symbian

Symbian OS Version 7.0

Functional description

David Mery, Technology Editor, Symbian Ltd
Revision 1.5, February 2003

Summary

Symbian OS is the advanced, open operating system licensed by the world's leading mobile phone manufacturers. It is designed for the specific requirements of advanced 2G, 2.5G and 3G mobile phones. Symbian OS includes a robust multi-tasking kernel, integrated telephony support, communications protocols, data management, advanced graphics support, a low-level graphical user interface framework and a variety of application engines.
This paper gives a detailed overview of features and functionality available in Symbian OS Version 7.0.

1. Introduction

Symbian OS is the advanced, open, standard operating system licensed by the world's leading mobile phone manufacturers. Symbian OS is designed for the specific requirements of open, advanced, data-enabled 2G, 2.5G and 3G mobile phones. Compact enough to fit in the memory of a mobile phone, Symbian OS was planned from the beginning to be a full operating system in terms of functionality. Symbian OS is already available in the Ericsson R380 smartphones, the Nokia 9200 Communicator series, the Nokia 7650 and the Sony Ericsson P800. With the introduction of Symbian OS v7.0, the range of mobile phones with Symbian OS will expand even further. The first available Symbian OS v7.0 phone is the Sony Ericsson P800. Symbian OS is characterized by:

- **Integrated multimode mobile telephony** – Symbian OS integrates the power of computing with mobile telephony, bringing advanced data services to the mass market
- **Open application environment** – Symbian OS enables mobile phones to be a platform for deployment of applications and services (programs and content) developed in a wide range of languages and content formats
- **Open standards and interoperability** – With a flexible and modular implementation, Symbian OS provides a core set of application programming interfaces (APIs) and technologies that is shared by all Symbian OS phones. Key industry standards are supported
- **Multi-tasking** – Fully object-oriented and component-based, Symbian OS includes a multi-tasking kernel, middleware for communications, data management and graphics, the lower levels of the graphical user interface framework, and application engines
- **Flexible user interface design** – By enabling flexible graphical user interface design on Symbian OS, Symbian is fostering innovation and is able to offer choice for manufacturers, carriers, enterprises and end-users. Using the same core operating system in different designs also eases application porting for third party developers
- **Robustness** – Symbian OS maintains instant access to user data. It ensures the integrity of data, even in the presence of unreliable communication, and shortage of resources such as memory, storage and power

2. Key features

Symbian OS is the basis of the next generation of mobile phones. Symbian OS is the common core of application programming interfaces (APIs) and technology that is shared by all Symbian OS phones. Key features of Symbian OS Version 7.0:

- **Rich suite of application engines –** including contacts, schedule, messaging, browsing, office, utility and system control; OBEX to exchange objects such as appointments and business cards; integrated APIs for data management, text, clipboard and graphics.
- **Browsing** – fit for purpose browsing engine for full web browser support and WAP stack for mobile browsing
- **Messaging** – multimedia messaging using MMS, picture messaging with EMS and text messaging using SMS; Internet email using POP3, IMAP4, SMTP, MHTML; standard attachments; fax
- **Multimedia** – shared access to screen, keyboard, fonts and bitmaps; audio recording and playback, and image related functionality (support for all common audio and image formats), including API for graphics acceleration, streaming and direct screen access
- **Communication protocols** – wide-area networking stacks including TCP, IP version 4, IP version 6 and WAP, and personal area networking stacks including infrared (IrDA), Bluetooth, USB
- **Mobile telephony** – abstract API for cellular standards. GSM circuit-switched voice and data (CSD and EDGE ECSD) and packet-based data (GPRS and EDGE EGPRS); CDMA: circuit-switched voice and data and packet-based data (IS-95 and cdma2000 1x); SIM Application Toolkit and SMS. Other standards can be implemented by licensees due to the extensibility of the APIs.
- **International locale support** – native Unicode characters, flexible text input framework, and additional font and text formatting (supporting the Unicode Consortium standard)
- **Data synchronization** – over-the-air (OTA) synchronization support using SyncML DataSync. PC-based synchronization enabled by PC connectivity toolkit, supported over serial, infrared, Bluetooth

and USB links. Framework provides synchronization of PIM data, transfer of files, and document conversion to and from non-Symbian OS formats

- **Security** – full-strength encryption and certificate management, secure communications protocols (including HTTPS, WTLS and SSL), WIM framework and certificate-based application installation
- **Software development** – four main programming and content development options: C++, Java (J2ME MIDP 1.0 and PersonalJava 3.0 with JavaPhone 1.0 options), WAP, and web; tools for building C++ and Java applications and ROMs, and for in-target debugging.
- **Support for multiple user interfaces** – any input mechanism from full QWERTY keyboard, pen-based touch screen, to numeric mobile phone keypad.

This diagram shows a general overview of the operating system:

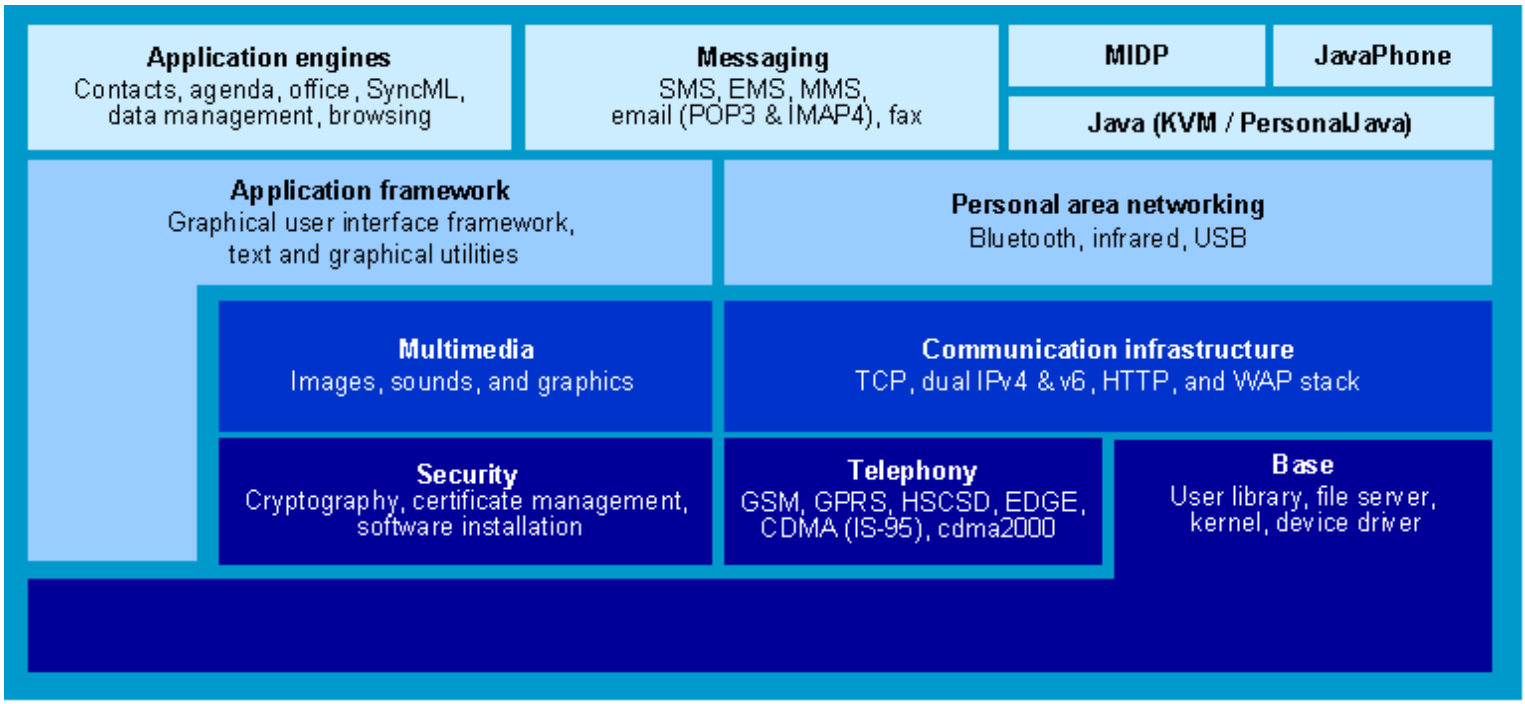

Diagram 1 – Symbian OS v7.0 architecture

3. Application engines

The core application engines include: the agenda engine, the contacts model, the sheet engine, the alarm and world servers and the help engine.
Main features are:

- agenda engine: generic client-server shared access schedule and to-do engine; vCalendar support, support for category field, connectivity requirements, SMS support (get a specified vCalendar ready for sending by providing the API to set the status property for each attendee the vCalendar is sent to), support for 'received' attribute (adds an extension property parameter to the attendee property to state whether the attendee has received the vCalendar; provide API to get and set this property), sophisticated repeating events
- contacts model: generic client-server shared access contacts database engine, integrates with messaging application for emails, faxes and SMSs, caller number matching, receive a vCard, group support, support for multiple templates, support for unknown field types, current item support, connectivity requirements, searching and filtering by contact item type
- sheet engine: spreadsheet support for multiple worksheets, rich text formatting for cells, borders and shading, many general, scientific, financial and statistical functions, formula evaluation and charts, added a power function
- help engine: context sensitive help engine consisting of four parts, a launching mechanism, the model (which describes the database to its clients), the SQL search engine (capable of relational-like searches across all the tables in the database), and incremental facilities for use by the PC-based help file authoring system client
- chart engine: renders the graphics for a chart view of a spreadsheet application
- text to Symbian OS Word converter: provides conversion between plain text and a Symbian OS Word model stream, and back again
- data application engine: the engine for a free-form database application
- word application engine: the engine for a word processor application

3.1. SyncML DataSync client

The SyncML DataSync client compliant to the SyncML DataSync 1.01 specification. The SyncML DataSync client has some particularly useful features:

- Database Adapter (DBA): allows the Sync Client Engine to extract synchronization information from and exchange data with the database that is being synchronized. This will typically include getting changes, fetching and updating items, etc. The Database Adapter Plug-in API (DAPI) enables the Sync Client Engine to be completely decoupled from the Application Database and Application Engine. The Sync Client Engine is thus totally generic and independent of any database and its data
- Transport API (TAPI): this provides a common generic API to the Sync Client Engine so that it is able to initiate connections, send and receive data etc. in a manner that is independent of the underlying transport protocol and transport media used. An HTTP TPA is provided
- A Contacts DBA and an Agenda DBA are provided to allow Contacts and Agenda synchronization respectively
- Database and transport adapters are plug-ins so that additional ones can easily be written and installed

3.2. Web engine

The Opera browser engine is a modern browser engine, with similar functionality as provided in Opera's desktop products. Opera uses the standard Symbian OS sockets mechanism, and as such has the ability to browse over GSM-CSD, GPRS, CDMA, TCP/IP, etc. The engine is also capable of browsing local files. The engine supports the following features:

- rendering: HTML 4.01, CSS-1 and CSS-2, and MHTML
- XML: XML 1.0 with support for XHTML 1.0 and XHTML Basic documents
- ECMAScript: support ECMA-262 version 2 standard (and most of version 3). JavaScript environment: partial DOM and DHTML

- Java: supports downloading and installation of Java MIDlets
- image formats: gif, animated gif, jpeg and png
- plug-ins support using subset of Netscape plug-in architecture
- FTP downloads
- handling of unknown URL schema. This includes support to pass “mailto:”, “fax:” and “sms:” schemas to the messaging application
- handling of unknown MIME types: files of types not natively supported by the browser are passed to other applications

4. Messaging

The messaging framework supports sending and receiving of text messages (SMS), enhanced messages (EMS), multimedia messages (MMS), email and fax messages. The framework uses polymorphic message type modules to handle specific types of message.
A send-as API enables the creation of messages (email, fax, SMS, EMS or MMS) straight from another application; the messages are placed in the outbox of the message store. For example, application-specific data such as vCards can be sent directly from Contacts.
BIO messaging uses a watcher framework to support messages sent over-the-air to the operating system rather than to the end-user. BIO message types currently supported include compact business card, vCard, vCalendar, email notification, operator logo, ring tone, and settings for internet access, MMS and WAP. The vCard and vCalendar BIO message types are also supported over infrared and Bluetooth links.

Main features of Messaging are:

4.1. Short Message Service (SMS)

SMS support consists of an SMS stack with a messaging API to send and receive SMS and provides the following features:

- the SMS stack is implemented as a plug-in protocol. The GSM (03.40) SMS protocol is provided
- The GSM SMS stack can be used as a bearer for the WAP protocol module
- transmission and reception of GPRS SMS
- SMS: send and receive streamed SMS messages. Enumerate, read, write and delete access to the SMS storage areas of the phone and SIM. Receive messages that match a specified text
- 7-bit SMS alphabet, 8-bit SMS alphabet and UCS2 data coding scheme are supported
- supports sending and receiving concatenated SMS messages
- scheduling of sending: on a specific date, now or upon request. Specify and review scheduled actions

4.2. Enhanced Messaging Service (EMS)

EMS support in Symbian OS is compliant with 3GPP release 4 (TS 23.040) and supports the following features:

- mobile originated pictures: variable picture, pre-defined picture: 16 x 16, pre-defined picture: 32 x 32
- mobile terminated pictures: variable pictures (1024x1 to 8x128), small pictures 16 x 16 and large pictures 32 x 32
- animations: pre-defined animations (multiple separate animations), black & white animations and mobile terminated user-defined animation
- sounds: iMelody
- formatting: both mobile originated and terminated formatting, text size (small, medium, large), text style (bold, underline, strikethrough, italic) and message alignment (left, center, right)

4.3. Multimedia Messaging Service (MMS)

MMS operates over CSD and GPRS and provides the following features:

- both WSP and HTTP transports are supported. Messages are received over WSP Get or HTTP Get, and sent using WSP Post or HTTP Post
- message notification is received over WAP 1.2.1 Push or over HTTP by a similar push mechanism

- parameters supported include: Message-Type, MMS-Version, Date, From, To, Cc, Bcc, Subject, Message-Class (only personal), Expiry, Priority, Delivery-Report, Content-Type, Response-Status and Response-Text
- both Internet and MSIDSN addressing are supported, including mixed addressing
- message presentation is based on SMIL 2.0. It is also possible to receive messages based on SMIL 1.0

4.4. Email

Email has the following main features:

- Internet mail: support disconnected mode, cache management, get-new-mail for both POP3 and IMAP4 MTMs, SMTP client enhancements (copy-to-self, separate emails for Bcc:, send email through a specified SMTP server via a specified ISP account, multiple SMTP connections with multiple send sessions), UUE and MIME, MHTML (apart from editing in place a received MHTML mail), automatic receipt notification, automatic MIME character set conversion, automatic email signature (or vCard). Character set conversion takes place during sending or receiving messages
- Internet access points (IAP): connection over GPRS is supported as well as over GSM CSD. Multiple IAPs, both GPRS and GSM CSD, can be associated with each email MTM to specify whether the default preferred connection or a specific connection should be used.
- secure socket connections: facility to establish a TLS socket connection to email servers

4.5. Fax

The fax system interfaces to the messaging fax components at its upper boundary and to fax devices at its lower boundary. Fax supports the following features:

- fax class 1, 2 and 2.0 (ANSI/TIA/EIA 578 and ANSI/TIA/EIA 592), conforms to the ITU T.30 specification
- multiple recipient outgoing faxes
- ITU T.4 1D and 2D-encoding
- scheduling of sending: on a specific date, now or upon request. Specify and review scheduled actions

5. Multimedia

The Graphics subsystem provides Symbian OS applications with shared access to the screen, keyboard and pointing devices input, bitmap fonts and scalable fonts (provided through the Open Font System), and bitmaps by using a shared heap. It also implements the Graphics Device Interface (GDI), providing a generic framework for drawing to any graphics device, and supplies concrete implementations for drawing to windows or to a printer. Main features are:

- direct screen access: the windows server allows video (as in any visual content or graphics) to be safely rendered from hardware or a device driver
- anti-aliasing support on all types of displays (monochrome, color, color using palette, etc) to improve readability

The Multimedia subsystem's media server provides audio recording and playback, and image related functionality. The main features are:

- core server framework consisting of a shared server utility library including support for runtime extension through plug-in libraries, a client side interface library, two shared data framework libraries, and two plug-in data framework libraries
- image framework consisting of a shared library to support commonly used image functionality, a plug-in library to support reading/writing common image file formats, and a client-side image utility library to simplify common image operations. Image formats supported include: decoding of JPEG, BMP, MBM, GIF, TIFF, PNG, WBMP, WMF and Smart Messaging images; encoding of JPEG, BMP and MBM
- 2D Hardware Abstraction Layer (HAL) to allow hardware accelerator support for some 2D graphics operations including BitBlt operations , rectangle-fill, polygon drawing, rectangle-invert and rectangle-fade operations

- audio framework consisting of a shared library to support commonly used audio functionality, a plug-in library to support reading/writing and streaming common audio file formats, a client-side audio utility library to simplify common audio operations, and reference audio hardware interface plug-in library. Audio formats include WAV, AU, WVE and RAW in various different sub-formats. The media server supports codec plug-ins with variable media frame size
- server interfaces: generic plug-in, audio hardware plug-in (supports local and telephony audio; 8/16 bit PCM, ALaw and μLaw encoding schemes, variable/multiple sample rates; hardware DTMF, tone and tune generators), audio file format plug-in, and image file format plug-in. Support for new formats can be added at runtime by the addition of extra plug-in libraries
- client interfaces: session, controller, resource (implements two types of ports, random access clips, and serial access streams), registry, timer, audio play utility, audio record utility, various image utility (converts image files to bitmaps), and audio tone utility

6. Application framework

The Application Framework subsystem provides a powerful environment for licensees and partners to create differentiated user interfaces while enabling applications written in C++ and Java, by Symbian, licensees, partners and third parties to run seamlessly on open Symbian OS phones. This subsystem is architecturally central to the support of graphical user interface (GUI) applications. It includes a system-wide plug-in mechanism for instantiating components at run-time, powerful reusable libraries for data, graphics and text support. A utility library for generic internet functionality is also provided. This includes some simple text, URI, WSP and time parsing utilities that are commonly used by Internet applications.

Main features of the application architecture:

- runs applications as separate processes
- associates a document with its application with the appropriate application icon
- provides a data-type recognition framework (MIME-types and others)
- defines a plug-in framework which enables the location of applications which may be added or removed by the user via removable media at any time
- internationalization support
- supports object embedding
- provides a generic error mechanism
- provides a notification scheme to third party and other add-on applications

6.1. Graphical user interface (GUI) framework

A principal objective of the graphical user interface (GUI) framework is to define as little policy as possible and therefore minimize the constraints placed on a product's UI designer. This eases the porting of the user interface of applications between different Symbian OS phones. Main features of the GUI framework:

- an event-driven GUI and widget architecture
- a windowing system for sharing screen, keyboard and pointer between applications; clocks and animated bitmaps, and a control framework for sharing an application window between application components
- direct navigation link (DNL) system enables close task-based integration between applications
- a mechanism for the licensee to customize the look and feel (LAF) of the GUI for each phone-type without affecting the drawing code in individual controls.
- a plug-in mechanism for the user to input non-standard data (e.g., for ideogram input or voice recognition for phones that may not have a keyboard)
- control factory structure allows the framework to be extended by multiple applications
- a notifier framework allowing system events and alarms to be handled more flexibly by the GUI
- a flexible screen indicator and status bar framework
- flexible listbox and scrollbar styles
- ability to automatically add menu panes to every application
- runtime changeable color schemes
- bitmap animation performed in the Window Server thread

6.2. Application support services

The application services are composed of components, primarily used by application engines, that provide core services:

- task scheduler: schedules launching of applications or initiation of specific application features. The task scheduler intercepts the return value of the application when it closes and interprets this value to determine if the application has exited with an error. If an error condition is detected, it is logged to the log engine
- system agent: a general repository for system wide dynamic state information. Typically it contains knowledge about communications related states, such as telephony signal strength, battery state, etc.
- log engine: recording the use of any on-board devices (especially the phone)
- alarm server: alarm persistence, sound playing
- world server: country codes, world country and city information database
- calendar conversion between the Gregorian and the Chinese calendars
- support for Eastern-Asian character sets in vCard and vCalendar
- onboard converters reference implementation for converters between Symbian OS and Windows (Microsoft Word and Excel 95, 97 & 2000) formats for spreadsheet and word documents, and between Rich Text and HTML. Conversions may be lossy: round-trip integrity is not guaranteed, however the text content is maintained

6.3. Internationalization support

The main internationalization features are:

- supports the Unicode Standard version 3.0
- a framework to support European, Chinese and Japanese locales
- a front-end processor (FEP) framework for text input using handwriting recognition or keyboard to enable input of far eastern ideographic characters. FEPs can that takes the form of floating window, fixed window, transparent window or be inline
- conversion between Unicode and other character sets, via a plug-in mechanism. The character sets that are implemented are:
  - General: UTF-7, UTF-8, modified UTF-7 (required by IMAP), modified UTF-8, 7 bit SMS (also known as 7-bit GSM), Code Page 1252, and ISO 8859-1
  - Japanese system: Shift JIS, JIS, EUC-JP (Packed), ISO 2022-JP, and ISO 2022-JP 1
- sorting rules via plug-in collation tables. The Unicode collation algorithm is used for collated string comparison. Collation can be is fully tailored for locales. Two reference rules (JIS sort order and dictionary sort order) for Japanese and one for Chinese (Pinyin method) are included
- character set conversion support for NTT DoCoMo pictographs
- partial auto-detection of character set encoding
- support for static multi-lingual ROM

6.4. Various text and graphical utilities

Main other Application Framework features and utilities:

- rich text rendering for various locales, providing a text model with character and paragraph formatting, embedded graphics, and a text view which supports efficient formatting, display and interaction
- two multi-level undo/redo capabilities: a plain text undo system that can undo text insertion, deletion and clipboard operations, and a rich text undo system that can cope with anything a rich text object can do, including embedded objects
- generic support for plug-in parsers that recognize certain strings, eg URLs, email addresses, phone numbers. This enables to run services or applications from any application in the system
- support for PC-style changeable color schemes in editable text and for auto-sizing text editors
- background images: arbitrary graphics can be drawn behind text, with control of parameters like transparency and background scrolling
- clipboard support for multiple media types (sound, images, etc.)

7. Personal Area Networking

7.1. Bluetooth stack

The Symbian OS Bluetooth stack is fully compliant with the Bluetooth v1.1 specifications. The Bluetooth stack fully implements the Generic Access Profile, the Serial Port Profile and the General Object Exchange Protocol. All other Bluetooth profiles are dependent on these three core profiles.
The stack is made up of a protocol module, a Bluetooth security manager, a Bluetooth communications server module and a Service Discover Protocol server module. It provides six interfaces:

- the Host Controller Interface (HCI) module lives at the bottom of the Bluetooth stack. The HCI communicates to standard HCI-compliant Bluetooth hardware through the Host Controller Transport Layer (HCTL). The HCTL is the protocol used to encapsulate HCI frames sent to the hardware. A default serial based HCI implementation is provided
- the Bluetooth sockets API is exposed through the standard sockets interface using the Bluetooth protocol plug-in component. The Bluetooth protocol plug-in provides support for both the Logical Link Control and Adaptation Protocol (L2CAP) and the RFComm protocol layers. The Bluetooth sockets API is the preferred means of managing Bluetooth connections as it offers more control and flexibility than the serial communications API
- the Bluetooth serial communications API is exposed through the communications server module. The Bluetooth communications server is a polymorphic DLL loaded by the communications server to provide a serial port emulation. The Bluetooth communications server module supports outgoing connections only. This is used for legacy PC datasync support, and for the Serial Port Profile. Incoming serial connections must be implemented using RFCOMM sockets
- the Bluetooth manager provides a highly configurable means of maintaining the security of the Symbian OS phone while using Bluetooth. It consists of a system server, the Security Manager and Registry module to store information about phones and services. It interacts with the user (via the configuration manager) to allow the implementation of flexible security access policies for the range of services supported by the phone, for example: high for services that require authorization and authentication (automatic access only granted to trusted devices), medium for services that require authentication only, and low for services open to all devices
- the Bluetooth Service Discovery Protocol (SDP) API exposes access to SDP client and SDP registration functionality. Registration and provision of an SDP service to remote devices is encapsulated within the SDP server, and client functionality is shared between a client API and the protocol plug-in
- the Object Exchange (OBEX) encapsulates all access to OBEX v1.2 transport layer functionality. A static library provides this functionality. The OBEX implementation is multi-transport and abstracts access to OBEX over Bluetooth RFComm sockets as well as over infrared TinyTP sockets

7.2. Infrared

The infrared IrDA stack is contained in a socket server protocol module that implements the following IrDA layers: IrLAP v1.1, IrLMP v1.1 and IrTinyTP v1.1. The following features are supported:

- infrared (SIR) supporting throughputs of 9.6 Kbps to 115.2 Kbps
- IrOBEX v1.2 (IrDA object exchange)
- IrTRANP v1.0 digital camera picture infrared transfer protocol
- IrCOMM v1.0 supports fax/modem functionality and is implemented in a serial communications server module

The infrared message type module integrates IrOBEX handling into the messaging framework.

7.3. USB

Symbian OS supports a USB 1.1 client interface through a kernel driver API. The USB client interface permits multiple USB functions and ‘USB classes’ to operate simultaneously.

8. Communication infrastructure

The Comms Infrastructure subsystem provides the key frameworks and system services for communications and networking. This includes:

- a communications database manager which controls the system-wide communications configuration
- a socket server and client-side API which provides a framework for implementing various communications protocols through sockets. Plug-in protocols are dynamically loaded.
- a network interface manager which provides a framework for connection to other computers or networks. The manager provides a mechanism for the client to monitor progress over e.g., a PPP connection
- a serial communications server provides a serial port (RS232C) abstraction to allow Symbian OS phones to function as a DCE and a DTE as required. Dynamically loadable plug-in communications modules are used to actually communicate with device drivers and other protocol stacks.
- HTTP and WAP stacks

8.1. Networking

A dual stack is provided that supports both IPv4 and IPv6. The IP stack provides a plug-in architecture allowing licensees or ISVs to implement extensions. An important plug-in delivered is IPSec, for secure communications. See Appendix A for the list of supported RFCs.

Networking support in Symbian OS includes:

- Transmission Control Protocol (TCP)
- User Datagram Protocol (UDP)
- IPv4/v6 stack. The TCP/IP stack provides a plug-in architecture. Plug-ins can interact with OSI level 2, 3 and 4 components and can be installed, loaded and unloaded at runtime. IP-based Symbian OS clients such as email, http, SSL, Java MIDP, SyncML DataSync over http and web can use IPv6 addressing as well as IPv4 addressing:
- Internet Control Message Protocol (ICMP)
- Point to Point Protocol (PPP)
- Domain Name System (DNS)
- dial up networking support
- security protocols for secure electronic commerce: Transport Layer Security (TLS) and Secure Sockets Layer (SSL). The TLS module is effectively an enhancement of the SSL protocol, client code can inter-operate with both SSL and TLS servers
- IPSec: IP layer protocol used to secure host-to-host or firewall-to-firewall communication. IPSec is a plug-in module to the IP stack providing tunneling, authentication and encryption for both IPv4 and IPv6. VPN clients based on IPSEC will be commercially available from Symbian partners
- Telnet Protocol engine
- File Transfer Protocol (FTP) engine
- Ethernet support: wired interface (PCMCIA cards for WINS and on-board Ethernet chip for development board) supports 10BaseT and 100BaseTX in full or half duplex. Wireless interface (IrLAN). supports SIR

8.2. HTTP stack

- HTTP 1.1 compliant client stack to enable applications such as SyncML DataSync, GPRS, OCSP and streaming multimedia to operate over TCP/IP. Third party browsers can use this stack as well
- Transport Framework architecture providing a generalized mechanism for HTTP-like protocols that operate over various transports. The framework provides a unified, high-level API independent of particular header representations, details of the HTTP-like protocol and choice of underlying transport. Using this API, a client can choose an HTTP-like protocol, encoding and transport. This is used as the interface to the WAP WSP Stack and the HTTP Text Mode stack implementations.

8.3. WAP stack

The WAP stack subsystem includes support for WAP 1.2.1 (WAP June 2000), push functionality and GPRS as a bearer. The WAP stack supports protocol specifications version 1.1 and 1.2.1 class C of the WAP Forum in

connection-oriented mode. The WAP stack supports the following bearers: GSM CSD, GPRS UDP, CDMA and cdma2000 1x for connection-oriented browsing, GSM CSD, GPRS UDP, GSM SMS and GPRS SMS for connectionless push. The WAP 1.2.1 compliant communications stack, with interfaces to each of the layers of the protocol stack, has the following layers:

- WSP, session protocol for WAP
- WTP, transaction protocol for WAP
- WTLS, transport layer security protocol for WAP
- WDP, datagram protocol for WAP, client and server
- a WAP push watcher listens for secure and non-secure push messages received using connectionless mode over all supported bearers (both GSM & GPRS are supported). Incoming push messages are identified by an application-id in the message header and handled by plug-in DLLs (a handler for the WML User Agent is provided). The WML User Agent handler identifies the content of the message by the MIME type specified in the header

9. PC-Connectivity toolkit

PC-Connectivity toolkit consists of components on the Symbian OS phone and components on a PC, which cooperate to provide connectivity services and enable data synchronization.

9.1. Connection manager

The Connection manager based on Intuwave's m-Router manages connections between a PC and a Symbian OS phone. It includes both PC-side and phone-side components. Key features of the Connection manager are:

- abstraction of the bearer from the protocol layer. The Connection manager works over a range of peer to peer bearers. Supported bearers include serial links, infrared and Bluetooth.
- ability to support multiple client applications on the PC. This is achieved by ensuring that all data transfers are atomic operations. The Connection manager can multiplex/de-multiplex data to/from a Symbian OS-side custom server component. In addition, the Connection manager has the ability to broadcast to all its clients changes of connection state
- supports for PC-side 'unify' operation. This permits a selection of tasks to be run at a single click/cradle button press
- support for PC or Symbian OS phone based connection initiation
- detection of unexpected disconnection of a phone and broadcast of disconnection to all clients on the PC. This ensures that all custom servers open on the Symbian OS phone are shutdown
- provision of APIs for the following functionality: engine interface, RRemoteLink API and custom server API

9.2. Connect toolkit

The PC-Connectivity toolkit has many features, some to offer functionality out of the box, and some to ease the job of developers wanting to plug into the synching architecture:

- framework architecture
  - framework UI
- view plug-ins: an engine and a UI component, which plug into the framework UI
  - archive application: lists and restores files archived using the backup facility
  - task scheduler: carries out a number of regularly scheduled tasks including sync and backup. A 'unify' facility allows a selection of tasks to be run at a single click/cradle button press
  - control panel: gives the user access to all the connectivity settings from one place. Individual control panel items are applets that plug into the control panel. These are: connection, log settings, machine manager, and file types
- task drivers: they consist of an engine and property pages, and plug into the task scheduler component
  - backup task driver: provides the ability to back up a connected Symbian OS phone to the PC. Backed up files can later be restored using the archives view plug-in
  - framework to enable sync components to be integrated

- miscellaneous
  - Explorer shell extension: allows for browsing, moving, copying, converting and deleting files on the Symbian OS phone via Windows Explorer (reference implementation)
  - error logger: engine that allows for logging and viewing of errors and information messages

10. Telephony

The Telephony subsystem provides a multimode API to its clients. This API abstract cellular networks including GSM, GPRS, EDGE, CDMA (IS-95) and 3GPP2 cdma2000 1x (Release A) and is ready for 3GPP WCDMA making it easier for handset manufacturers to port Symbian OS from one mobile phone standard to another. The multimode telephony abstraction is key in Symbian OS providing an integration with the rest of the operating system that enables the creation of advanced data services.
Functionality common to all networks includes:

- phone and network information: retrieve signal and battery strengths, provide access to the network names detected by the phone, information about the current network, receive notifications when there are network registration changes and retrieve the phone identity information
- phonebook: read, write, search and delete access to the phonebook storage areas of the phone and SIM (GSM 11.11) or R-UIM (cdma2000 1x)
- both one-box and two-box configurations are supported

10.1. GSM/EDGE telephony

Support for GSM, GPRS and EDGE conforms to the 3GPP GSM Phase 2+ (releases R97/98).

10.1.1. GSM

The GSM telephony framework provides an abstract telephony interface for GSM voice, data and fax, and for landline modems for data and fax as well as phone number resolution and SIM Application Toolkit. Main features are:

- voice calls: initiate, terminate and answer voice calls
- circuit-switched data calls: initiate, terminate and answer data calls including HSCSD. Pass the control of serial port to communication protocols to stream data
- the abstraction supports a wide variety of ETSI GSM phase 2+ functionality
- GSM phase 2+ SIM Application Toolkit, Class 3 (ETSI 11.14 R98), with the exception of class ‘a’ and class ‘b’
- supplementary services supported include: Alternative Line Service (ALS), Alternating Call Services (between voice and data, and voice and fax), retrieve NITZ time information, call forwarding, call waiting, call barring, Called/Calling Party Identity Presentation (CLIP) and Restriction (CLIR), setting up Closed User Group (CUG) call, User-User signaling (UUS), conference call, charging information, message waiting identification, network service requests (USSD)

10.1.2. GPRS

The General Packet Radio Service (GPRS) framework provides an abstract telephony interface for GPRS class B functionality. GPRS Phase 1 Release 97 ETSI are the specifications implemented. With class B functionality, phones are able to make and receive GSM calls while simultaneously remaining registered with GPRS. If a Packet Data Protocol context is active, GPRS services are automatically suspended and resumed. The main features are:

- attachment and detachment from the GPRS network
- activation and deactivation of a Packet Data Protocol (PDP) context for data transfer
- ability to activate and deactivate the PDP context automatically with no explicit client intervention
- ability to automatically suspend a GPRS data connection when an incoming or outgoing GSM voice call is made, and to resume a suspended GPRS data connection on notification from the GPRS network
- information and notification service to the client software of network information such as GPRS capabilities, current GPRS network availability, change in the current state of a GPRS connection and general PDP contexts parameters

10.1.3. EDGE

The Enhanced Data-rates for Global Evolution (EDGE) framework provides an abstract telephony interface for 3GPP GSM/EDGE. In addition to supporting the GSM and GPRS functionality described above, main features are:

- supports EDGE enhanced CSD (ECSD)
- supports EDGE enhanced GPRS (EGPRS)

10.2. CDMA telephony

10.2.1. CDMA (IS-95)

The CDMA telephony framework provides an abstract telephony interface for CDMA (IS-95) voice, data (circuit- and packet-switched) and fax. Main features are:

- voice calls: initiate, terminate and answer voice calls
- circuit-switched data: support for service options: asynchronous data and fax for both rate Set 1 and rate Set 2
- packet-switched data: support for service options: CDPD for both rate Set 1 and rate Set 2
- text messaging (SMS): SMS support is provided by an abstraction of the interface between the SMS teleservice layer and the SMS transport layer (IS-637)
- operation in AMPS (Voice only) networks
- forward compatibility with cdma2000 networks
- single stack Quick Net Connect (QNC): packet data service that runs on top of a circuit switched connection, typically at a rate of 14.4 Kbps. This service enables fast set-up of a direct connection to the Internet
- supplementary services supported include: set preferred band class operation (band class A only/preferred, or band class B only/preferred), set preferred network operation (digital only/preferred or analog only/preferred), call forwarding, call waiting, Called/Calling Party Identity Presentation (CLIP) and Restriction (CLIR), conference call, message waiting identification, network service requests

10.2.2. cdma2000 1x

The 3GPP2 cdma2000 1x telephony framework provides an abstract telephony interface for 3GPP2 cdma2000 1x (Release A) voice, data (circuit- and packet-switched) and fax. In addition to the functionality of CDMA (IS-95) described above, main features are:

- circuit-switched data: support for IS-95B services
- packet-switched data: support for IS-95B services plus service options 22-29, 33, 34 for high speed packet data
- Removable-User Identity Module (R-UIM): support access to R-UIM files such as phonebook entries and stored SMS messages
- phonebook synchronizer: mechanism to synchronize phonebook entries stored on a SIM or R-UIM card to the contact database so that clients can access all contact data via the contacts model API

11. Security

The security subsystem enables data confidentiality, integrity and authentication by providing underlying support for secure communications protocols such as TLS/SSL, WTLS and IPSec. It also supports the authentication of installable software using digital signatures

11.1. Cryptography module

The cryptography module includes the following significant components:

- cryptography algorithms allowing data to be encrypted and decrypted and supporting symmetric ciphers: DES, 3DES, RC2, RC4 and RC5, and asymmetric ciphers: RSA, DSA and DH
- hash functions: MD5, SHA1 and HMAC
- pseudo-random number generator for generating cryptographic keys

11.2. Cryptographic token framework

The cryptographic token framework enables licensees to integrate support for removable hardware devices, such as WIM modules, in a flexible manner. It consists of two parts:

- a framework which enables application code to query the system for the availability of implementations of specific cryptographic interfaces and their attributes (e.g., whether they are implemented in hardware, whether they are removable, whether they implement their own access control mechanism)
- the definition of a set of cryptographic interfaces. Licensees may supply their own implementations of any of the defined interfaces and these will be picked up by applications using the framework (so for example they may provide a WIM implementation which implements the certificate storage interface, and then certificates stored on the WIM will be visible in the certificate management application and available to the certificate validation module).

The following cryptographic interfaces are defined:

- a read-only certificate storage interface, supporting the retrieval of X.509 and WTLS format certificates, and certificate authority (CA) and user certificates
- a writable certificate storage interface supporting the retrieval, addition and deletion of X.509 and WTLS format certificates, and CA and user certificates
- a read-only private key storage interface
- an interface to authentication objects (e.g., PINs)
- a standard interface for performing security-critical user interface operations (e.g., PIN entry)

The following implementations of these interfaces are provided:

- an implementation of the read-only certificate storage interface providing access to the WTLS certificates used by the WAP stack supplied in this release
- an implementation of the writable certificate storage interface providing access to certificates used by all other software (e.g., TLS and Software Install),
- an implementation of an application to manage stored certificates
- a reference implementation of the interface for performing security-critical user interface operations

11.3. Certificate management module

The certificate management module is used for authentication of other entities (e.g. third-party developers, web servers) to the user of the phone, and for authentication of the user of the phone. It supports WTLS certificates (as per the WTLS specification version used for the December 2000 WAP Conformance Release) and X.509 certificates according to the PKIX Certificate Profile (RFC 2459). This module provides the following services:

- storage and retrieval of certificates using the cryptographic token framework
- assignment of trust status to a certificate on an application-by-application basis
- certificate chain construction and validation
- verification of trust of a certificate
- certificate revocation checking using the Online Certificate Status Protocol (OCSP)

11.4. Software installation

The software installation system provides a secure and fast installation process. The installation tool supports:

- install, select, run and remove installed packages and Java MIDP MIDlets. The MIDP OTA recommended practice document is fully supported
- authentication of software components using digital signatures and certificates to provide a measure of confidence that applications being installed onto a Symbian OS phone are from a known reputable vendor
- compression of the install package to reduce disk space and download times. The compression library is a generic shareable DLL which can be called by other applications
- different varieties of phones, allowing the installation package creator to ensure the correct software is installed onto an appropriate phone

12. Base

The Base subsystem provides the programming framework for all other components of Symbian OS. Base provides an abstraction to facilitate design across multiple platforms and resources making it easier to port Symbian OS to new types of hardware. The Base ensures Symbian OS robustness, performance and efficient power management – all essential in a mobile phone. The main user-visible parts of the base are the user library and the file server.

12.1. Kernel and user library

The kernel runs in privileged mode, owns device drivers, implements the scheduling policy, does power management and allocates memory to itself and user-mode (that is, unprivileged) processes. It runs natively on ARM cores. The kernel implements a message-passing framework for the benefit of user-side servers (such as the networking and telephony stacks and the file system). The user library is the lowest-level user-mode code, which offers library functions to user-mode code, and controlled access to the kernel. Here are the main features:

- process, thread, program and memory management
- error handling and cleanup framework
- descriptors: strings of characters and buffers of binary data
- container classes: arrays and lists
- active objects, for event-driven multi-tasking without requiring the overheads of multi-threading
- client-server architecture, for simple and efficient inter-process communication. The client-server architecture supports both thread-relative and process-relative client resource ownership. The latter is to ease porting of code written for other platforms to Symbian OS, and delivers considerably enhanced Java performance
- a hardware abstraction layer (HAL) presenting a consistent interface to hardware across all devices
- a kernel-side power model, to allow fine-grained power management
- silent running mode: device can operate with screen switched off
- locale support including currency, time and date formatting
- internal and tightly-coupled RAM support
- the kernel can be extended by the use of DLLs (such as device drivers and kernel extensions) that can link dynamically against the kernel

12.2. Target CPU architectures

The following CPU cores are supported:

- ARMv4: StrongARM SA1
- ARMv4T: ARM710T, ARM720T, ARM920T, ARM922T, ARM925T
- ARMv5T: XScale, ARM1020T
- ARMv5TJ: ARM926EJ
- Intel x86 (for the emulator)

12.3. Device Drivers

The Base subsystem provides device drivers and/or software controllers for the following devices:

- DTE serial port
- DCE serial port
- Infrared
- HWA (Driver implementing the hardware acceleration API for managing DSP hardware)
- USB client
- Audio drivers (playback and record)
- PC Cards
- MultiMediaCards (including support for password protected cards)
- SD Memory Cards
- LCD
- Keyboard
- Digitizer

The majority of these are split into a logical layer component implementing the higher layer functionality common to all devices of that device type together with a physical layer component implementing the hardware specific functionality.

12.4. File server

The File server provides shared access to the filing systems, a client-side interface which hides the client-server architecture and a framework for dynamically mounting plug-in file systems, with physical storage of files associated with each filing system.
The ROM filing system is built into the File Server. Two concrete plug-in filing systems are implemented: VFAT and Logging Flash. File System (LFFS). Media drivers have been developed to communicate with the following media types: RAM (non-removable), Flash (non-removable), ATA/CF, MultiMediaCard (MMC), SD Memory Card (including both the User and Protected areas of these devices).
Main features:

- file system drivers can be added when required without having to reboot
- clients can register for notification of file-server events, for example, entries changing in given directory, changing disk or disk space crossing a specified threshold
- the VFAT file system supports a 'rugged' mode of operation which provides improved data integrity in machine power loss situations

12.5. Standard library

Base also contains middleware widely used across Symbian OS. Here are the main ones:

- the C standard library
- a relational database access API. Two DBMS implementations are provided: a small and relatively lightweight client-side implementation; and, a client-server implementation for when multiple clients must have write access to a database. Databases can be manipulated either through a subset of SQL or through a Symbian OS proprietary C++ API
- a stream store that defines two major abstractions: streams (an abstract interface to convert between an object's internal and external representations) and stores (an abstract interface to manipulate a network of streams). Stores allow externalizing and internalizing data structures as complex as whole application documents (e.g., word-processor or spreadsheet documents) or databases. Several implementations are provided for both streams and stores including memory-based and permanent file stores. It is possible to define stackable streams doing pre-processing, for example encryption and decryption streams are provided

13. Software development for device creation

The Symbian OS device creation community builds Symbian OS, device drivers, middleware components, GUI frameworks and applications into Symbian OS phones. This activity is supported with Symbian OS kits, together with hardware and software tools focused on C++ development.

13.1. Symbian OS Kits

Symbian delivers Symbian OS to its licensees and development partners in two products

- Symbian OS Customization Kit, to enable licensees to quickly integrate Symbian OS into their code bases and support continued development
- Symbian OS Development Kit, a super-SDK supporting all forms of device creation development activity

The Symbian OS kits include

- virtually all Symbian OS source code
- extensive documentation and examples
- TechView, a device-neutral GUI framework
- the Symbian OS emulator, supporting quick development and debugging of all Symbian OS based code (except kernel or device drivers) on Windows-hosted PCs
- ROM building tools to build ROMs for hardware development boards, for prototype or for final phone hardware

13.2. C++ Development Tools

Symbian OS kits support two IDEs and emulator-targeting compilers:

- Metrowerks CodeWarrior Development Tools for Symbian OS, an evolving product line being developed by Metrowerks in partnership with Symbian
- Microsoft Visual C++ Professional Edition 6.0. There are no plans to support Microsoft Visual Studio .NET, or to support Microsoft Visual Studio in future Symbian OS releases

For building binaries on target phones, Symbian OS kits include GCC 98r2, developed by Cygnus (now owned by RedHat).

The target compiler and all tools that invoke it support three instruction sets:

- ARM4, for ROMs where performance matters more than space, or for applications if the ROM is known to be ARM4
- Thumb, for ROMs where space matters more than performance, or for applications if the ROM is known to be Thumb
- ARMI, for applications which interoperate with ROMs built in either ARM4 or Thumb

13.3. On-target application debugging

In addition to debugging with the emulator, Symbian OS supports on-target debugging. The tool chain produces PE-COFF/STABS object format for target binaries. Both the GDB GNU Debugger and Metrowerks' CodeWarrior support on-target debugging of user-mode programs. A stub on the target Symbian OS phone communicates, over a serial link, with the host debugger.

- GDB uses the open DGB remote debug protocol and on the host supports both a GUI and a command-line interface. The standard GDB command-line interface is extended to support Symbian OS-specific features: downloading files from the host to the target, selecting the program to debug and setting its command-line, connecting to the target, limited facilities to help debugging Unicode programs and loading DLL debugging information. On the target, the stub can work in background mode without any UI or in interactive mode where it can accept arguments from the command-line and output information to a text console
- CodeWarrior includes an application debugger in its CodeWarrior for Symbian OS Professional product line. The debugger is tightly integrated into the CodeWarrior development environment

13.4. On-target kernel debugging

Symbian OS has support for a number of JTAG debuggers for stop-mode kernel debug. These debuggers are suitable for engineers developing kernel code, writing device drivers, and porting the Symbian OS kernel to new hardware.

Debuggers with Symbian OS support include: ARM RealView Debugger, Intel XDB Nordheim debugger, Lauterbach TRACE32, Metrowerks CodeWarrior for Symbian OS OEM Edition, and Texas Instrument CodeComposer. These debuggers work across a wide range of target hardware and offer different feature sets. Some of the features available are: kernel object viewer, SW breakpoints in ROM, task-aware breakpoints, simultaneous debug sessions across multiple thread contexts, process and memory protection aware memory view, ROM build and download, executable download, user-side stop-mode debugging, integrated task-aware ETM trace, and multi-processor/DSP support.

13.5. Reference boards

Symbian OS has been tested and verified on the following hardware reference platforms:

- Intel Assabet (SA-1110)
- ARM Integrator SPP2 (ARM920T)

13.6. Hardware integration boards

Specific features and functionality have been verified on the hardware integration boards based on:

- Texas Instrument OMAP
- Intel PXA-250

13.7. Telephony stack integration and testing

A GSM telephony stack integration module is provided for test purposes. It supports voice, data, SMS, and phone information. The software module has been developed to work in conjunction with reference telephony platform from TTP Com. It may also be used by licensees as the basis of their integration component.

14. Application development

14.1. Symbian OS licensee SDKs

Symbian's licensees develop phones and support software development on these devices with SDKs for the ISV community. The Symbian OS Customization Kit provides tools required by licensees to build SDKs. SDKs may be delivered to the ISV market either directly by licensees or, along with tools offerings, by tools companies.

14.2. C++

Application development is supported in C++, for high performance, access to Symbian OS native APIs, and native application look-and-feel.

14.3. Java

Symbian OS Version 7.0 offers two configurations:

- for smartphones: MIDP v1.0 and CLDC
- for communicators: PersonalJava with JavaPhone, plus MIDP v1.0 and CLDC

The lighter smartphone configuration provides a Java environment suitable for smaller devices. The communicator configuration is the richest Java environment on mobile phones providing access to PIM, messaging and telephony functionality. With PersonalJava and JavaPhone, mobile phones combine the full power of Java with access to more specialized functionality such as messaging. With the lighter J2ME MIDP, phones have access to the many MIDlets being developed.

14.3.1. PersonalJava

PersonalJava on Symbian OS implements the 1.1.1a PersonalJava Application Environment specification. Features include:

- support for Unicode across the Host Porting Interface
- support for ARM JTEK (Jazelle) technology for Java hardware acceleration and ARM VTK (VMA technology kit) for Java software acceleration
- JVMDI for remote debugging (TCP/IP over the serial link)
- Symbian OS SDK for Java tools. Runtime customization tools

14.3.2. JavaPhone

The JavaPhone component delivers a set of APIs that extend the PersonalJava runtime to access important native functionality including telephony, agenda, contacts, messaging and power monitoring. Symbian OS provides the JavaPhone 1.0 reference implementation.
The following APIs are provided:

- JavaPhone APIs: address book (based on the contacts application engine), calendar (based on the agenda application engine), user profile, network datagram, and power monitor (minimal implementation).
- optional PersonalJava interfaces: serial communications, secure socket communications (HTTPS is supported, javax.net.ssl is not implemented)
- Java Telephony API: JTAPI Core package
- Java Telephony API (mobile): Java Mobile API Core interfaces, MobileProvider, MobileProviderEvent, MobileProviderListener, MobileAddress, MobileTerminal, MobileNetwork, MobileRadio, MobileRadioEvent and MobileRadioListener

14.3.3. MIDP

J2ME MIDP provides a Java environment for even the most memory-constrained mobile phone. It provides an installation and execution environment for the many MIDlets being developed.

- Symbian's MIDP implementation is compliant with V1.0 of the MIDP specification and v1.0 of the CLDC specification

- supports installation of JAR and JAD files
- fast implementation with a small footprint
- supports ARM's VMA technology kit (VTK), which provides software acceleration for the virtual machine's byte code interpreter
- uses light-weight threading with non-blocking IO support to ensure that waiting for IO on one thread will not block all other threads
- heap size and persistent storage are unconstrained, allowing larger, more powerful MIDlets to be run
- MIDlets look and behave very much as native applications, they use the native application installer and launcher, and native UI components
- supports native color depth (4096 colors)
- Generic Connection Framework implementation includes sockets
- Conforms to Over-The-Air Recommended Practice document for MIDlet provisioning

Appendix A. Supported messaging and networking RFCs

- Transmission Control Protocol: RFC 793 Transmission Control Protocol (TCP), RFC 1122 Requirements for Internet Hosts – Communication Layers, RFC 1323 TCP Extensions for High Performance, RFC 2018 TCP Selective Acknowledgement Options, RFC 2581 TCP Congestion Control, RFC 2414 Increasing TCP's Initial Window, RFC 2582 The NewReno Modification to TCP's Fast Recovery Algorithm, RFC 2873 TCP Processing of the IPv4 Precedence Field, RFC 2988 Computing TCP's Retransmission Timer, Proposed Modification to Nagle's Algorithm, RFC 3042 Enhancing TCP's Loss Recovery Using Limited Transmit and RFC 2001 TCP Slow Start Algorithm
- User Datagram Protocol: RFC 768 User Datagram Protocol (UDP)
- Internet Protocol v4 RFCs supported: RFC 791 Internet Protocol (IP), RFC 919 Broadcasting Internet Datagrams, RFC 922 Broadcasting Internet Datagrams in the presence of subnets, RFC 1144 Compression of TCP/IP Headers for Low Speed Links, RFC 1191 Path MTU Discovery and RFC 1853 IP in IP Tunneling
- Internet Protocol v6 RFCs supported: RFC 2460 Internet Protocol v6, RFC 2373 Addressing Architecture, RFC 2374 Aggregatable Global Unicast Address Format, RFC 2461 Neighbor Discovery, RFC 1981 Path MTU Discovery, RFC 2462 Stateless Address Autoconfiguration, RFC 2473 Generic Packet Tunneling, RFC 2464 Transmission of IPv6 Packets over Ethernet Networks, and RFC 2893 Transition Mechanism for IPv6 Hosts and Routers (dual stack functionality)
- Internet Control Message Protocol: RFC 792 Internet Control Message Protocol (ICMP), RFC 2463 ICMP v6 and RFC 950 Internet Standard Subnetting
- Point to Point Protocol: RFC1661 Point to Point Protocol (PPP) including use of LCP echo/reply packets to ensure link quality, RFC 1662 PPP in HDLC like framing, RFC 1334 PPP Authentication PAP and CHAP, RFC 1994 Challenge Handshake Authentication Protocol (CHAP), RFC 1332 Internet Protocol Control Protocol (IPCP), RFC 1877 IPCP Extensions for DNS (Domain Name Service) and NBNS (NetBios Name Service), RFC 1570 PPP LCP Extensions (callback, identification and time-remaining), RFC 1962 PPP Compression Control Protocol (CCP), RFC 1974 PPP Stack LZS Compression Protocol (only modes 1, 3 and 4, since no routers support mode 2), RFC 1978 PPP Predictor Compression Protocol, RFC 2118 Microsoft Point-to-Point Compression Protocol, RFC 2433 Microsoft PPP Chap Extensions and PPP Callback Control Protocol, RFC 2472 IPv6 over PPP, and RFC 2153 PPP Vendor Extensions
- Domain Name System (DNS): the parts of RFCs 1034, 1035 and 1101 appropriate to an IP client, namely the PTR, CNAME, NS and A DNS record types, and RFC 1886 DNS Extension to support IPv6
- security protocols for secure electronic commerce: RFCs: RFC 2246 TLS Protocol Version 1.0, RFC 2487 SMTP Service Extension for Secure SMTP over TLS and RFC 2595 Using TLS with IMAP, POP3, and ACAP
- IPSec: RFC 1828 IP Authentication using Keyed MD5, RFC 1829 The ESP DES-CBC Transform, RFC 2085 HMAC: Keyed=Hashing for Message Authentication, RFC 2401 Security for the Internet Protocol, RFC 2402 IP Authentication Header (AH), RFC 2403 The Use of HMAC-MD5-96 within ESP and AH, RFC 2404 The Use of HMAC-SHA-1-96 within ESP and AH, RFC 2405 The ESP DES-CBC Cipher Algorithm with Explicit IV, RFC 2406 IP Encapsulating Security Payload (ESP), RFC2407 The Internet IP Security Domain of Interpretation for ISAKMP, RFC 2048 Internet Security Association and Key Management Protocol, RFC 2409 The Internet Key Exchange (IKE), RFC 2410 The NUKK Encryption Algorithm and its Use with IPSec, RFC 2459 Internet X.509 Public Key Infrastructure Certificate and CRL Profile, RFC 2367 PF_KEY Key Management API Version 2 and RFC 2560 X.509 Internet Public Key Infrastructure Online Certificate Status Protocol – OCSP
- Telnet Protocol engine: RFC 854 Telnet Protocol Specification, RFC 855 Telnet Option Specifications, RFC856 Telnet Binary Transmission, RFC857 Telnet Echo Option, RFC858 Telnet Suppress Go Ahead Option, RFC859 Telnet Status Option, RFC727 Telnet Logout Option, RFC1091 Telnet Terminal Type Option, RFC1073 Telnet Window Size Option, RFC1079 Telnet Terminal Speed Option
- File Transfer Protocol (FTP) engine: RFC 959 File Transfer Protocol and RFC 2428 Extensions for IPv6 and NATs

- Ethernet support: connectionless IEEE 802.2, RFC 826 Ethernet Address Resolution Protocol (ARP), RFC 2131 Dynamic Host Configuration Protocol, RFC 1497 BOOTP Vendor Information Extensions, 2132 DHCP Options and BOOTP Vendor Extensions, Ethernet (EtherII) and IEEE 802 Encapsulation as defined in RFC 1122 Requirements for Internet Hosts - Communication Layers, RFC 894 Standard for the transmission of IP datagrams over Ethernet networks and RFC 1042 Standard for the transmission of IP datagrams over IEEE 802 networks
- HTTP and HTTPS: RFC 2616 Hypertext Transfer Protocol -- HTTP/ 1.1, RFC 2617 HTTP Authentication: Basic and Digest Access Authentication, RFC 2817 Upgrading to TLS Within HTTP/1.1, RFC 2818 HTTP Over TLS, RFC 2965 HTTP State Management Mechanism
- IMAP4: RFC 2060 Internet Message Access Protocol – Version 4
- POP3: RFC 1939 Post Office Protocol - Version 3
- SMTP: RFC 2821 Simple Mail Transfer Protocol (commands useful on a client), RFC 1869 SMTP Service Extensions (EHLO command)
- message presentation: RFC 822 Standard for the format of ARPA Internet text messages, RFC 2045 Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies, RFC 2046 Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types, RFC 2047 MIME (Multipurpose Internet Mail Extensions) Part Three: Message Header Extensions for Non- ASCII Text, RFC 2557 MIME Encapsulation of Aggregate Documents, such as HTML (MHTML), RFC 2392 Content-ID and Message-ID Uniform Resource Locators, RFC 2806 URLs for Telephone Calls, RFC 2368 The mailto URL scheme, RFC 2183 Communicating Presentation Information in Internet Messages: The Content-Disposition Header Field, RFC 2279 UTF-8, a transformation format of ISO 10646, RFC 2152 UTF-7 A Mail-Safe Transformation Format of Unicode, and RFC 2306 Tag Image File Format (TIFF) - F Profile for Facsimile

Acknowledgements

The author wishes to thank the many Symbian staff who supplied valuable information and review comments.

Symbian licenses, develops and supports Symbian OS, the platform for next-generation data-enabled mobile phones. Symbian is headquartered in London, with offices worldwide. For more information see the Symbian website, http://www.symbian.com/.