessed January 7, 2003]

ADDITIONAL RESOURCES

Open Mobile Alliance, <http://www.openmobilealliance.org/> [Accessed December 9, 2002]

The Source for Java Technology, <http://java.sun.com> [Accessed December 9, 2002]

DEFINITIONS, ACRONYMS AND ABBREVIATIONS

API	Application Programming Interface
CSD	Circuit Switched Data
DFRD	Device Family Reference Design
EPOC	Former name for Symbian OS
FPU	Floating Point Unit
GPRS	General Packet Radio Service
GSM	Global System for Mobile communications
HSCSD	High-Speed Circuit Switched Data
HTTP	Hyper Text Transfer Protocol
IDE	Integrated Development Environment

IR	Infrared
IrDA	Infrared Data Association
J2ME	Java 2 Micro Edition
MIDP	Mobile Information Device Profile
MMS	Multimedia Message Service
OEM	Original Equipment Manufacturer
OS	Operating System
OTA	Over-The-Air
PC	Personal Computer
PDA	Personal Digital Assistant
QWERTY	Standard key layout used on computer keyboards
SDK	Software Developers Kit
SIS	Symbian Installation System
SMS	Short Message Service
SyncML	Open standard for data synchronization and device management
TCP/IP	Transmission Control Protocol/Internet Protocol
UI	User Interface
UIQ	User interface for Symbian OS (formerly known as Quartz)
VM	Virtual Machine
WAP	Wireless Application Protocol
WCDMA	Wideband Code Division Multiple Access
XHTML	Extensible Hypertext Markup Language