
S60 Platform: FAQ

Version 1.6

December 23, 2005

S60 p l a t f o r m

Legal Notice

Copyright © 2005 Nokia Corporation. All rights reserved.

Nokia, Nokia Connecting People, Nokia 6600, Nokia 7650, Nokia 7710, and Nokia 9200 are trademarks or registered trademarks of Nokia Corporation. Java and all Java-based marks are trademarks or registered trademarks of Sun Microsystems, Inc. Other product and company names mentioned herein may be trademarks or trade names of their respective owners.

Disclaimer

The information in this document is provided “as is,” with no warranties whatsoever, including any warranty of merchantability, fitness for any particular purpose, or any warranty otherwise arising out of any proposal, specification, or sample. Furthermore, information provided in this document is preliminary, and may be changed substantially prior to final release. This document is provided for informational purposes only.

Nokia Corporation disclaims all liability, including liability for infringement of any proprietary rights, relating to implementation of information presented in this document. Nokia Corporation does not warrant or represent that such use will not infringe such rights.

Nokia Corporation retains the right to make changes to this specification at any time, without notice.

License

A license is hereby granted to download and print a copy of this specification for personal use only. No other license to any other intellectual property rights is granted herein.

Contents

1.	Key Concepts	7
1.1	What is the S60 platform?.....	7
1.2	Briefly, what general features does the S60 platform provide?.....	7
1.3	What do the different “editions” of the S60 platform represent?.....	7
1.4	What are feature packs?	7
1.5	What does the S60 platform mean for developers?	8
1.6	How is the S60 platform deployed?	8
1.7	What are the different editions and feature packs of the S60 platform?	8
1.8	What consumer markets are targeted by S60 devices?.....	8
1.9	What user interfaces are available on S60 devices?	8
2.	S60 Platform Architecture	9
2.1	What is the architecture of the S60 platform?	9
2.2	What does <i>Symbian OS</i> refer to?	9
2.3	What does <i>base software</i> refer to?	9
2.4	What does <i>lead software</i> represent?	10
2.5	What does <i>applications</i> represent?.....	10
2.6	Does the S60 platform provide a user interface?.....	10
3.	Features of the S60 Platform.....	11
3.1	What is the underlying operating system for each edition and feature pack release?	11
3.2	What is included in the base software for S60 1 st Edition?	11
3.3	What main features are available in S60 1 st Edition, Feature Pack 1?	11
3.4	What is included in the base software for S60 2 nd Edition?	11
3.5	What main features are available in each feature pack for S60 2 nd Edition?	12
3.6	What is included in the base software for S60 3 rd Edition?.....	13
3.7	What main features are available in each feature pack for S60 3 rd Edition?.....	13
4.	S60 3rd Edition	14
4.1	What are the key concepts and issues for S60 3 rd Edition?.....	14
4.2	What are the costs and benefits of these new features?	14
4.3	Does the binary compatibility break mean higher porting costs?	15
4.4	What features are included in platform security?	15
4.5	What are the capability sets?	16
4.6	What trust does my application need to access the capabilities?	16
4.7	What are the different signing requirements for an S60 3 rd Edition application?.....	17
4.8	Will all applications need to be signed?	18
4.9	Will all applications need to be Symbian Signed?.....	18

4.10	If an application does not require Symbian Signed to work, are there any other reasons why it should be Symbian Signed?	18
4.11	Can protected capabilities be accessed during testing?	18
4.12	What is involved in obtaining Symbian Signed?	18
4.13	Are there any tools to assist with the Symbian Signed process?	19
4.14	Obtaining Symbian Signed status has a cost, so how are freeware applications going to get to market?	19
5.	Development Technologies Supported by the S60 Platform	20
5.1	What development technologies are available?.....	20
5.2	C++ Application Development.....	20
5.2.1	Is standard C or C++ used for application development?.....	20
5.2.2	What types of applications can I produce using the available C++ APIs?	20
5.2.3	Will my C++ application be compatible with other S60 devices?.....	20
5.2.4	Is there any information on porting existing applications to the S60 platform?....	20
5.2.5	How are applications installed on an S60 device?	21
5.3	Java™ Development.....	21
5.3.1	What version of Java™ MIDP does the S60 platform support?	21
5.3.2	What Java™ APIs does S60 1 st Edition support?	21
5.3.3	What Java™ APIs does S60 2 nd Edition support?.....	21
5.3.4	What Java™ APIs does S60 3 rd Edition support?	21
5.3.5	Does platform security affect Java™ development?	22
5.3.6	Will my Java™ application be compatible with other S60 devices?	22
5.3.7	Will my Java™ application be compatible with devices based on other platforms?.....	22
5.3.8	Will my Java™ application be compatible with other MIDP devices?.....	22
5.3.9	How are applications downloaded and installed on an S60 device?	22
5.4	Content Development	23
5.4.1	What browsing standards does the S60 platform support?	23
5.4.2	What messaging standards does the S60 platform support?	23
5.4.3	Will browser applications for S60 3 rd Edition run on devices based on earlier editions?.....	23
5.4.4	Will MMS content and applications for S60 3 rd Edition work on the earlier editions of the platform?	23
5.4.5	What is OMA DRM?	24
5.4.6	What is OMA Client Provisioning?.....	24
5.4.7	What enhancements to DRM are implemented on S60 3 rd Edition?	24
6.	Development Tools for the S60 Platform	25
6.1	What IDE options are available for C++ developers?	25
6.2	What tools are available for C++ development?	25
6.3	What Java™ development tools are available for the S60 platform?.....	26

6.4	What tools are available for content developers?	26
6.5	Are all of Nokia's tools available to all developers?	26
7.	Developing for the S60 Platform	27
7.1	Why target the S60 platform for initial development of mobile applications?	27
7.2	What issues are associated with optimizing applications across the range of S60 devices?	27
7.3	What issues are associated with migrating applications across other Nokia platforms?	28
7.4	What issues are associated with porting applications across other devices based on Symbian OS?	28
7.5	What issues are associated with porting applications to devices based on unrelated operating systems across the mobile market?	28
8.	Known Issues	29
8.1	What are the known issues that S60 application developers should be aware of?	29
9.	Business Case	30
9.1	What is the business case for using the S60 platform?	30
10.	Getting Applications to Market	31
10.1	How can Nokia help get my application to market?	31

Change History

October 29, 2003	Version 1.0	Initial document release
June 28, 2004	Version 1.1	Document updates throughout
April 25, 2005	Version 1.2	Updated for S60 3 rd Edition
October 10, 2005	Version 1.3	Updated for S60 3 rd Edition, Feature Pack 1
November 9, 2005	Version 1.4	Updated Chapter 4 with additional details on signing requirements Updated Chapter 5 with information about new Carbide.c++ tool
December 16, 2005	Version 1.5	Updated DRM details in Chapter 5
December 23, 2005	Version 1.6	Updated JSR support details in Chapter 3 and 5

1. Key Concepts

1.1 What is the S60 platform?

The S60 platform is the market-leading smartphone platform built on the Symbian operating system (Symbian OS). It incorporates all key mobile technologies expected by increasingly sophisticated consumers and provides revenue opportunities for the full range of stakeholders in the mobile marketplace. As the S60 platform has developed, it has raised the bar on smartphone feature provision by taking the lead in developing and implementing many innovations.

1.2 Briefly, what general features does the S60 platform provide?

Devices based on the S60 platform feature:

- Advanced telephony.
- At minimum, a 176 x 208-pixel color screen.
- Innovative form design and keypad layout.
- Personal information manager (PIM) applications (such as Contacts list and Calendar).
- Messaging.
- Internet browsing.
- Digital camera.

Several S60 devices also feature:

- Music player.
- Media gallery.
- Video recorder.
- Sound recorder.
- FM radio.

1.3 What do the different “editions” of the S60 platform represent?

The S60 platform is continually developing and adding new features. Stakeholders must be assured that they are buying into the biggest and best platform. A new edition of the S60 platform represents a collection of significant feature updates and additions, often accompanied by an edition change of the underlying Symbian OS.

1.4 What are feature packs?

Feature packs are important releases of additional features that become available between releases of new editions. For example, S60 2nd Edition had three feature packs, all of which brought enhanced capabilities to the edition. The ongoing release of feature packs allows the S60 platform to develop continually, while maintaining a common baseline across the editions.

1.5 What does the S60 platform mean for developers?

Stakeholders are continually looking to maximize their total addressable markets at the lowest possible cost. The development of a new application or content for a target device always introduces a trade-off between the costs of moving the product to new devices and the benefits of having a bigger target audience. The S60 platform allows stakeholders to have the best of both worlds. It provides for software implementation across all devices based on a particular edition of the platform. Developers can be certain that applications created using only platform-edition features will be easily transferable to any device based on that edition.

1.6 How is the S60 platform deployed?

The S60 platform is licensed by an increasing number of stakeholders that have developed devices for the market place. These include Lenovo, LG, Nokia Mobile Phones, Panasonic, Samsung, and Siemens. The range of S60 devices is constantly changing, and detailed information on the current range of devices can be found in the Products section of the S60 Web site (www.s60.com/products).

1.7 What are the different editions and feature packs of the S60 platform?

The main editions and feature packs available for the S60 platform are:

- S60 1st Edition.
 - S60 1st Edition, Feature Pack 1.
- S60 2nd Edition.
 - S60 2nd Edition, Feature Pack 1.
 - S60 2nd Edition, Feature Pack 2.
 - S60 2nd Edition, Feature Pack 3.
- S60 3rd Edition.
 - S60 3rd Edition, Feature Pack 1.

1.8 What consumer markets are targeted by S60 devices?

The S60 platform is designed to serve the widest consumer and enterprise audience through mass-market devices, also allowing for extremely focused consumer-device segmentation in areas such as enterprise, entertainment, personal productivity, and games.

1.9 What user interfaces are available on S60 devices?

Because the S60 platform does not mandate particular user interface characteristics, screen resolutions, or input methods as part of its specification, it allows manufacturers to produce highly visible device customizations. It does have a user interface style and APIs as part of its specification; therefore, it is expected that the common user interface style will be adhered to in any user interface implementation. This is reinforced by the Avkon UI libraries on which all user interface implementations are based.

2. S60 Platform Architecture

This section examines the underlying architecture of the S60 platform.

2.1 What is the architecture of the S60 platform?

The architecture of the S60 platform is shown in Figure 1.

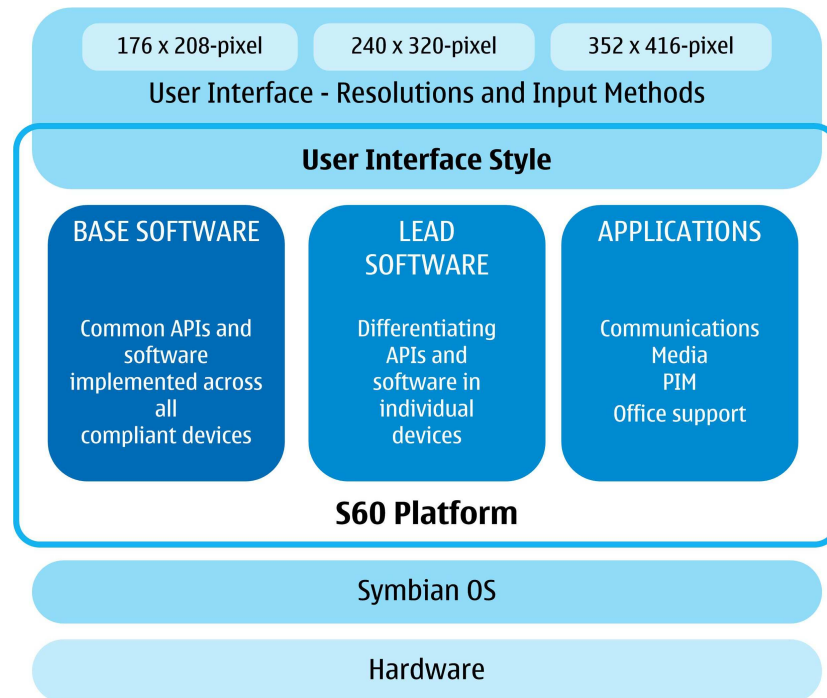


Figure 1: The S60 platform architecture is depicted.

The constituent parts of the S60 platform encompass the base software, lead software, applications, and UI style built on top of Symbian OS. These terms are discussed in the following sections.

2.2 What does *Symbian OS* refer to?

The underlying operating system for the S60 platform is Symbian OS. Before starting development, developers should refer to the Forum Nokia Web site (www.forum.nokia.com) to check feature details of the Symbian OS version that is incorporated in each edition or feature pack.

2.3 What does *base software* refer to?

The term *base software* is used to describe the collection of software components that provide the technology support that is guaranteed for all devices based on a particular edition of the S60 platform. It is therefore possible, with minimal changes, to port an application developed using only base-software components to the range of devices that are based on the same edition of the S60 platform.

2.4 What does *lead software* represent?

Lead software refers to technological innovations that are implemented for specific devices. Some elements of lead software may become part of future base software for the S60 platform in a subsequent feature pack or edition. Developers should not assume that any piece of lead software is on all devices based on a platform edition; they must be aware of potential restrictions to portability when using these features.

2.5 What does *applications* represent?

The term *applications* refers to a device's consumer capabilities, including personal information manager (PIM), messaging, media applications, and profiles.

2.6 Does the S60 platform provide a user interface?

The S60 platform defines a user interface style and APIs, but it does not mandate the screen size or input methods. Licensees are free to implement their own customized user interfaces. Developers must program user interface applications with scalability in mind because specific user-interface dimensions cannot be assumed.

3. Features of the S60 Platform

This section examines the underlying operating system, base software, and lead software found in each version of the S60 platform.

3.1 What is the underlying operating system for each edition and feature pack release?

The following releases and their respective operating systems are:

- S60 1st Edition — Symbian OS v6.1.
- S60 2nd Edition — Symbian OS v7.0s.
 - S60 2nd Edition, Feature Pack 1 — Symbian OS v7.0s.
 - S60 2nd Edition, Feature Pack 2 — Symbian OS v8.0a.
 - S60 2nd Edition, Feature Pack 3 — Symbian OS v8.1a.
- S60 3rd Edition — Symbian OS v9.1.
 - S60 3rd Edition, Feature Pack 1 — Symbian OS v9.1.

3.2 What is included in the base software for S60 1st Edition?

S60 1st Edition includes the following base software:

- Java™ 2 Platform, Micro Edition (J2ME™) APIs, including:
 - Mobile Information Device Profile (MIDP) 1.0.
 - Connected Limited Device Configuration (CLDC) 1.0.
 - Wireless Messaging API (JSR-120).
 - Mobile Media API (JSR-135).
- XHTML/WML browsing.
- Multimedia Messaging Service (MMS) messaging.

3.3 What main features are available in S60 1st Edition, Feature Pack 1?

The introduction of Java™ APIs for Bluetooth Java Specification Request (JSR-82) to S60 1st Edition, Feature Pack 1 is a notable addition.

3.4 What is included in the base software for S60 2nd Edition?

S60 2nd Edition includes the following base software:

- Skinning (theme) and Digital Rights Management (DRM) C++ APIs.
- Java™ 2 Platform, Micro Edition (J2ME™), Mobile Information Device Profile (MIDP) 2.0, including enhanced performance through the inclusion of the Connected Limited Device Configuration (CLDC) HotSpot compiler.

- XHTML browsing over TCP/IP.
- Multimedia Messaging Service (MMS) support for HTTP transport and advanced presentation capabilities through enhanced Synchronized Multimedia Integration Language (SMIL) support.
- Open Mobile Alliance (OMA) Client Provisioning, which allows device settings for services such as browsing, MMS, and over-the-air (OTA) Calendar synchronization. This technology enables easy device configuration and ensures that a consumer can take full advantage of a device's capabilities.
- DRM via OMA forward-lock.

3.5 What main features are available in each feature pack for S60 2nd Edition?

The following features are the main items provided by the feature packs for S60 2nd Edition. For more details, please see www.forum.nokia.com.

- Feature Pack 1
 - Presence Open and Bluetooth notifier C++ APIs.
 - Wireless Messaging API 1.1 (JSR-120) and Mobile Media API 1.1 (JSR-135) Java™ APIs.
 - Support for a megapixel camera with 4x zoom as well as recording and playback of video clips.
- Feature Pack 2
 - C++ APIs for browser plug-in, connection monitor server, Digital Rights Management (DRM) license manager, simulation file, DRM helper, pictograph, DevASR, speech recognition utility, MMF SRS custom commands, Data Synchronization Profile Listing, content access framework, GIF scaler, Huffman encoding and decoding, message queue, Publish & Subscribe, Location Acquisition, OpenGL ES, EGL, and Symbian XML framework.
 - Mobile 3D Graphics API for J2ME™ (JSR-184), FileConnection API (JSR-75), and PIM API (JSR-75) Java APIs.
 - Support for a 1.3-megapixel camera with 6x zoom, WCDMA and Enhanced Data Rates for Global Evolution (EDGE) networks, and IPv6.
- Feature Pack 3
 - APIs for feature discovery, scalable icons, browser control, download manager user-interface library, download manager engine, image transform library, image transform plug-ins, Universal Serial Bus (USB) client driver, and an updated utilities API.
 - J2ME™ Web Services Specification (JSR-172) and Java APIs for Bluetooth (JSR-82), with OBEX support Java APIs.
 - Support for scalable user interfaces (176 x 208-pixel, 240 x 320-pixel, and 352 x 416-pixel screens).

3.6 What is included in the base software for S60 3rd Edition?

In addition to the features delivered in S60 2nd Edition, the following APIs are included in the base software for S60 3rd Edition:

- C++ APIs, including:
 - Location APIs (including Landmark API, Landmark UI APIs, Landmark Search API, and BLID application API).
 - Web Services APIs (including WS Connection API, WS Description API, WS Manager API, and XML Extensions API).
 - Session Initiation Protocol (SIP) APIs.
 - Open Mobile Alliance (OMA) Digital Rights Management (DRM) v2 API.
 - OMA Datasynch 1.2 API.
 - Multimedia Framework (MMF) DRM API.
 - Back-light control API.
 - Instant messaging (IM) API.
 - IM Application Launch API.
 - Bluetooth 1.2 support.
 - EXIF API.
 - Find Item API.
 - Accessory API.
- Java™ APIs, including:
 - Security and Trust Services API (JSR-177).
 - Java Location API (JSR-179).
 - SIP API (JSR-180).
 - Wireless Messaging API (WMA) 2.0 (JSR-205).

3.7 What main features are available in each feature pack for S60 3rd Edition?

The following features are the main items provided by the feature packs for S60 3rd Edition. For more details, please see www.forum.nokia.com:

- Feature Pack 1 provides a Web browser based on open source components, device management extensions, Open Mobile Alliance (OMA) standards-based Push-to-Talk capability, and a flexible XML-based user-interface customization framework.

4. S60 3rd Edition

This section examines the various characteristics of S60 3rd Edition, such as binary compatibility breaks, platform security, and the application certification process.

4.1 What are the key concepts and issues for S60 3rd Edition?

There are several important ideas and issues that stakeholders must be aware of with respect to S60 3rd Edition:

- Platform security.
- Data caging.
- Trusted computing base.
- Developer self-signing of applications not submitted for Symbian Signed.
- Symbian Signed certification.
- Capabilities.
- Binary compatibility break.
- Real-time kernel.
- New compiler and build tools.
- Increased device differentiation and segmentation.
- Market volumes.

4.2 What are the costs and benefits of these new features?

Developers will incur various costs. For instance, new development tools will be required for S60 3rd Edition, although free tools will also be available (see question 6.1, “What IDE options are available for C++ developers?”). Further, developers will bear costs of certification, including:

- Obtaining a VeriSign Authenticated Content Signing (ACS) Publisher ID for use in Symbian Signed certification. Currently an ACS Publisher ID costs \$350 (290 euros) per annum.
- Symbian Signed certification testing. At the time of writing, testing rates are between 185 euros (\$215) and 560 euros (\$670) for first application submissions and between 160 euros (\$190) and 280 euros (\$335) for subsequent submissions of the same application. For content, low volumes cost 40 euros (\$48) per item and are reduced for bulk submissions. Current testing prices can be found on the Symbian Signed Web site (www.symbiansigned.com/app/page/testhouses)

The cost of the development tools can be spread across several projects, although certification incurs per-application charges. Note that it will still be possible to avoid these certification costs by creating applications that are not Symbian Signed (see question 4.8, “Will all applications need to be signed?”). In addition, Handango, Inc., also provides a Symbian Signed service that can be paid for from sales royalties, eliminating the need for an up-front payment.

The new features of S60 3rd Edition offer a number of benefits. For instance:

- The new compiler produces a smaller ROM footprint and more efficient code, enabling smaller and faster applications.
- The enhanced platform security reduces the likelihood of corruption or unauthorized disclosure of application data.
- Market confidence will receive a significant boost because the S60 platform will proactively deal with security issues and provide a mechanism for application verification. Cautious consumers will be reassured about applications from previously unknown vendors.
- Consumers can deny permission to unsigned applications if they are unsure about them.
- More APIs will be available to third-party developers through certification, thereby enabling enhanced customization of a wider range of platform functionality.
- Enhanced Digital Rights Management (DRM) capabilities will make software piracy less likely.
- Single-chip hardware solutions are available for S60 devices, thereby increasing the flexibility of hardware configuration and providing for cheaper devices.

4.3 Does the binary compatibility break mean higher porting costs?

The binary compatibility break does mean that applications will need to be rebuilt using different tool sets in order to be available to devices on either side of the break. However, with reasonable solution design, this cost need not be high. Most of the functionality will be unaffected, and most serious UI issues can be avoided through scalable design.

4.4 What features are included in platform security?

The platform security introduced in S60 3rd Edition has several important aims. It is intended to protect the integrity of the device and to limit access to sensitive data and operations. Consumers and enterprise users have greater protection from viruses, while operators, licensees, and third-party developers have greater brand and data protection. There are three key concepts involved in platform security.

- **Data caging.** A new directory system allows applications to store data securely with restricted access.
- **Trusted computing base.** This encompasses software authentication and authorization, secure storage and execution, boot protection, and trusted hardware services. In other words, it provides the mechanism for ensuring that untrusted applications are controlled and that the device is protected.
- **Capability model.** Capabilities are sets of features and functions within Symbian OS grouped according to how their use could affect a device. Each capability requires a level of trust with any application that tries to use it. The more sensitive the capability's operation, the more trust is required before an application can use it. The majority of APIs require no trust, but some require an application to be Symbian Signed before they can be accessed.

4.5 What are the capability sets?

There are four sets of capabilities:

- **Open.** These capabilities cover some 60 percent of Symbian OS APIs. The APIs in this set are sufficient to create stand-alone applications with access to the UI and the ability to store data.
- **Basic.** These capabilities cover most communications features, personal area networks, Internet access, messaging, and phone calls as well as the ability to access sensitive user data such as contacts and agenda entries. It also covers access to basic location information.
- **Extended.** These capabilities cover more-advanced device information and control features such as device settings, power management, event generation, and extended location information.
- **Manufacturer.** These capabilities provide fundamental access to the operating system and secured data.

4.6 What trust does my application need to access the capabilities?

The trusts required by an application to access capabilities are as follows:

- **Open.** These capabilities are open to all applications, and applications do not need to be Symbian Signed.
- **Basic.** These capabilities (with the exception of full location information) can be granted by the user to the application on installation, this subset is known as the *unsigned-sandbox* set. Symbian Signed applications can access all these capabilities without requiring the user to grant permission.
- **Extended.** These capabilities require an application to be Symbian Signed before they can be accessed. In addition, a subset of these capabilities require a declarative statement, explaining why the capabilities need to be accessed, to be provided with the Symbian Signed submission.
- **Manufacturer.** These capabilities require business support from the device manufacturer to allow the capabilities to be granted during the Symbian Signed process.

4.7 What are the different signing requirements for an S60 3rd Edition application?

The various signing requirements for S60 3rd Edition applications are shown in Table 1.

	Seeking Symbian Signed	Not Seeking Symbian Signed
On-device testing	Sign with Symbian Developer Certificate (if capabilities being used)	Sign with self-generated certificate — “self-sign”
Submit for Symbian Signed	Sign with Authenticated Content Signing (ACS) Publisher ID	Not required
Symbian Signed granted	Signed with a single-use ACS Publisher ID that is trusted through the Symbian Root Certificate (by test house or self-certifier)	Not required
Deploy to users		Sign with self-generated certificate — “self-sign”

Table 1: The various signing requirements for S60 3rd Edition applications are shown.

The various signing requirements are:

- Self-signing — Application and *.sis package content that are not being submitted for Symbian Signed must be self-signed by the developer. The signature is provided by a key pair (certificate) created by the developer using tools provided in the S60 3rd Edition SDK. Self-signing does not grant any capabilities to the application. The consumer can grant an application access to unsigned-sandbox basic capabilities on installation.
- Symbian Developer Certificate signing — An application that requires access to capabilities granted under Symbian Signed or uses unsigned-sandbox capabilities and is to be submitted for Symbian Signed (to bypass user granting of unsigned-sandbox capabilities) must be signed with a Symbian Developer Certificate to enable testing on a device during development. These certificates are available for free from the Symbian Signed Web site (www.symbiansigned.com).
- Authenticated Content Signing (ACS) Publisher ID signing — Applications and *.sis packaged content that are to be submitted for Symbian Signed testing must be signed by the developer using its ACS Publisher ID. This signing is to ensure that the test house can verify the source of the application or content.
- Symbian Signed certification signing — This signature, provided by a single-use ACS Publisher ID that is trusted through the Symbian Root Certificate installed on Symbian OS v9 devices, shows that an application or *.sis content package has passed Symbian Signed and unlocks the application’s declared capabilities. Applications and *.sis packaged content can be signed in this manner by a Symbian Signed test house or a developer who has achieved Self Certification status.

4.8 Will all applications need to be signed?

Yes. The application installer in S60 3rd Edition requires that all applications and *.sis packaged content be signed. This is done using a self-signing mechanism (using a self-generated certificate) or as a result of successful attainment of Symbian Signed.

4.9 Will all applications need to be Symbian Signed?

No. Whether an application requires signing depends on whether or not it accesses trusted capabilities. Some applications, such as games, may be installed and run without certification. Applications that use Basic capabilities, those to which the consumer can grant access, will not require certification. They would, however, benefit from certification, since it removes the need for the consumer to grant access to protected capabilities during installation. All applications accessing basic capabilities outside the unsigned-sandbox, extended, and manufacturer capabilities will need to be Symbian Signed.

4.10 If an application does not require Symbian Signed to work, are there any other reasons why it should be Symbian Signed?

Obtaining Symbian Signed status for an application means that the user no longer receives a warning message when attempting to install the application and is not prompted to grant Basic capabilities (if the application uses them). As a result, the installation process is more straightforward, and the user is more likely to install the application because no warning is issued.

More important, Symbian Signed is a requirement for accessing many sales channels; for example, all native Symbian OS applications delivered through Nokia sales channels must be Symbian Signed.

4.11 Can protected capabilities be accessed during testing?

Yes. The S60 SDK emulator allows the enforcing of capability access to be turned off, so applications can be tested even though they are not Symbian Signed. For device testing, a Symbian Developer Certificate is required. These certificates, which are available free of charge, allow an application to be signed for a specific device, thus allowing on-device debugging and acceptance testing to be performed on applications that do not have Symbian Signed status.

4.12 What is involved in obtaining Symbian Signed?

The Symbian Signed process is as follows:

1. Register on the Symbian Signed Web site (www.symbiansigned.com) using the online form and completing the contractual agreement.
2. Using the user name and password supplied, log on to the Symbian Signed account and download the tool for exporting a VeriSign Authenticated Content Signing (ACS) Publisher ID to a standard format for use with Makesis.
3. Sign the application's *.sis file (with downloaded tool and Makesis) and submit it to the chosen test house, along with the application's *.pkg file and user guide. A receipt will be provided.

4. Once the test house has verified the ACS Publisher ID, it will e-mail a quote for the testing.
5. Once payment has been made, the tests will take place.
6. Notification will be provided as to whether the application has passed or failed.
 - o If the application fails, it will need to be modified and submitted again at step 3.
 - o If the application passes, it is re-signed using a single-use ACS Publisher ID that is trusted through the Symbian Root Certificate embedded on S60 devices (Symbian B). The application is then uploaded to the developer's account, from which it can be collected for distribution.

4.13 Are there any tools to assist with the Symbian Signed process?

Yes. Symbian Ltd., in association with SysOpen Digia Plc, has made available a testing tool that allows applications to be verified against the Symbian Signed test criteria before they are submitted for signing. The beta version of the tool currently supports applications for platforms up to and including S60 2nd Edition, Feature Pack 3. For more details, see www.symbiansigned.com.

4.14 Obtaining Symbian Signed status has a cost, so how are freeware applications going to get to market?

Freeware applications are eligible for the Symbian Signed Freeware Route to Market program. This provides the opportunity for freeware to be Symbian Signed for free. The program covers applications that provide the user with full access to the application's features for free. It does not cover demonstration applications, shareware, or other types of licensing whereby the user must pay to use the full features of the application, or applications where the developer generates revenue by other means, such as adware. Honorware or donationware, where users are under no obligation to pay to access all the application's features but may do so if they wish, is covered by the program.

The program is available to free applications and places no restrictions on the developer's business model. Therefore, commercial developers creating freeware have the same eligibility to use the program as open source or hobbyist developers.

5. Development Technologies Supported by the S60 Platform

5.1 What development technologies are available?

C++ is the native programming language of the S60 platform. Developers can use C++ to create applications that access the application engines (including Photos and Phonebook) and that work with a multitude of technologies (such as Bluetooth connectivity, infrared, multimedia, messaging, networking, and telephony).

Applications can also be developed using Java™ MIDP 2.0, while content can be developed for XHTML browsing and Multimedia Messaging Service (MMS).

5.2 C++ Application Development

5.2.1 Is standard C or C++ used for application development?

The C++ language used in the S60 platform has been optimized for small devices with memory constraints. The Standard Template Library (STL) is not supported, and other changes to standard C++ have been made. Native Symbian OS applications are developed using Symbian Ltd.'s own version of C++, which utilizes most of the C++ idioms but has a unique nomenclature. ANSI C is supported but is rarely required.

5.2.2 What types of applications can I produce using the available C++ APIs?

Development of applications using C++ allows developers to realize the full potential of the platform and to produce a wide variety of applications. For example, public APIs allow developers to create applications that manipulate onboard cameras, construct and send Multimedia Messaging Service (MMS) messages, exchange data with other devices using Bluetooth technology, and access the telephony engine.

5.2.3 Will my C++ application be compatible with other S60 devices?

C++ applications written for a particular edition of the S60 platform can run on other devices that use the same edition, provided that the applications use no feature pack or lead software APIs. As previously noted, there is a binary compatibility break between S60 2nd Edition and S60 3rd Edition. Nokia is committed to minimizing compatibility issues between the different versions of the platform, but there are also some known issues between S60 1st Edition and S60 2nd Edition. More information about these issues can be found in Chapter 8, "Known Issues."

5.2.4 Is there any information on porting existing applications to the S60 platform?

Yes. The Porting section of the Forum Nokia Web site (www.forum.nokia.com/porting) contains a series of documents that describe the requirements and considerations involved in porting existing applications from platforms such as UIQ, Palm OS, and Pocket PC, as well as devices in the Nokia 9200 Communicator series.

5.2.5 How are applications installed on an S60 device?

There are several ways of downloading applications. Local PC transfers using Bluetooth connectivity or an infrared (IrDA) link are possible, and over-the-air (OTA) downloading with a mobile browser is another option. The downloaded applications are installed in the S60 device by an application installer.

5.3 Java™ Development

5.3.1 What version of Java™ MIDP does the S60 platform support?

S60 1st Edition supports the Java™ Platform, Micro Edition (Java ME), Connected Limited Device Configuration (CLDC) 1.0, and Mobile Information Device Profile (MIDP) 1.0.

S60 2nd Edition and S60 3rd Edition support MIDP 2.0 and CLDC 1.0 with the CLDC HotSpot virtual machine implementation. The S60 implementation of MIDP 2.0 includes support for secure networking.

5.3.2 What Java™ APIs does S60 1st Edition support?

The Java™ APIs supported by S60 1st Edition include:

- Wireless Messaging API (JSR-120), which supports the sending and receiving of Short Message Service (SMS) messages via GSM and CDMA.
- Mobile Media API (JSR-135), which enables applications to use sound, video, and animation playback, and video and audio recording.
- Nokia UI API, a proprietary Nokia Java API, which provides an interface for sound generation, low-level graphics, and access to the entire screen.

5.3.3 What Java™ APIs does S60 2nd Edition support?

The Java™ APIs supported by S60 2nd Edition include:

- All of the Java APIs supported by S60 1st Edition.
- The Wireless Messaging API (JSR-120), which has been enhanced to support Short Message Service (SMS) Push.
- Java APIs for Bluetooth (JSR-82) for exploiting Bluetooth connectivity.

5.3.4 What Java™ APIs does S60 3rd Edition support?

The Java™ APIs supported by S60 3rd Edition include:

- All of the Java APIs supported by S60 2nd Edition.
- Security and Trust Services API for J2ME™ (JSR-177).
- Location API for J2ME™ (JSR-179).
- SIP API for J2ME™ (JSR-180).
- Wireless Messaging API (WMA) 2.0 (JSR-205).
- Mobile Service Architecture for CLDC (JSR-248).

5.3.5 Does platform security affect Java™ development?

No, Java™ development is not affected by platform security. Only native Symbian C++ applications are affected because of their inherent low-level capabilities.

5.3.6 Will my Java™ application be compatible with other S60 devices?

Java™ applications should be compatible with devices that use the same edition of the S60 platform, provided that the applications do not make use of any other Java Specification Requests (JSRs) that may have been implemented as lead software. There are some variations among the editions of the S60 platform (as detailed in Section 5.3.1, “What version of Java™ MIDP does the S60 platform support?”) that may give rise to compatibility issues. Also, developers must always be aware of issues related to user interface scaling and input methods when migrating applications among S60 devices.

5.3.7 Will my Java™ application be compatible with devices based on other platforms?

The Series 40 Platform, the Series 80 Platform, and the Nokia 7710 widescreen smartphone implement the same Mobile Information Device Profile (MIDP) APIs and Java™ Specification Requests (JSRs) that are available on the S60 platform, so a high level of compatibility should be achieved. There will be the usual issues with user interface variations and differing memory conditions. For example, Series 40 devices can have smaller screen sizes and more limited memory than S60 devices.

5.3.8 Will my Java™ application be compatible with other MIDP devices?

The extent of compatibility with other devices that run applications that conform to the Mobile Information Device Profile (MIDP) will depend on the supported MIDP version and Java™ Specification Requests (JSRs) on the target device. Developers who wish to take full advantage of the Java support provided by the S60 platform may need to make changes to their application to achieve compatibility with other devices. Developers who wish to create applications that can run on the widest range of devices should consider limiting their use of APIs to those that are most commonly implemented.

Obviously, any use of the Nokia UI API will preclude application compatibility with devices based on other platforms.

5.3.9 How are applications downloaded and installed on an S60 device?

There are a number of ways to deploy a MIDlet to a device. The choice of method will depend mainly on the hardware available. The main deployment methods are: Bluetooth connectivity, infrared (IrDA) link, over the air (OTA) (downloading from a WAP or Web site), e-mail, Multimedia Messaging Service (MMS), or a serial cable.

5.4 Content Development

5.4.1 What browsing standards does the S60 platform support?

S60 1st Edition supports:

- WML and WMLScript.
- XHTML Mobile Profile.
- WAP and related services, such as WAP Push.
- WAP Cascading Style Sheets (WAP CSS).

S60 2nd Edition and S60 3rd Edition support:

- HTTP/1.1 Protocol over TCP/IP.
- XHTML over TCP/IP.
- Secure Sockets Layer (SSL) v3.0.
- Transport Layer Security (TLS) 1.0.

5.4.2 What messaging standards does the S60 platform support?

The S60 platform supports:

- Multimedia Messaging Service (MMS). MMS is based on the specifications produced by the WAP Forum (www.wapforum.org) and 3GPP (www.3gpp.org). MMS has a store-and-forward model similar to Short Message Service (SMS), but MMS greatly expands the earlier technology by handling content such as audio clips and pictures.
- S60 2nd Edition (and later) offers enhanced Synchronized Multimedia Integration Language (SMIL) support for MMS presentation.

5.4.3 Will browser applications for S60 3rd Edition run on devices based on earlier editions?

Yes. The browser available with each version supports both XHTML MP and WML. Nokia recommends use of XHTML MP for creating services.

5.4.4 Will MMS content and applications for S60 3rd Edition work on the earlier editions of the platform?

Multimedia Messaging Service (MMS) content should work with all S60 devices. S60 3rd Edition uses MMS Digital Rights Management (DRM). These features are not available for earlier editions. Also, because S60 1st Edition does not support Synchronized Multimedia Integration Language (SMIL), developers will need to ensure content is ordered correctly when it is sent to S60 1st Edition devices.

5.4.5 What is OMA DRM?

S60 2nd Edition supports the Open Mobile Alliance (OMA) forward-lock method of content protection. This means that certain types of files can be protected to prevent them being forwarded (via the full range of messaging and communications technologies) from a device. This protection applies to a variety of media file types and Java™ MIDlets. Content that has been protected using OMA forward-lock can be received only via HTTP download or Multimedia Messaging Service (MMS). The forward-lock feature is automatically activated when protected content is received.

Applications that handle protected objects must use the encryption services provided by the Digital Rights Management (DRM) Engine to store or temporarily save such objects. Most preinstalled terminal applications, such as the Voice Recorder and the Media Gallery, are designed to recognize objects protected by forward-lock.

5.4.6 What is OMA Client Provisioning?

Device settings can be configured via Open Mobile Alliance (OMA) Client Provisioning with a minimum of user interaction. For example, a device can receive a message containing the required setup for a WAP client and configure itself accordingly. Other settings that can be communicated this way include Multimedia Messaging Service (MMS), e-mail, instant messaging (IM), and access points.

The configuration information can be delivered by a variety of mechanisms, including over the air (OTA) or by means of media such as Subscriber Identity Module (SIM) cards.

5.4.7 What enhancements to DRM are implemented on S60 3rd Edition?

The following APIs and features have been provided on S60 3rd Edition with respect to Digital Rights Management (DRM):

- S60 Open Mobile Alliance (OMA) DRM v2 API migration.
- Multimedia Framework (MMF) DRM API.
- OMA DRM 2.0.
- E-mail synchronization (local and remote over the air [OTA]) based on OMA data synch 1.2, e-mail filtering, and time zone support for Calendar.
- Rich Push e-mail (IMAP/POP, OMA e-mail notification, OMA Data Synchronization (DS), polling e-mail).

6. Development Tools for the S60 Platform

6.1 What IDE options are available for C++ developers?

S60 C++ applications can be created using the CodeWarrior® Development Studio for Symbian OS, Borland C++ Mobile Edition, Microsoft Visual C++ 6.0, and Microsoft Visual Studio .NET 2003.

During 2006 Nokia will be replacing the CodeWarrior Development Studio for Symbian OS with a set of new tools based on Eclipse called Carbide.c++. Three versions will be available:

- Carbide.c++ Express. This version will be available free of charge and will contain all the tools necessary to create C++ applications with an S60 SDK, Series 80 SDK, or UIQ SDK.
- Carbide.c++ Developer Edition. This version will include a graphical design tool and provide the ability to perform on-device debugging. Supporting development for the S60 platform, Series 80 platform, and UIQ platform, it is expected to cost 299 euros (\$357).
- Carbide.c++ Professional. This version will provide mobile phone manufacturers with all the tools required to create Symbian OS devices.

For more information on Carbide.c++, see www.forum.nokia.com/carbide.

6.2 What tools are available for C++ development?

SDKs supporting the CodeWarrior® Development Studio for Symbian OS, Borland C++ Mobile Edition, Microsoft Visual C++ 6.0, and Microsoft Visual Studio .NET 2003 development environments for C++ development are available for download from the Tools and SDKs section of the Forum Nokia Web site (www.forum.nokia.com/tools). Each SDK includes the following:

- S60 APIs.
- An S60 emulator for testing.
- Sample applications.
- Documentation.

SDKs are currently available for editions up to S60 3rd Edition. An SDK for S60 3rd Edition, Feature Pack 1 is expected to be made available during the first half of 2006.

Note that use of Microsoft Visual Studio .NET 2003 requires the Nokia Developer's Suite for Symbian OS.

6.3 What Java™ development tools are available for the S60 platform?

The S60 Mobile Information Device Profile (MIDP) SDK is available for download from the Tools and SDKs section of the Forum Nokia Web site (www.forum.nokia.com/tools). It includes:

- An S60 emulator.
- Java™ libraries.
- APIs (including the Nokia UI API).
- Documentation.

The kit is compatible with professional Java development environments from IBM Corporation, Sun Microsystems, Inc., and Borland Software Corporation. More recent SDKs also support NetBeans and Eclipse integrated development environments (IDEs).

IDEs from Borland and Sun can be used in conjunction with the Nokia Developer's Suite for the Java™ 2 Platform, Micro Edition (J2ME™). The suite is also a prerequisite for using Eclipse. This tool accepts SDK plug-ins for each of the supported devices and includes the S60 MIDP SDK.

6.4 What tools are available for content developers?

The Nokia Mobile Internet Toolkit provides the tools that developers need to create and test browsing and messaging applications. The kit contains, among other tools, a reference implementation browser. For more details, go to the Tools and SDKs section of the Forum Nokia Web site (www.forum.nokia.com/tools).

Developers of messaging applications will benefit from the Nokia Developer's Suite for Multimedia Messaging Service (MMS), while those working on WAP/XHTML and Push applications should get the Nokia Mobile Internet Toolkit. Another resource is the Nokia Mobile Server Services API and Library for developers working with the Nokia Multimedia Messaging Service Center (MMSC) and server deployment projects. All of these tools are available free of charge from the Tools and SDKs section of the Forum Nokia Web site (www.forum.nokia.com/tools).

For multimedia projects, there is the Nokia Audio Suite, a tool set for developers of SP-MIDI (Scalable Polyphony) ring tones.

6.5 Are all of Nokia's tools available to all developers?

Nokia offers developers the opportunity to join Forum Nokia PRO. One of the benefits of the Forum Nokia PRO program is that members get early access to new and updated tools. Once the early-access period on Forum Nokia PRO has expired, all tools are made available through the Tools and SDKs section of the Forum Nokia Web site (www.forum.nokia.com/tools).

7. Developing for the S60 Platform

7.1 Why target the S60 platform for initial development of mobile applications?

The mobile device market has expanded dramatically in terms of design variation and technology adoption since the introduction of the first smartphones. The S60 platform has occupied a leading position within the marketplace since the introduction of the Nokia 7650 imaging phone in 2002 and has been followed by a profusion of new-device releases and announcements from a range of S60 licensees.

For two important reasons, the S60 platform is the ideal starting point for developing mobile applications. First, it is the leading platform within the smartphone market, with numerous devices available. There is, therefore, an established market for new applications with millions of consumers who are in the habit of regularly accessing sales channels.

Second, software developed for the S60 platform is highly portable. For example, it is relatively easy to migrate applications designed for the S60 platform to other Nokia platforms. There are also other platforms based on Symbian OS, such as UIQ, to which S60 applications can be ported with changes being required primarily for the user interface, and this is true for both native applications (written in C++) and MIDlets. There is literature available to explain how to port from the S60 platform to those platforms (as well as to Microsoft Pocket PC and Palm OS) in the Porting section of the Forum Nokia Web site (www.forum.nokia.com/porting).

7.2 What issues are associated with optimizing applications across the range of S60 devices?

The major sources of variation for devices based on the S60 platform are user interface, platform editions, feature packs, and lead software.

The S60 platform does not mandate a particular UI resolution, display orientation, or input method as part of its specification (although the UI style is part of the platform). Developers should therefore create applications with flexibility and scalability in mind and use features such as the feature discovery API wherever possible.

There are some known issues concerning binary compatibility between the first two editions of the S60 platform. (See Chapter 8, “Known Issues,” for details of the relevant literature.) Clearly, applications written for S60 3rd Edition may use features unavailable in S60 2nd Edition and S60 1st Edition. Most problems can be avoided if developers acquaint themselves with the technologies available on each edition and design to allow for feature differences.

Developers may wish to use features provided by lead software, which will not be available across the entire S60 device range. When using lead software, developers should check to ensure that those devices on which they wish to deploy have implemented these features — this will naturally limit the extent of the product’s total addressable market.

The introduction of S60 3rd Edition, with its binary compatibility break from the other editions, obviously creates an issue for developers who want to access both new and existing devices. However, carefully designed applications will simply need to be recompiled using a different tool chain.

7.3 What issues are associated with migrating applications across other Nokia platforms?

Nokia has produced two other platforms, the Series 40 Platform and the Series 80 Platform, and it has a device, the Nokia 7710 widescreen smartphone, that is based on Symbian OS. All of these are capable of running applications based on Java™ MIDP. All of them, except for the Series 40 Platform, are based on Symbian OS and are capable of running applications written in native C++. Note that features of the Nokia 7710 smartphone will be fully integrated into the S60 platform in 2006.

The major issues that will be encountered by developers who want to migrate applications across these platforms will be the differences in user interface styles and the underlying APIs, differences in memory constraints (especially with Series 40 devices), and differences in implemented technologies. (See the Platforms page at www.forum.nokia.com/platforms.) It should be noted also that on existing device implementations, there are differences in screen orientation between the landscape-style UI on Series 80 devices and the Nokia 7710 smartphone and the portrait-style UI on Series 40 devices. Early editions of the S60 platform also used a portrait-style UI, but from S60 3rd Edition onward, landscape support has been added for screens with certain resolutions. This may mean that a UI designed to use the width available on Series 80 devices and the Nokia 7710 smartphone may require significant redesign to successfully operate on S60 devices and Series 40 devices. If a high degree of UI consistency is required between different device versions of the same application, then the initial UI design stage will have to take these factors into account.

7.4 What issues are associated with porting applications across other devices based on Symbian OS?

All devices based on Symbian OS are capable of running applications written in native C++ and Java™ MIDP, so a certain amount of compatibility can be expected. The most common source of differentiation can be found in the user interface, but developers should investigate the range of differences in the capabilities and technologies as well.

7.5 What issues are associated with porting applications to devices based on unrelated operating systems across the mobile market?

The natural choice to ensure the relative ease of cross-platform compatibility is to develop applications in Java™ MIDP, but the choice will depend on the capabilities required by the specific application.

Applications written in Symbian OS C++ will probably need to be completely rewritten for platforms using other operating systems. However, because Symbian OS supports development in ANSI C, it is possible to write an application engine for the S60 platform that is independent of the operating system and then use that engine as the basis of applications developed for other operating systems. This method requires developers to understand the limitations of all their target platforms and to create an engine that accommodates all these limitations.

It should be noted that while tools are available for writing in Standard C++ on platforms such as Palm OS, this is not supported on devices based on Symbian OS.

8. Known Issues

8.1 What are the known issues that S60 application developers should be aware of?

Nokia is committed to maintaining compatibility among the different editions of the S60 platform and among the various platforms wherever possible.

Where compatibility is not achieved, Nokia will maintain and publish a record of any issues, such as minor differences in API implementations or problems with technology implementations. To this end, the following documents are available from the Documents section of the Forum Nokia Web site (www.forum.nokia.com/documents):

- *2nd Edition Platforms: Known Issues.*
- *Series 40 and Series 60 Developer Platform 1.0: Known Issues.*
- *Known Issues in the Nokia 6600 MIDP 2.0 Implementation.*
- *Series 60 Platform: Source and Binary Compatibility.*

Details of known issues and technical solutions (comprising mainly developer questions that have been answered by Forum Nokia Professional Support) are also available in the Nokia Technical Library on the Library page of the Forum Nokia Web site (www.forum.nokia.com/library). The Nokia Technical Library is available in Microsoft HTML Help, WebHelp, and PDF formats.

9. Business Case

9.1 What is the business case for using the S60 platform?

The S60 platform is the established leader in the smartphone market, making it an important source of revenue for third-party application and content developers. S60 3rd Edition significantly extends the business opportunities and introduces major enhancements designed to provide long-term stability for stakeholders. There are greater opportunities for market segmentation with a range of features and applications targeted at the enterprise, game, music, video, personal productivity, and other sectors.

The S60 platform gives the developer community access to industry-standard technologies and a market that can be measured in millions of consumers. The opportunities expand beyond S60 devices, since the platform provides standard technologies that allow developers to build applications and content for Nokia Series 40 devices, Series 80 devices, and the Nokia 7710 widescreen smartphone, as well as devices provided by other manufacturers.

The S60 platform is well supported by development tools and documentation provided on the Forum Nokia Web site (www.forum.nokia.com) and by a number of tools companies. Established sales channels help developers realize their investments as quickly and easily as possible.

There are three compelling reasons for developers to use the S60 platform as the basis for mobile applications and content.

First, users of devices that incorporate platforms are significant consumers of applications and content.

Second, Nokia alone has shipped more than 25 million S60 devices (as of May 2005) and expects the smartphone market to exceed 50 million units in 2005. Some 27 S60 devices have been launched, with S60 devices retailed by the majority of operators worldwide. With six licensees, the S60 platform represents a significant and growing market of potential customers.

Third, the platform approach allows developers to create content for one device or a range of devices that can, with minimal additional development work, be optimized to work with other devices that incorporate platforms. This allows developers to multiply the potential market for their application or content, with only a small incremental investment in optimizing or migrating their offering.

10. Getting Applications to Market

10.1 How can Nokia help get my application to market?

Nokia connects mobile application developers to the market in multiple ways.

Preminet Solution is a hosted content-distribution tool that connects mobile content providers, operators, content aggregators, and consumers. Preminet is open and customizable and brings global sales opportunities to content providers. Preminet makes it easy to deliver quality content and applications into the hands of mobile consumers around the world.

Nokia also offers developers lucrative opportunities to sell mobile content and Java Verified™ applications through Nokia sales channels: Nokia Software Market and Nokia MultiMediaCards (MMCs). In addition, some applications may be chosen to be embedded in new Nokia devices.

For full details, please follow the Marketplace link in the Developers section of the Nokia Web site (www.nokia.com/developers).