

What's New for Developers in Symbian OS v9.4

Mark Shackman

Published by the Symbian Developer Network

Version: 1,0 – October 2007

1 INTRODUCTION	1
2 ARCHITECTURAL ENHANCEMENTS	2
2.1 DEMAND PAGING FOR INTERNAL FIXED STORAGE.....	2
2.2 RAM DEFRAGMENTATION.....	2
2.3 FILE CACHING	3
3 NEW AND EXTENDED APIS	4
3.1 AUDIO AND VIDEO CODEC INTERFACE.....	4
3.2 MEDIA TRANSFER PROTOCOL (MTP) OVER USB.....	4
3.3 URI WHITE/BLACK LIST SUPPORT.....	4
3.4 RAM DEFRAGMENTATION.....	5
4 CONCLUSION	5

1 Introduction

Symbian OS v9.4 is a major upgrade to Symbian OS, delivering a number of architectural enhancements and new or extended APIs that are of interest to developers. Key enhancements introduced since Symbian OS v9.3 include:

- **Demand Paging for internal fixed storage** – reduces RAM usage
- **RAM Defragmentation** – better memory usage and conservation of power
- **File Caching** – improved file system performance

These are discussed in the following sections.

2 Architectural Enhancements

Symbian OS v9.4 includes a number of enhancements that are of interest to the developer. This section outlines the major architectural changes.

2.1 Demand Paging for Internal Fixed Storage

Demand paging only loads into memory the parts of a library DLL that are required by a process, rather than loading the whole library into memory. This reduces RAM usage, allowing more applications to run simultaneously or allowing more memory-hungry applications. Additionally, it helps to reduce the time required for a device to boot.

2.1.1 Demand paging details

Symbian OS v9.4 extends demand paging from the ROM file system (as supported in v9.3) to enable the paging of read only code and data from internal fixed storage.

Internal applications in mobile phones have traditionally executed in place (XIP) from NOR flash ROMs. Currently there is a trend to use hardware architectures in which applications are copied (and maybe decompressed) from NAND flash ROMs or hard disks into RAM before execution. On such architectures, Symbian OS currently copies application code and data to RAM on demand at the granularity of a whole executable. RAM consumption can be reduced by loading applications into RAM at a finer granularity - that of a hardware "page" of size 4K - and by proactively unloading from RAM those parts of an application's code or data that are deemed to be no longer in active use.

Demand paging from a compressed ROM image can improve start up responsiveness in NAND-based devices due to code loading being deferred. However, demand paging changes the run time characteristics of a system and the system therefore has to be tuned to avoid the risk of excessive reloading of pages ("thrashing"), which would impact performance and battery life.

2.2 RAM Defragmentation

RAM defragmentation contributes to better memory utilisation by neatly ordering memory blocks and releasing free memory trapped between used blocks. This frees up banks of memory that can be switched off, thereby conserving power.

2.2.1 Kernel defragmentation support

The Symbian OS Kernel system supports the defragmenting of physical RAM and provides two kernel-side APIs that can defragment physical RAM:

- The existing kernel-side memory APIs are extended to enable the allocation of physically contiguous RAM - the Kernel frees up or moves allocated RAM as necessary in order to meet the request
- A Defragment RAM API has been added kernel-side, which defragments specified areas of RAM as much as possible and reports on current memory usage

The existing implementation of the Kernel's memory allocation policy has been modified and tuned to minimise the RAM fragmentation within the system, and to provide power up and down information to memory banks.

There are a number of reasons for defragmenting physical RAM, two of which are:

- Many device drivers require physical RAM to use for buffers etc. For example, a camera driver may require a physically contiguous buffer for holding the output of the CCD. Currently such memory is typically pre-allocated at boot time. This practice increases initial RAM consumption after boot. Total RAM consumption can be reduced if memory for such buffers is only allocated as required, rather than at boot time.
- Typically, memory consumption of a phone while it is idle is less than the total memory available. Idle and active power consumption can be decreased by powering down unused RAM chips or unused RAM banks within a RAM chip.

2.3 File Caching

Files are now cached even more intelligently in Symbian OS v9.4. Read-ahead caching speeds up file access, particularly for sequential file reads. This can have a beneficial effect on device boot and application start up speeds by reducing the time required to read in resource files. "Lazy write" caching can improve the performance of applications which stream data to disk, and also reduce the need for applications to implement their own buffering.

File caching can also improve battery life, for example allowing an SD card containing MP3 files to be powered down more often, as the data is read from the card in larger chunks.

2.3.1 Improved file system performance with server-side caching

The Symbian OS file server has been enhanced to cache file and file system data, and to "read ahead" file data. The caching and read-ahead strategies are tuned to recognise and deal appropriately with typical use cases, including streaming of large media files.

The file system and media driver HAIs (hardware adaptation interfaces) have been extended to report information required by the file server to determine the optimal caching strategy. The file server uses default values (which may not be optimal) if the file system and/or media driver does not supply an implementation of this HAI.

The file server uses otherwise unallocated memory for its caches, which are automatically reclaimed if required. In low memory situations, performance degrades to no worse than current levels. Additionally, the file server provides an API to enable write caching on a per-session basis and on a system-wide basis. Write caching is disabled by default.

As part of its operation, the file server provides "fair" scheduling of client accesses to media; it breaks up long-running client requests into multiple smaller requests to ensure that no long-running client request can cause other clients' requests to be blocked for an unbounded period of time.

As a result of these enhancements, application developers should generally see faster and more consistent file read performance. Applications that use the new API to enable write caching will experience faster write performance, but are expected to cope with the consequences of data loss due to events such as power loss or media removal.

3 New and Extended APIs

3.1 Audio and Video Codec Interface

The existing multimedia hardware abstraction framework (MDF) has been extended to provide an interface for plug-in audio and video codecs. This allows developers to supply codecs that will plug into Symbian's existing multimedia hardware abstraction framework.

At the moment, although a Symbian API exists for applications and MMF Controller plug-ins (e.g. media engines from Packet Video or Real Networks) to record or play audio and video using MDF codecs, there is no defined interface for the codecs themselves. This means that current DevSound and DevVideo implementations have 'hardwired' codecs. With a suitable codec interface definition, developers will be able to provide standardized codecs which will work with various DevSound and DevVideo implementations.

Of particular note is that the interface closely follows the emerging OpenMAX industry standard for hardware acceleration, and, where possible, Symbian's codec interface definition is aligned with the proposed OpenMAX API version 1.0.

3.2 Media Transfer Protocol (MTP) over USB

Symbian OS v9.4 includes a new MTP daemon running over USB with an API for MTP Data Provider plug-ins. It implements the responder role of MTP (rather than the initiator role) and is compatible with Microsoft's "MTP Enhanced" specification v0.95 (assuming that the device conforms to the Windows Portable Device (WPD) architecture). It is thereby interoperable with Windows XP with Service Pack 2 or later (with Windows Media Player 11 or later installed) and all Windows Vista releases which include WPD. Supporting MTP allows Symbian OS-based devices to fully interact with Microsoft Windows without the need to install additional software on Windows (so, for example, the phone's file system can be seen and browsed from within Windows Explorer). Only advanced users and enterprises will need to install a PC Suite.

MTP is integrated with the USB manager, allowing it to be selected in the same fashion as other USB classes. When configured, it consumes 4 USB end points and appears as an implementation of the Still Image Class with Vendor Extensions to the USB host.

MTP is an optional component, with device manufacturers selecting whether or not to include it. If included, the first release only allows device manufacturers to create plug-ins.

The MTP daemon routes all PTP and MTP basic, enhanced and vendor extension operations based on the set that the Data Providers have registered at start up. On its own, the MTP implementation does not provide any data transfer but does allow a PC to identify the Symbian OS device and to ascertain the media types that are supported. Application services and stores that wish to have their data managed by the PC should implement a Data Provider plug-in.

The API for Data Provider plug-ins allows for a Data Provider plug-in to manage more than one media type but a single media type may not be managed by two Data Providers.

3.3 URI White/Black List Support

3.3.1 WAP Push white/black lists

The existing Symbian OS WAP Push component has been extended to compare the origin of messages against Uniform Resource Identifiers (URIs) in a "white list" and "black list", silently

discarding blacklisted messages and flagging whitelisted messages for processing by the WAP Push Handler.

The matching occurs before any WAP Push handler is invoked.

If a WAP Push message matches a black list entry, the WAP Push message is silently discarded by the WAP Push component. Otherwise, the WAP Push message is flagged as to whether or not it matched a white list entry and then dispatched to the WAP Push handler.

3.3.2 White/Black List URI Service

To support WAP Push white and black lists, Symbian OS provides a new service for querying and updating a list of URIs and associated remote access permissions. The URI syntax supported is as specified in IETF RFC 3986.

The URI syntax covers both URLs (Uniform Resource Locators, for example HTTP addresses) and URNs (Uniform Resource Names, for example OIDs). IETF guidance is that specifications previously referencing URLs should now refer to URIs (see RFC 3305).

The service provides an API which can be used to query the permission associated with a specified URI (possible permissions returned are 'allowed', 'denied', or 'unknown'). The API includes a parameter indicating the purpose of the query, so as to disambiguate similar URIs which may have been included in the list for different purposes. This query API is protected with ReadDeviceData capability.

The APIs for updating the URI list are expected to be superceded by more general APIs in a future security settings service.

3.4 RAM Defragmentation

To complement memory usage improvements in Symbian OS v9.4, a Defragment RAM API has been added to the kernel. When called, it defragments specified areas of RAM as much as possible and report on current memory usage.

Note that there is no direct user-side API to invoke RAM defragmentation.

Further details can be found in 2.2 above.

4 Conclusion

A brief summary of the key features and technical specifications of Symbian OS v9.4 can be found on the Symbian web site in the [Symbian OS Releases information page](#).

Further information on these changes will be available initially in the Migration Guides in the Symbian OS Developer Library which comes with the DevKit.

Want to be kept informed of new articles being made available on the Symbian Developer Network?

[Subscribe to the Symbian Community Newsletter](#).

The Symbian Community Newsletter brings you, every month, the latest news and resources for Symbian OS.