telephony technologies. It is not clear what mix of standards will succeed or which technologies will dominate. But clearly, to retain its current advantage, Symbian OS needs to be flexible, adaptable and to support this technology evolution seamlessly.

To some extent, the beginnings of the telephony architecture evolution go back a long way, to Symbian OS v7 and even earlier.

Andy Cloke:

We have invested a lot of time and effort in multimoding the ETel API to support the North American market where the CDMA2000 specification, the Qualcomm-influenced specification to put it like that, was prevalent. Our original ETel API was very GSM-centric. Since 3GPP specs are based on GSM, that translates nicely across to 3G in Europe and Japan and anywhere else with a GSM footprint that is upgrading to 3G, but it doesn't translate well to the North American and Korean markets. So that was the ambition, to support both GSM and CDMA, with a view toward 3GPP and CDMA2000.

Since the lead time for getting these sorts of software components right is quite long, the work started off quite early. So we had to scour the Qualcomm CDMA specifications and work to align data formats, data structures, requests, responses and notifications, and sort out where we could align and where we had to create two separate functions, one which would be used in GSM and one which would be used in CDMA2000. And, of course, the design principle is to support keeping the phone engine as simple as possible. You don't want to say 'Dial a Qualcomm voice call' or 'Dial a GSM voice call'; you just want to say, 'Dial a voice call and here's the number I want you to dial'.

The aim was not so much to support multimode phones capable of both GSM and CDMA operation, as to support phone vendors with portfolios of phones targeting all markets. For the vendor, being able to build both GSM and CDMA phones, or their 3G equivalents, from a single source base is a significant advantage (even though the source base is more complex), allowing maximum reuse of software across a phone portfolio.

Andy Cloke:

The core ETel stuff actually is very thin. So, for example, SMS transmission and reception is an ETel extension. And it turns out that the multimode ETel extension is almost so standard that you'd always have to use it. But the point is that it did give us another opportunity to redesign. For example, when we created the original GSM API, we had a split between Basic and Advanced. The idea was that you would have the core ETel which would be suitable for standard wireline modems, and the Basic ETel extensions would be suitable for a phone connected by wire or infrared, a two-box solution, and Advanced

would extend on top of that and would be suitable for a phone where the signaling stack is built in. That was the original design intent, but we never really ended up using just Basic ETel, so it was good to be able to deprecate that.

It may seem an obvious point, but in the early days of Symbian, phone expertise was hard to find. The company's own background had been strongly PDA-based, with a particular flair for squeezing powerful UIs and applications into diminutive machines. Conventional networking technologies were well understood and fixed-line telephony was a well-understood technology. But mobile telephony was still highly specialized and mostly disjoint from more conventional computing. Embedded systems was a similarly highly specialized field. Few individuals (and companies) had expertise across all those boundaries.

Andy Cloke:

By the time we got to develop Multimode ETel, we were a lot more experienced in the whole phone area. By then, we had people who had come in from companies where they had been developing GSM phones, so they thoroughly understood all this stuff.

Supporting all the available global telephony standards is a prerequisite for a genuinely capable phone operating system. Because those standards continuously evolve, the operating-system support for them must continuously evolve. Without an extensible telephony architecture that kind of evolution is impossibly hard. The goal of an extensible architecture is not simply to make evolution possible, it is also to make it as easy as possible and as safe as possible since, in any system, change is expensive; in a complex system, all change is also a risk to the stability of the system.

Another rather more minor example of the way that the telephony architecture has changed in unforeseeable ways is exemplified by the introduction of the ETel third-party API in Symbian OS v6.1. In the original release of Symbian OS v6.0, ETel was substantially open to third-party developers. The release of Symbian OS v6.0 SDKs which documented in detail the ETel APIs for the first time caused concern for some licensees,[4] reflecting extreme operator nervousness about possible implications for network security. It should be remembered that Symbian OS was not just the first, it was also the only open operating system being used as a platform for phones. (And, arguably, it still is.) Operators

[4] At that time I was managing the documentation team which had published the controversial APIs and I saw very clearly the upheavals it caused.

had simply never been in a position in which their networks were even potentially open to third-party developers. A solution was rapidly agreed, the API documentation was withdrawn and an open subset API was introduced (and documented) to abstract public telephony behavior to enable applications to control phone calls, while the underlying API was locked down for internal use only.[5]

Again, future change is foreseeable in principle but never in detail. Extensible design is the only assurance against future disruption.

17.6 Sound and Vision Evolution

Another area of rapid recent technology evolution is multimedia, and this is again an area in which Symbian OS has undergone significant architectural change. Symbian OS has always supported sophisticated sound- and image-based applications, with Symbian OS v6.1, for example, being the platform for the first Nokia camera-phone in Europe, the Nokia 7650. Multimedia really arrived for the first time in Symbian OS v7.0s which introduced a new multimedia server. In Symbian OS v8.x, the architecture was reinvented to support a full suite of sophisticated phone functions. Between the releases of Symbian OS v6.0 and v8.1 therefore the change has been dramatic, from still camera to movie camera; from instamatic-like snapshots to full-function, multimegapixel digital camera replacement; from ring tones to music player. With image- and sound-recording, -editing and -manipulation capabilities, Symbian OS is evolving to support capabilities that until recently were limited to multimedia workstations.

The underlying rationale for the multimedia architecture redesign was support for the increasingly complex built-in hardware of the new generations of phones, with their dedicated multimedia components such as graphics hardware accelerators, MIDI-tone generators, dedicated DSPs and voice synthesizers, as well as high-end imaging and audio hardware parts.

Evolution of the licensee UIs for Symbian OS has been another big driver for change in the graphics services which underlie the UI framework support. Numerous UI enhancements aimed at supporting specific UI effects, such as fading, transparency and multimillion-entry color palettes, have required a steady stream of changes, mostly focused at the level of either the Window Server or GDI and BitGDI. The Window Server is a central component of the operating system, in many senses underpinning the GUI application model of the operating system, responsible not just for the windowing model, managing screen real estate and serving display

[5] Incidentally, incompatible changes were back-ported to Symbian OS v6 to ensure that later Symbian OS v6 products could not be 'hacked' based on the previously published documentation.

regions to applications, but also responsible for the system-event model. An example of the complexities which can be involved are the challenge to the fundamentally single-threaded Window Server model by explicit multithreading in the UI. It is easy to arrive at a position in which a clash of programming models (the native asynchronous model of Symbian OS based on Active Objects versus explicit multithreading) turns into an architecture problem.

There are other challenges too. A good example of the unpredictable emergence of new technologies is the emergence of vector graphics as the future display technology of choice (or at least a plausible candidate) within the wider industry. Traditional display technologies are bitmap-based (or have been for a good many computer generations), but vector graphics for display is certainly not new. Since the days of the NeXTstep Unix machine which put Display PostScript to work to manage the phone display, it has remained an interesting possibility.

However, recently there has been a vector revolution. The big driver has been 3D and naturalistic graphics for games. 3D graphics are calculation driven and vector graphics are a natural vehicle for their expression.

Charles Davies:

Now we have the challenge of how to deal with moving to Open GL and SVG, and what happens to Window Server in that context. There's a huge amount of technology and value in the Window Server and in the GDI and, for example, none of that's available in Linux; people have to license Trolltech above Linux. That makes the Window Server quite a critical asset. But it's under threat, because the world might be moving on to vector graphics models of drawing. So we had better be careful about that. It's not going to happen tomorrow, but it's one of the challenges.

17.7 Defining the Skin

One of the trickier problems Symbian OS faces is the boundary problem, or as Charles Davies puts it, the problem of defining the skin of the operating system, the boundary which determines what is in the operating system and what is not, what value Symbian creates and what value Symbian retains compared with the value that licensees themselves create and retain. There is a complex relationship between the architecture of the operating system and the business model, which is designed to encourage platform unity while enabling variation. At the same time, Symbian has deliberately sought to create an open platform capable of supporting a broad ecosystem of partners, third-party developers, software and tools vendors, enterprise customizers and hardware vendors, as well

as licensees. Against external competition from competing systems such as Linux and Microsoft's Windows Mobile, Symbian must maximize the value it creates for its customers, thus maximizing Symbian benefit, without doing so at the expense of the value chain, which includes the ecosystem of Symbian's own partners as well as the partners of Symbian OS licensees. The essential rule, of course, is that Symbian must maximize the value it delivers without cannibalizing the value chain.

Charles Davies:

When I was at Psion, when we were building a PDA, I understood where the PDA ended and where the things outside the PDA began. I knew the boundaries of the product. I came into Symbian OS and I thought, 'Where are the boundaries?'

One of the things I've done since being here is to try to identify the scope of Symbian OS. We have had arguments like 'Should MMS have been inside or outside?' and I would have liked, dearly loved, to say, 'We do all the APIs, but we don't do the implementation'. That would be a simple thing to say, but the moment's gone. S60, MOAP and UIQ have extensive APIs. So that's not the boundary and it's really tough for both our customers and us to know where the boundary is.

To look at it another way, we've got, say, 750 people in Software Engineering doing Symbian OS and we can't make that 1500 overnight and we're certainly not going to make that 200. So with 750 people, what boundary can we draw that matches a decent product?

There are no easy answers. Not only is the system complex, the technologies are complex and the market is complex. The Symbian OS licensee model is complex.

Charles Davies:

For example, we could do an OMA DRM plug-in. Or is that an application? Is that in the scope of the OS or is that something for the UI? And the answer is borderline, borderline. With the borderline cases we will work with our customers when our capacity doesn't match it.

Keith de Mendonca has been on the engineering side of similar decisions in the past. MMS is an example. Writing MMS messaging plug-ins is complex. Providing support for it within the operating system clearly has an immediate benefit for licensees. The problem is that if some licensees take the Symbian OS solution and some do not, then both fragmentation and duplication result.

Keith de Mendonca:

We did produce an MMS solution when MMS first appeared. But we ended up with multiple solutions, with us and licensees each investing and producing basically the same technology. It was a waste of resources. Nowadays, Symbian OS no longer supports MMS directly at all, and so our customers have to provide those MTMs themselves if they want them, and they obviously do want to support MMS. So although our MMS solution is still in some products, say the Motorola M1000 phones or the v7.0 phones that are shipping, we don't supply it any more. I think the truth is, it was one of those situations where it's hard to decide who should really innovate in areas that seem to be very hot, where the technology was moving very quickly.

Avoiding the trap of duplication of effort, however, is hard. Arguably, precisely because customization is required to adapt the operating system to a given hardware platform, it may be unavoidable toward the bottom of the operating system in the hardware adaptation interfaces. On the one hand, the risk is of under-providing, which leaves too much work for licensees in porting the operating system to their hardware, while on the other hand the risk is over-providing, duplicating solutions that at least some licensees wish to supply for themselves.

17.8 Moving Towards Standard C++

A quite different, but no less important, perspective on renewal emerges from considering recent moves in Symbian OS towards a more standard use of C++, as discussed in Chapter 15.

David Wood:

Historically, we were biased towards coping with programmers who could handle quite a lot of complexity themselves, so while the operating system hides many, many complexities, still the bits that spill out are pretty difficult, and it requires a serious and very capable C++ software engineer to deal with it.

We used to say, or some of us used to say, that if you're a Java programmer stick with Java; if you're a C++ programmer, convert to Java unless you happen to be a very good C++ programmer, in which case you can stick with C++ when you're programming Symbian OS. I'm not sure we quite said it like that but that was the implication. We didn't make concessions for the 'mid-range' C++ programmer.

This is changing now, because we are at the stage where Symbian OS is going mainstream. When I say mainstream, I don't just mean that more phones are running Symbian OS, I mean that mainstream developers want to get their

applications ported onto Symbian OS and they're not prepared to put up with the level of complexity that early adopters have.

We are looking at alternative run-time environments that hide more of the complexity. And you don't get quite so much control, certainly, and you probably end up with programs that are less efficient in several ways, but the added power of the machine probably makes that a more acceptable trade off than before.

Python may be one of the examples David Wood has in mind. But providing alternative language environments for application developments does not address the fundamental problem, that for those system programmers who must use C++ (whether they are porting the operating system or extending it), Symbian OS C++ sometimes appears arcane and willfully difficult in a world where standards rule. Symbian C++ idioms include its `Leave()` mechanism and cleanup stack, Active Objects rather than explicit multithreading, descriptors rather than C++ string classes, ordinal-based function calling into DLLs rather than more conventional name-based calling, and so on.

David Wood:

We did our own implementation of exception handling, because we took the view that exception handling was insufficiently standard, which at the time it was, and insufficiently efficient. As it happens that efficiency condition was borne out to be correct too, because when we eventually we turned the compiler flag that said 'Right, enable native exception handling of C++' in Symbian OS v9, something like 10% or more in bulk was added to the code. Not only does the native mechanism require more code, it's also slower to unwind the stack. We did various things in Symbian C++ where we would unwind the stack in one whole lot just by calling `User::Leave()`. So we avoided going up the stack layer by layer by layer, unlike the corresponding function inside native C++ – I think it is `throw()`. When you `throw()`, it doesn't jump up all at once, it goes up layer by layer of the stack.

Standard C++ error handling also requires memory allocation, which, of course, fails if the error being handled is an out-of-memory error.

David Wood:

There is occasional memory allocation needed by our cleanup stack as well, but it's done in such a way that each individual call to the cleanup stack will succeed, although if it's going to run out of memory for the next allocation it then unwinds. But whatever you push onto the cleanup stack is guaranteed to still be there. So that was very carefully designed in.

It is easy enough now to look at these mechanisms and condemn them as idiosyncratic, quirky, and frustrating. However, the fact is that at the time there was no C++ standard and there were no standard mechanisms. While memory leakage on desktop systems is still frequently taken for granted as part of the programming context, that approach was simply not acceptable in the context in which Symbian OS emerged, and nor is it for the mobile-phone market as it exists today. Power, memory and CPU cycles remain scarce resources in mobile devices and desktop assumptions do not hold.

Andrew Thoelke:

Memory allocation and out of memory was very important on the Series 3 and it was going to continue to be a big deal on Series 5 and it still is a big deal on any mobile device, even though some of the assumptions have changed. So we assumed that the operating system had to run for days or months or years without memory leakage or failure resulting from managing memory because you never switched off a Series 5. C++ isn't a garbage-collected language, so a lot of the rules actually came about from asking, 'How do we help programmers write code and review code to make sure that it is memory correct?'

It also predated exception handling in C++, so the cleanup stack, which is an idea from Series 3, was brought across, as was the actual exception mechanism for error propagation. This was followed by the naming convention too, basically to say this is a class that you should only ever allocate on the heap (or sometimes as an embedded member of another object on the heap) and this is a class you can safely have as a stack item, but if you've got to close it you need to make sure that you always close it, or you have to find some way of ensuring that it can be closed as part of an exception propagation.

Martin Tasker argues that while the detailed rationale may have changed as Symbian OS has evolved for new classes of device, the design choices were sound, and remain sound. But there is no doubting the barrier they pose for entry for developers coming to Symbian OS for the first time.

Martin Tasker:

There are barriers to entry, and there are what you might call ongoing costs. What happens in a lot of programming is that people assume that clean up is either an issue that you never have to handle, or that you can terminate your program if ever a resource is unavailable. It's very easy to write that kind of code. But what gets difficult is if exceptions can occur pretty much in every function you call and then the old-style programming gets cumbersome. At this point you have got to invest some thought in asking 'What happens if any of the functions you are about to call fail? Should you set up some kind

of trap?' Whatever system you have, you've got some thinking to do which doesn't come quite so naturally. I think the solution we chose was actually quite simple compared to other solutions for the same problem. However, from the perspective of a third-party application programmer or a developer within a Symbian OS licensee, that is one of the more tricky areas. But we had to do this and our design choices were actually pretty good.

On this argument, discounting the high initial cost of the Symbian OS idiom against the resulting low ongoing costs may be a better option for developers than the opposite choice, that of using a standard mechanism which is immediately familiar but which is not optimized for the device class. Like it or not, the reality is that programming for small devices, and for phones in particular, is a specialist discipline, whether at application level (working with the different UI considerations imposed by mobility, the small footprint constraints, and so on) or at system level (ROM-based driveless devices, hardware with awkward properties such as NAND flash memory and intermittent connections). Desktop assumptions and techniques do not carry across.

Bob Dewolf:

I quite like the Symbian OS programming culture. Of course it seemed odd at the beginning because you had to learn all these strange new ways of working, like two-stage construction and the cleanup stack, but I'm quite a fan of that now; I quite like it. And I don't think that's held us back, I think that's been a good story, because it gives clear rules about how to deal with memory allocations. It's also a very strong OO-design pattern, which I also think is good in general.

Nonetheless, the pressure to standardize is being responded to.

Andrew Thoelke:

In some cases, our constraints go beyond the advisory, 'This is advisable because of the way the system's written', and become, 'If you don't do it like this your code won't work'. We definitely intend to move away from this now, and try to support a much more mainstream C++ programming environment. This is partly because there is such a thing now, which there wasn't when we started, but also because it will aid more programmers to get onto the platform and it will allow them to take code that already exists outside the platform and more easily to deploy on the platform.

The investment in already-written software has often proved decisive in relation to whether new systems succeed or not. Ease of porting is

an essential requirement for an operating system which aims to become the standard in its market. To enable Symbian OS to continue to grow as a software platform requires providing better options to those looking to port existing code from other operating systems, whether at system or application level. Requirements have been framed and the changes will come.

Of course, it is an open question how much existing Unix or Windows code would make suitable candidates for running on a phone. The porting market is almost certainly not for standard third-party code, except for games perhaps, so much as for proprietary solutions. Languages may turn out to be as significant as mechanisms.

The company's language strategy has not always been consistent. While there has always been a commitment to native C++ development, at times Java seems to have been promoted as the programming language 'for the rest of us' working on Symbian OS as a platform. Martin Tasker takes some pains to explain the truth.

Martin Tasker:

It was always assumed that C++ would be used externally and there were conscious design decisions around that. In fact, Colly Myers used to feel quite strongly about it. He would say, 'We can't assume that everybody understands operating systems and we cannot expose an API in such-and-such a category to people who don't understand operating systems'. That was because the C++ APIs were exposed. And, if you look at active objects, if you look at the C and T types which offer a very very simple guide to the programmer as to how to use these types operationally, as simple as Java objects and built-ins, then in some ways we are as simple as Java. We don't do garbage collection, because C++ doesn't do it, so programmers have to do that stuff manually. But otherwise we're as simple as Java.

For application developers, languages such as Java, Visual Basic and Python are obvious options for enabling cross-platform portability, rapid development and improved productivity on Symbian OS.

David Wood:

Java is only one option. There are now other run-time environments as well, including Visual Basic from AppForge and the run-time environment they call Crossfire. They also support a .NET library or run-time environment for Symbian OS. And then there's OPL, a very old and venerable programming environment and there are more recent contenders such as Python. The common theme of these interpreted languages is that they are less efficient in terms of manipulating the bare metal of the device, but they require less of a learning curve to become productive.

> What I've seen is evidence that you don't need to be anything like such an experienced programmer to use them, indeed non-programmers such as journalists and students of journalism or students of media or students of arts and technology are able to learn how to create quite impressive programs in Python from a course based on just one lesson per week, and that is part of Symbian OS coming to the mainstream.

18

Creative Zoo or Software Factory?

18.1 Introduction

This case study takes a step back from Symbian OS itself to explore some of the broader questions about how software is created. One consequence of the success of Symbian OS has been the rapid growth of Symbian, the company, and of its software engineering organization. From its small-company origins, Symbian has become a middle-sized company, established in a global market. Success, inevitably, brings its own particular slant to the perennial problems of how best to make software.

18.2 The Software Problem

Like all software companies, Symbian wrestles with the problems of efficiency, effectiveness and predictability. Every software development organization faces the same basic question: what is the right way to organize so as to be as effective as possible at making and shipping great software? In other words, how should development teams be organized (or how they should organize themselves), with respect to the software base and the need to continuously maintain, evolve and improve it? And there are softer questions too: what should the project culture be and how should it feel to work in a project?

These are not new problems and a lot has been written about them, but nonetheless they are difficult problems. One reason they are difficult is that they are ill-defined problems, not the kinds of problems with which software companies like to deal and problems with which, typically, software companies are bad at dealing. (It is always a mistake to seek engineering solutions to non-engineering problems.) They are not made easier by the fact that there is disagreement among practitioners about even the basics.

Typically, software creation is counted as an engineering practice, which is to say one based on measurable, formal procedures and systems, although not necessarily formally mathematical. (A dissident, formalist view is that it would better be classed as mathematical practice). Development methodologies therefore are engineering methodologies (process engineering and product engineering) which should deliver validated solutions to well-specified problems (customer requirements), predictably and repeatably.

There is a minority view that the very use of the word engineering in this context is a category mistake. Software creation is not an engineering discipline but something more like a craft practice, carried out by skilled professionals making intelligent, intuitive, but necessarily 'soft' or 'fuzzy' or simply underdetermined decisions. Worse, it is not just the case that requirements are often poorly specified; they are, in many cases, unable to be specified before the development activity (i.e. outside the context of a proposed solution). Development methodologies, in this view, should be designed around these basic matters of fact.

The business demands imposed by the commercial context do not help to reconcile the differences. When software companies are product companies (it is easier when they are simply internal suppliers), they are subject to the same commercial disciplines as any other business. Less orthodox approaches to requirements capture and design, based on prototyping and experimentation, trial-and-error and iteration, because they are inherently uncertain and therefore high risk, are in immediate conflict with a command and control business culture. As much at issue as methodological questions, then, are cultural and sociological ones and the underlying questions of control and where it resides in the organization.

A final difficulty of the pure engineering approach is that whatever the other merits of the argument, there is no doubt that the underlying activity of actually writing software looks as much like an art as a science. It is full of subtleties, is strictly non-deterministic, is highly context-sensitive, lends itself to multiple possible solutions, and requires experience, expertise, imagination and inspiration. These are the facts that underlie the familiar statistics about individual programmer productivity. No matter what the business needs, it simply is not possible to mandate software productivity.

If proliferation of theory is an indicator that a research area is underdetermined by the available facts, then software practice is up there with the best of them. Development methodologies proliferate and it can be hard to sort quack remedies from principled alternative practices. Theories are rapidly inflated into fully marketed, patent medicines for all software development ills. Gurus abound and none of them agree about much at all. Metaphors abound too, from 'design factories' and 'software factories' to 'total quality' and 'Sigma 6' to 'Scrum' and 'Extreme'.

18.3 Too Many Dragons

[Aho, Sethi and Ullman 1986] has a fire-breathing monster bearing the label 'Complexity of Compiler Design' on its cover. There are many dragons to slay in software development, of which the innate complexity of the endeavor is the fiercest and fieriest. Presumably, complexity is also what Stanley Lippman, the author of more than one well-regarded C++ primer, is alluding to when he chooses Durer's engraving of the knight and the devil for a frontispiece [Lippman 1996].

But almost as fierce and fiery, and certainly as famous, is the dragon 'of poor programming productivity', the problem to which [Brooks 1976] first drew attention and which, for example, [Gabriel 1996] confronts in an influential essay, without reaching any very hopeful conclusions. Estimates of programmer productivity vary from 10 to a few hundred lines per month, or perhaps 1000 to 2000 'non-commentary source lines per programmer per year.' As [Gabriel 1996, p. 127] points out, that is about four lines a day – 'There is a software crisis.'

The phrase 'software crisis' was coined as long ago as 1968,[1] and most of the current software creation infrastructure has evolved in its shadow, including the dominant operating systems, programming and modeling languages, and analysis methodologies, not to mention the modern software–hardware infrastructure.[2] All can be seen as part of the same calculated effort to move the practice of making software from a black-art to a well-founded science. The 'crisis' was created by the impossibility of reliably planning, implementing and maintaining systems beyond a certain size and complexity threshold. Interestingly, another phrase coined at the same conference, by Doug McIlroy, one of the pioneer creators of Unix, was 'software engineering'. McIlroy also observed that without some evidence of a components industry, there was no sound basis for thinking of software production even as an industry (let alone an engineering industry) [Assmann 2003].

Of course, what McIlroy had in mind was a research and engineering program that might lead to an industry founded on the manufacturing of software out of components. Since components and composition of components explicitly underlie and underwrite today's object-oriented approaches (and, in fact, most other recent programming methodology research too), it seems fair to say that McIlroy's call has been heeded. But still, the striking fact is that, close to 30 years later and what should therefore be a good way beyond the black-art and wizardry stage, the crisis seems as strong as ever. It's not so much that no solutions were

[1] At an international conference of software professionals and academics called to address the question, 'How can software be produced systematically?' [Assmann 2003, p. 6]

[2] Take your pick, but Unix first appeared in 1969; the PC in 1981; the C language in the early 1970s; C++ in the early 1980s; and Java not until 1995.

found, as that the problem has simply continued to grow exponentially. For every solution, it seems, there is an immediate test case problem for which, in one dimension or another, the solution is inadequate.

There is extensive literature on the software crisis and on the proposed solutions to it, from structured techniques to object techniques and reuse, along with a host of related approaches and methodologies, from Class Responsibilities Collaborators (CRC) cards[3] and patterns, to feature teams, to Extreme and Agile programming. Abstracting the detailed differences, the common core of the radical solutions is an emphasis on incremental and iterative development and the freeing of programmers to think, approaches which are promoted by a generally broad and probably wise consensus. But, as [Gabriel 1996] somewhat regretfully concludes, there is little evidence that the industry overall is either pleased to hear the remedies or attempts to apply them in practice on any large scale. On the whole, they do not offer solutions that a traditionally managed and typically 'over-managed and under-led' industry wants to hear. It carries on as it always has, even though eventually, as [Gabriel 1996, p. 128] puts it, 'we'll find that traditional software development methodologies are among the least productive and possibly produce the lowest quality.'

What drives the traditional software development methodologies are the basic commercial imperatives of management control and process repeatability and predictability. The problem is that these imperatives seem to be at odds with the practices which actually deliver better software productivity and quality. Perhaps software is 'just different', but if Gabriel and others are right, achieving productivity and quality seem to require non-traditional approaches to control. Traditional management doesn't want to let go.

18.4 Software Development Approaches

One argument, or perhaps it is more strictly a metaphor or another analogy, which has proved very popular is that making software is a bit like making buildings. Prefabrication and componentization are only small parts of any answer as to how to do it better, more reliably or more predictably. Following the analogy of 'habitability', making 'better' software means making software which is more elegant, more long-lived and more adaptable; on the analogy of buildings which do not fall down, reliable software does not fail unexpectedly; and predictability means completing the job in something like the predicted time.

This metaphor is at the heart of the software patterns movement, which argues that not only is making software like making buildings, but that it is an irreducibly human activity and team activity. Relationships and

[3] A good introduction is [Beck 1999].

interactions between individuals, and individual and team behaviors and all the subtleties and difficulties associated with them, come into play and must be managed by the software creation process in order for the process to succeed [Rising 1998, p. 143]. Different organizations have tried different approaches, including Extreme and Scrum[4] programming approaches and similar team-based systems, which along with ideas such as 'creative chaos' (as a tool for fostering innovation and creativity within the industrial organization) have been the subject of industrial experimentation across many different industries in Japan.[5] Teams, of course, are just groups of collaborating individuals plus leadership (teams without leaders are groups) and the health of a team is a function of the health of its individuals plus the quality of its leadership.

The common contention behind all of these approaches is that the core processes of software development cannot be understood from a purely task-based perspective. A broader perspective which explicitly makes room for the 'people' dimension, as well as mapping task outputs (for example, an 'artifacts, roles, actors, and agents' model [Rising 1998, p. 122]) is essential to understanding what is an essentially dynamic activity. Traditional process models, especially those such as ISO9000 but including also the popular Capability Maturity Model (CMM), have their place but are not perfect at capturing software development as it is actually practiced and, indeed, as it must be practiced.

Another point that these approaches all stress is that if making software is a creative process then it depends for its success (presumably) on enabling creative individuals to do the creating, and creativity is not easily prescribed. A preponderance of strong personalities among the creative people also means that command-and-control approaches are doomed to fail, if not absolutely (some software will get made) then relatively (team potential will have been poorly exploited and software quality will be poorer than it could have been).

Old-fashioned, waterfall development is very thoroughly and almost universally deprecated. Full specification followed by full design followed by full 'coding' does not work, because 'full specification' and 'full design' have both proven unachievable in practice. Iterative development, in some form or another, seems to be the only rational alternative [Rising 1998, p. 148], allowing a full specification of the problem and a fully designed solution to emerge through successive cycles of partial design and implementation. However, in practice, software organizations like most others have an almost insatiable desire for detailed specification and up-front design, not to mention planning and budgeting, ahead of any resource commitment. The organization defeats itself, because it is risk-averse and wants certainty, or relative certainty, when in fact it should

[4] 'Scrum', or 'relay', is an attempt to translate the Japanese word 'sashimi'.

[5] At Nissan, Fuji Xerox and Matsushita for example. See [Nonaka and Takeuchi 1995], especially for the theory of creative chaos.

be organizing to manage uncertainty. But to many planners, uncertainty does not sound much like engineering. It is also easy to fall into the trap of seeking to control process artifacts, rather than managing processes through the people who implement them.

Another problem is that iteration depends on short cycles, but often enough the short cycle is subverted by the planning process. The implementation phase is repeatedly postponed for the sake of a little more planning certainty. The result is not a short cycle at all, but a traditional long one (six to nine months, say), with a planning front-end which lasts for six months out of nine, followed by a hasty development tail. Inevitably the tail turns out not to be short at all, development takes as long as it takes, and the overall cycle reverts to being a one-year or 18-month cycle.

Organizational fear of 'randomness' and the indeterminate or merely underdetermined fuels the urge to centralize and control, to legislate, plan and create metrics (define and measure!). Randomness, for want of a better word, is a necessary part of the process of exploration.[6] Uncertainty, whether we like or not, is a given of creativity. The creative factory is probably an impossibility.[7]

Building construction mixes art and science, but it also has another important dimension. 'Ethical' architecture is essential, because the buildings that architects create determine our physical spaces and, done badly, despoil our towns and cities. 'Ethical software' might be something we had better start to consider too. Software increasingly permeates our lives and, in some cases, is beginning to dominate and control them, not necessarily for the good. Identity cards and 'big' databases are one example; software-driven munitions are another;[8] as is the 'Google in China' question.[9] An ethical-software manifesto, if there was such a thing, would fit well with the original goals of the software pattern movement for habitability and would fit well with the original aims of Christopher Alexander in his architectural patterns work.[10]

---

[6] Error, indeed, is objectively indistinguishable from creativity, in so far as both are underdetermined and only subjective measurement against a goal can tell one from another. To stretch the point only a little, the software on the Ariane rocket made an error, not a creative leap; but in other circumstances, departure from the norm might be called inspiration.

[7] A factory is a place where mechanized production takes place, originally organized around the principle of machine-minding ('satanic mills', for example, with steam-driven looms), re-imagined by Henry Ford and others around the idea of the production line and the factoring of the production task into simple actions performed by highly specialized but unskilled labor. A factory has only lately been reinvented as a place where teams work together to meet their production targets.

[8] Mobile phones are not ethically neutral. Mobile phone access to emergency and rescue services saves lives. But equally, mobile phones are increasingly being used to trigger bombs and as homing devices for remotely delivered munitions to target individuals.

[9] How can we achieve the 'borderless Internet' when it runs up against state censorship?

[10] The classic text is [Alexander 1979].

In some senses, the open-source and free-software movement is an ethical movement, but in other senses it is more obviously mercantile and market-led and markets know no ethics. But the hacker ethic (see [Himanen *et al.* 2002]) proposes a rather different work ethic from the conventional one: work is fun, programming is play and the passion for the machine has a moral dimension.

18.5 What Making Software Is Really About

'Shipping great software on time', as [McCarthy 1995, p. 2] puts it, is what making software is all about. But as he emphasizes, software is peculiarly intangible. It is not simply 'stuff'; it is embodied thought, idea plus design plus implementation; each of which is an intellectual (not mechanical) process. What makes the process interestingly different from other intellectual processes (thinking, writing and painting) is that beyond a certain level of size and complexity, it is necessarily a team activity.

For McCarthy at least, the word 'development' (as in 'software development') is a clue to the nature of the enterprise, a dynamic process of maturation, in which what matures is precisely that embodied thought: as he writes in [McCarthy 1995, p. 85], 'It's the team ideation, gradually migrating from highly individual (even private) notions toward a group articulation in the shipping code.' This way of putting it will resonate with anyone who has been involved in the software development process (and that means, as he says, 'everybody on the team', whether planning, scheduling, creating or validating the software product). The essential development act is this development of individual ideas into embodied intellect, productized thought: intellectual property, in other words.

This is the view, of course, that sees making software not as an engineering process (although there are engineering aspects to it, as there are in any construction process) but 'as primarily a sociological or cultural phenomenon' [McCarthy 1995, p. 87]. Perhaps more even than engineering 'aspects', there are some engineering fundamentals involved. Machines are engineered constructs and all software is in some sense 'soft machine'. But, nonetheless, if the software industry is fundamentally a creative industry, then there are necessary limits to how far industrialization (and even formalization) can go. Every developer is familiar with the notion of the 'death march'[11] and it is hard to imagine that anyone would willingly adopt it as a project model. But without a development methodology that understands, and serves to support, the reality of the software creation process, it is probably the inevitable end for all software projects.

[11] The 'death march' is the bringing to final completion of a long and difficult project and its repeated slippage as an 'exhausted, physically and emotionally spent' team marches, stumbles, and lurches to final shipping of the product [McCarthy 1995, p. 33].

Software is not the only creative industry to have attempted a 'factory' approach. Hollywood is probably the most famous, if not the most obvious, example. Thus at Warner Brothers studios in the 1930s (see [Bordwell *et al.* 1985, p. 326]), all creative people (directors and writers included) had set working hours and a per-day expected piece-rate production (which may not in the end have been so much more than the equivalent of four lines of code per day). In one sense, it worked and the films are there to prove it, but it almost certainly did not work for the simplistic reasons that Jack Warner thought that it worked. More likely, it worked because a hot-house of talented people did good things despite the more foolish aspects of the regime.[12] In an updated context, it might also be legitimate to ask whether the independent cinema sector produces higher quality films than the modern Hollywood factory (surely it does!) and, if so, how much of the difference is due to their different production models. The analogous comparison between conventionally and open-source produced software might be similarly instructive and for similar reasons.

Software development, says [McCarthy 1995, p. 87], is more like a jam session than an orchestrated event and, in so far as that is true, it cannot be factoryized and cannot be scripted. The jam session requires motivated, independent, skilled, autonomous individuals to work together with a common aim. What the development model can enable is a team context which makes it likely that the goal will be achieved, including time goals if time-boundedness is part of the project specification.

Flavor of the Times

The stereotype of the early days of the Protea project which created the precursor to Symbian OS is, on the one hand, Charles Davies and his elite band of application and 'middleware' programmers with their object-oriented cloud and, on the other, Colly Myers and his lieutenants with their heads down in the practicalities of leaner, meaner and faster code, only vaguely aware of the cloud but exploring some interesting object-oriented ideas of their own. Somewhere between the two elites is David Wood, throwing new recruits at the interesting problems of creating a complete object-oriented GUI from the ground up and laboring mightily

[12] According to legend, Jack Warner was accustomed to taking a mid-afternoon stroll around the studio lot every day. If when he passed the Writers' Building there were any offices from which he did not hear the sound of a typewriter from the window, he would have his assistant enquire why the writers in question had not been writing.

(and often singlehandedly) into the night to integrate the raw code they created into a complete and elegant system. Bill Batchelor stalks the corridors with a furiously scribbled and rescribbled back-of-envelope project plan looking distracted. In the basement, meanwhile, Richard Harrison presides over a rack of home-made-looking black boxes (ROM writers) and a heap of yellow, later green, 'Banana' and 'Lime' Psion Series 5 prototype devices.

Davies, according to the stereotype, is the cerebral purist and Myers the bulldog-like pragmatist, the one who is actually building the system and needs therefore to know what to build. Bollen meanwhile, as a relative newcomer, was neither exactly in Myers's corner nor in Davies's corner.

Geert Bollen:

I was 'piggy in the middle'. I had arguments with both.

Others who were there tell a broadly similar story. Most strikingly, for all its informality, it was not a particularly relaxed environment. As Martin Tasker puts it, it was a frontier atmosphere, exciting, charged but driven.

Martin Tasker:

Colly Myers and Charles Davies led the project, and they had three trusted lieutenants who they put in key positions, namely Nick Healey, David Wood and Bill Batchelor, but Colly and Charles were formally Psion Group directors and they directed the software on the project. They had absolutely poles apart different management styles. With Colly, everything came from within and was asserted. His style was to assert everything more or less out of his own brain and his own experience, you know, an 'I've already written three OSs, this is my fourth and I know what I'm doing' kind of mentality. And people who followed that had an easy time, and people who thought otherwise had a hard time, which didn't mean to say that Colly couldn't be challenged, but it was difficult.

Whereas Charles's style was very much to read books about other projects and basically to design the interfaces, and then to train up people and get them to implement stuff to the specification, and then he would iterate. He had a group of people who he basically taught Rose clouds to and he architected iteratively with them. I mean Charles was quite assertive but he had broader input than Colly and was gentler in the way he let things out as well. And David Wood, meanwhile, although he was slightly under Charles's guidance, he gave a huge degree of autonomy to his people, basically he used to say, 'these guys are bright and they can do it, I won't give them any brief at

all, I'll just comment on what they've done when I integrate it', whereas the people who worked directly for Charles got quite a lot of attention at a certain interface level.

There were a lot of different styles, and I would say overall there was a lot of creative tension around. I think the atmosphere was really very frontier. There were probably people who didn't have the experience to deliver on their potential, for whom that environment wasn't the right environment. And there was also a class of people who kind of came in and always wore a white shirt and would have been a lot happier with Symbian as it is today than Psion as it was then. You know, the frontier mentality didn't suit.

There was an architecture committee, ArchComm, and Bollen was on it, as were Davies and Wood. There was also a UI design committee, presided over by Nick Healey and Bill Batchelor, the two UI gurus, dedicated to keeping down the number of key strokes required to drive the system in keyboard-shortcut modes and to keeping the application UIs clean and consistent.

Howard Price:

There was a lot of counting of key presses. Navigating with the keyboard, you want to avoid having more than the strictly minimum number of key-presses.

Peter Jackson, who had joined the company well before the Protea project began and was by then almost an old hand, also has another, quite specific recollection of the transition from the early design activity to the construction of a real system.

Peter Jackson:

I have a very clear recollection about what life in Psion was like at the end of 1994. There was a lot of quite deep thought going on about architecture and the way things should be modeled and so on, and then one day Colly made his world-changing submission. He heroically wrote all this code and arrived with an implicit statement of, 'Here we are, here's EPOC32'. And then all hell broke loose, because basically all this deep thought stopped and everyone started coding.

You can't argue against it, because time to market is everything. But at the same time, what Colly catalyzed also included some poorly considered design. At the same time, we were recruiting a lot of people and letting them loose on difficult problems. One example was David Wood assigning some relatively new people to write a menu system which would include things like drop-down-menu technology.

I remember that I had spent a lot of time, because of the lack of documentation in SIBO, reverse-engineering the design of the menu system to implement on the MC400 laptop, so I knew how difficult it was to actually get it right. And all these people were kind of fresh out of university! David Wood would collate all their output, and he would make a huge effort to put it all together to produce a user-interface system. And the good thing about it was that we made progress and it cheered people up. The bad thing about it was we had to throw a lot of it away that wasn't right.

Lotus Notes had been adopted by the company quite early on and provided an important part of the mechanism which allowed strongly decentralized team-working without completely abandoning control of the high-level design. The result was a kind of hub-and-spokes model, in which strong design directions were transmitted from the hub to the individual teams at the end of the spokes but without much direct communication between the different teams. But the Lotus Notes culture enabled a lot of direct communication between individual developers, through the medium of the databases.

Howard Price:

Design was pretty much a central activity, though you didn't get people showing their Rose diagrams to each other especially. I remember going to Charles Davies with my design for review just once. My design was only really seen then and the rest of it was up to me. But there was a design database which was very active. Charles Davies had a central Rose design on the network where each team was supposed to design their own Rose diagrams that fitted into it and you could link them together.

But there was a way you did things, definitely. There was a strong Notes culture. There would be a lot of activity on databases discussing ways to do things and it would be hard to miss the right way to do it, at least if you were coding something standard anyway. So there was some very interactive development discussion on the databases.

In fact, for a system in which in-house idioms were important (for example, the idioms of descriptors and active objects, client–server and framework–plug-ins), Notes was an essential mechanism for providing guidance on how to use the idioms effectively and, indeed, for enforcing their proper use.

While high-level design was always to a large extent autonomous, although beneath the eye of an overall Architecture Committee, coding standards were always strongly defined (they had to be, in a company which was getting to grips with a new language, for which many engineers had only had the brief exposure of a formal training course and no hands-on experience).

Howard Price:

Client–server, for example, was treated as something that everybody had to know. The way it worked was quite particular, the way you package and pass data across the server boundary. You couldn't simply invent your own way of doing that, you had to stick to the right ways of doing it and it was enforced. If you hadn't done it before, you had to find out how to do it.

The mechanism was very much developer to developer, either within teams by direct coaching and code review, or directly between individual developers using Lotus Notes as a universal read–write discussion medium. The Lotus Notes culture was always extremely strong (and remains so, although to some extent it has been diluted more recently by the scale of expansion of the company and by new ways, for example Wiki, leaking in).[13] But among the older guard, it remains strong.

Culture, Culture, Culture

The strong culture of autonomy inherited by Symbian dates back to Psion's beginnings as a small company of highly technically capable individuals, with not much hierarchy but with highly visible leaders, in which everyone contributed to make things work. In particular the company was engineer-led, with a deep culture of engineer-led design, pulled together by a centralized but loose design process.

Peter Jackson:

Culture, not process, is what is important. And it really was the case that everyone was given permission to do what it took.

In many respects the early engineering culture in Symbian seems close to what has more recently (and fashionably) become formalized as 'agile' programming: strongly team-based, strongly team-meeting- and review-based, strongly decentralized, highly localized and organized around clear goals. There was a strong concept of team ownership of code and, indeed, not just of code but more strongly of ownership of the design and implementation of a well-defined, discrete piece of the system.

[13] Wiki also has great potential to give the culture of collaboration a new lease of life and take it in new directions.

Howard Price:

I think we were Agile, but we didn't have the culture of daily meetings Agile recommends, though we did have weekly meetings. For instance, I think it was for the Series 3, Charles Davies would organize a weekly meeting and all the team leads would go in, and the meeting would go round the table and you'd say how you were doing and maybe Charles would decide that some team should implement a certain feature or that some other team shouldn't implement some other feature. And everyone would agree, but maybe that team would go and do it anyway, because they thought it was the right thing to do. So you had a lot of ownership of your own code, which was good, and you could decide largely what you wanted to do. However, you had to answer for what you'd done if it turned out to be a bad decision.

Charles Davies was very much the driver of the culture of design, a believer in design if only because, as he puts it, 'It helps you be dumb'.

Charles Davies:

So I'm a believer in design, which I tried to promote using design tools. It wasn't UML, it was Rose at that time, so I was maintaining Rose diagrams. I think you do have to have a design ethic. If you just end up putting code where you happen to have the editor open... well, that's a bit too harsh... but you do need to take grasp of it and keep simplifying it. I believe you're a much better programmer if you're a bit forgetful or can't remember things easily, because then you have to simplify until anyone can remember how it's done. I'm a great proponent of having design idioms so that you can recognize designs. You can see that there is a design and that it makes sense.

Martin Tasker thinks the design culture was particularly important in enabling an informal culture nevertheless to be particularly effective.

Martin Tasker:

Charles Davies led the design and trained people who had been in related disciplines – he got them to do object-oriented software. In any management view there are massive benefits out there if you take the enabling steps to achieve them and Charles did this extremely consistently and well. Charles, in particular, paid minute attention to the details of his APIs. He used their explanatory power to motivate his people and he almost didn't need to look at what they produced in terms of implementation

code or test code. He basically believed that if the code met the requirements of the API and if you felt it was correct then he trusted you that it was correct. Charles used that to massively good effect. He used object orientation as a means of controlling a sea of junior programmers very successfully.

Perhaps of all teams in the company, the Base team (which developed the kernel and low-level systems) had the capacity to survive longest before compromising the quite particular culture which had evolved: informal, devolved, expert and committed, attributes which it retained certainly well into the Symbian OS v6 release projects and beyond into Symbian OS v7. Morgan Henry was working in the Base team on the original port of EPOC32 to Nokia's new 9210 Communicator hardware and what eventually became Symbian OS v6.0.

Morgan Henry:

Certainly in the Base team there was a mentality that you get the best people you can find, people who are interested and excited by the hardware and software interaction, and broadly you let them get on with it and they come up with good code. And it was led by people who were interested in the technology and enjoyed what it was doing. I think they understood that if there's enough space there, that if you allow talented people to do the things they're interested in, you end up with a good quality product. That's not to say there were no processes and no project management, but, that it's all about the balance.

Transition of the Development Model

Formalization of the software development model was probably an inevitable consequence of success and growth and, in particular, of the evolution from a software company making a software product for a single customer (and, to all intents and purposes, a customer within the same company), to one which was suddenly faced with multiple licensees, all competing in a high-growth and relatively new, global consumer market. What's more, compared with Psion these companies were industry giants, the likes of Nokia, Ericsson, Motorola and Matsushita. They had established markets, established practices, global reach, and strong (and quite different) internal cultures. The small-scale and home-grown practices which had evolved within Psion and carried over into Symbian were suddenly confronted by a very different external reality.

Certainly the reality is that the company now is very different to the one it used to be. Having grown by several multiples, the approach to software creation is inevitably very different.

Morgan Henry:

Even during the process of Psion Software becoming Symbian, there were many attempts to regularize development processes. The Base team was always a bit different, but now it seems very much like the technology architects have less freedom to make the architectural choices that they need to make, and they don't get the time to write any code or do any proper design. So Symbian as a company has begun to regulate more, which is no bad thing, because you have to have reliability of delivery over everything else if that's what your customer is asking for. Back in the Nokia Communicator and Psion Series 5 days, we were a technology-driven company, and this transition to a marketing-led company has probably contributed to the difference.

The most immediate loss is probably that of developer autonomy or, at least, the sense of autonomy.

Morgan Henry:

With the Nokia Communicator, you'd see a problem and you'd be responsible for it, up to the point where you'd be responsible for going to see Nokia. For instance, I'd be sent out to Finland and I'd be put in a room with ten Nokia engineers and they'd ask lots of questions and it would be my responsibility to come up with a solution, and possibly even go back and implement the solution.

There was no project manager doing it for you, it was a case of having to manage your time on that and work out when it was going to be done, which meant you had much more freedom and you had much more contact with the hardware. So I was prepared to put in a lot of time and a lot of effort, doing long nights and weekends if I needed to, just to make sure that it worked, because you cared about it and you wanted to see it ship.

As Symbian has evolved, driven by the need to scale to many more licensees and to support a multitude of licensee projects, responsibility for porting to new hardware has moved away from the core engineering teams to a dedicated porting team and dedicated licensee support teams. Inevitable and necessary as it may be, one consequence is the distancing of the core software-engineering teams from real mobile-phone products.

Morgan Henry:

The Base team has been detached from the base porting activity and delivering a base port for a 'real' product. At best they're delivering a base port for a hardware reference platform. There are many reasons for this, but certainly the desire to regularize and the desire to control the time investments are part of it. The fact is that they are less in contact with the final product.

Having said that, there are some parts of the organization which still have a close relationship with the final product and they still do the long weekends and they still go and see the customer and get bombarded with questions, so that's still there. I'd argue it's probably the best way of doing that kind of job, as opposed to the process-driven activity that is common in Symbian.

While the Lotus Notes database and discussion culture remains strong, the design culture that used to underlie it has been dissipated as the company has grown.

Howard Price:

The Lotus Notes culture has survived to a fair extent and there's a lot of discussion of how to do things. However, there doesn't seem to be huge amounts of design discussion going on, it's more programming discussion. Maybe that's because there's a feeling that everything is stable and there's not really that much new design and big new design now goes through the System Design Authority.

There is no doubt that the design culture is very different in Symbian now from what it was in the early days.

Charles Davies:

Do we do enough design now? Of course programmers like doing design, but it's whether you do design as an explicit activity. I'm a believer in modeling. I'm not up to the UML standards these days but I've found UML useful. But even if you just draw it on flipcharts, you do need a design. Design gets you to reusable patterns, so if you've got a community of people doing design it's very useful, and if you leave a programmer on their own they won't get there. You need a culture of design, of people sharing designs and talking about it. So I did used to do that. Design, ultimately, is the thing that cuts your defect cost down, because good designs have fewer defects. But then of course you need good requirements. It's all 'Page One' of the software-engineering manual. So we know that, but it's not just knowing it, it's actually doing it.

Control and flexibility are not easy to reconcile. The deadline pressures of time-to-market inevitably conflict with the need to refactor and redesign as part of the daily software development approach. David Wood still considers iterative design an essential part of any successful development style.

David Wood:

I do think it's worthwhile having enough project time set aside to allow for some degree of refactoring of designs, because you can never be sure of getting it right first time. Even with the cleverest people, the most experienced people, there are things that only become clear as the design evolves, so you should, if possible, allow yourselves the freedom to put that into future products. It gets tough when you've got to maintain compatibility as well, so it may be that for a while you've got to run the two systems in parallel, you've got to run the old system for older products on new machines, but then also you can have the new APIs which gradually people can switch to.

Size tends to work against flexibility, but in fact flexibility is not the only thing that gets lost with increased size. Increased size also works against the consistency and design elegance of the system, tending to cause dilution. Peter Jackson argues that design dilution is an almost inevitable consequence of a traditionally devolved approach to design confronted by rapid expansion.

Peter Jackson:

A general problem that has been faced as Symbian OS development teams have grown is the varied experience of the people you employ to work on the operating system. If you try to do something new, if you try and invent a sophisticated locale system, the problem is that somebody from outside who may have, say, a Unix background, doesn't understand the design principles of what you're doing and may get it wrong or may take longer to learn what you're up to.

Jackson's own area of specialization in the system in the early days, internationalization and localization, is a case in point.

Peter Jackson:

The problem with internationalization is that typical programmers like to believe that everybody is American! They can ignore basic issues, like for example sometimes you can't assume that pluralizing a word means putting an 's' on the end of it. So we were always compromising saying, 'this would be the elegant thing to do, but everybody expects it to happen a different way', and we have to do it the way everyone expects. It is a fact of life working in this industry, that software tends to devolve towards the lowest common denominator rather than towards the most elegant thing.

'Worse is better' is the label Richard Gabriel coined to describe the problem that inferior systems or designs tend to beat superior ones in the

market by getting to the market quicker and occupying the market niche with a system that is just good enough to make the switching cost to the better system unattractive.

Peter Jackson:

It's probably a particular case of the 'Worse Is Better' thesis, which basically says, 'You can put a lot of effort into doing the right thing at all times, but meanwhile your competitor will have ignored all the difficult cases and got to the market first, and the difficult cases happen so rarely that people can be conned into accepting an inferior product.' That's a paraphrase of Worse Is Better, but that's how I think of it.

It is a truism that success can be a dangerous thing in its own ways. Success for Symbian has meant near continuous growth and expansion.

Peter Jackson:

The software development process determines your success in producing software. I'm thinking of things like the role of testing in software development, the question of where defect-fixing comes at the end. Issues like these are cultural issues. You want to institutionalize something that's good, the right thing, and I don't think it's easy at all, because as soon as you have started doing the right thing, you lose focus and you change the focus to something else. Meanwhile, the company grows a bit more and suddenly the thing you thought was taken care of isn't taken care of at all. So it's quite hard even to work out what you have to do to institutionalize the things you care about.

But the biggest impact of growth is in a way the most obvious one, if also the most puzzling one. With responsibility for managing the code repository within the integration and build organization within the company, Jackson is well placed to observe it.

Peter Jackson:

The number of submissions into the repository is enormous. You watch it, and it just scrolls up the whole time, you just see it, submission after submission after submission. And yet we produced the first version of EPOC with an engineering community of closer to 100, it was of that order of size. So sometimes you think, What is all this work going on? What is it? Why do we need so much of it? So looking back, the scale really is quite different.

The Value of 'Whole Product' Development

Charles Davies identifies a particular problem which has become highly relevant for the company, and which has deep roots not so much in the details of the development model, but in the wider business model and the place in which Symbian finds itself in the market. The evolution from being a product company to being a pure software company (and indeed, a pure operating-system company) tends to work against the holistic, whole-product understanding of what the company is producing, which makes validation much more complex. In the worst case, focus on validation can be lost altogether.

Charles Davies:

In our context, validation is hard. It used to be much easier in Psion, because we would provide an API with our OS hat on, and then applications would use it in the same organization, and then the device would use the application. If it wasn't any good, people would say so! And the application is working this way because the API works this way, so we'd change the API to work better. That cycle doesn't exist now. We deliver APIs to our customers, and there isn't that natural process of validation.

Pick up a book on software engineering. All of them say that verification means, in software terms, that you've met the specification, and validation means that the specification that you agreed with the customer was actually useful in the event. And we used to do that in Psion without realizing it, because it was part of being a product company. Now we have become too remote from the final product for that.

The problem is that Symbian, by the nature of its business model, is at least one step removed from the true product cycle of making and shipping phones.

Charles Davies:

So, we've produced an API! It's in the release and the job is finished. But the job is really only finished when the value has been delivered, not when the API has been delivered, and that's what validation means. It means a customer conversation to say, 'Was that valuable? Did it do it for you?' If the API wasn't used, the answer is, 'No!'

In other words, creating the OS is not just a matter of delivering software that works and delivering well-designed, well-abstracted and well-implemented APIs. It is also a matter of delivering the right APIs to customers, at the right time, and validating them with the customer. In a sense the problem is the familiar one of the need for iteration, to

enable validation and where necessary refactoring and redesign. And the difficulty is successfully creating a project model which delivers that, while meeting the other needs of supplying into the global product manufacturing businesses of phone-vendor licensees.

Charles Davies:

For any new API, it would be remarkable if it was right first time. It is guaranteed to be wrong for any non-trivial API. So you should expect in the normal course of events that for a new API, you'll have defects at the requirements level and you have to rely on that iteration. For a new API, you need to factor in the extra work you get from a process that does validation.

It's all too easy, instead, simply to finish the API and ship it.

Charles Davies:

Imagine you ship an API, and that's it, you then move on to work on something else. What if the customer says it doesn't meet their needs? What do you do? Ignore it? In Psion they would have said, 'This API doesn't work and I'm not using this unless you fix it!' And so it would be fixed and the APIs got good because of that. And that's validation!

This is an inherent problem of the business model and one that exercises many people in the company and one on which the company is continually striving to do better.

Putting the Magic Back

The goal of a successful development methodology must be to reconcile the inevitable conflicts between the different demands of the business. Performing that balancing act is an essentially dynamic activity and it is hard to build dynamic behavior into organizations. Effective software development, as if we didn't know it already, turns out to be a hard problem to solve. It may be that there is no such thing as a software company above a critical size threshold that can get it right. Keith de Mendonca certainly believes that it is hard, but not impossible, to scale up without losing agility. Successfully scaling up probably requires a broader perspective than just that of the development methodology and raises the larger question of the way the development organization is structured.

Keith de Mendonca:

Agility is much bandied around as a hot topic, but certainly the attributes it should represent are that you are close to your customer, you understand better what you are doing, you are continuously improving, you are basically making the best decisions that you can at each stage, and being able to take each step and improve until you deliver the best version of the code to your customer.

Organizationally, we are committed to this concept and that motivates the way the software engineering organization is aligned and organized into technology streams today. We consciously said we wanted to create the organization around the technology architecture rather than just managerial units. And although we maybe haven't achieved it yet, there was meant to be a much greater degree of autonomy inside the technology streams, where they own their own roadmaps. So effectively if they were given clear instructions about what they needed to do by when, they would have much more autonomy to actually do those things in the best way that they see fit.

The key ideas are to align around a technology vision of where the product should be going and to devolve responsibility for delivering the vision to technology-focused engineering streams.

Keith de Mendonca:

A company which is still growing and still maturing must try to control itself, while market forces constantly challenge that autonomy. The struggle is to provide that flexibility and allow that flexibility to the technology streams when external issues often arise that cause changes in direction. Naturally, this makes an organization wish to have very tight control of what each component is actually doing.

The forces of control and centralization are almost inevitable in the face of commercial pressures – without which of course there would be no system at all.

Keith de Mendonca:

Of course, success in a new market needs many changes in the way that a software organization is controlled. Symbian has control of its own destiny in engineering terms, but it will take time to balance that autonomy and responsibility on the smaller units with the execution and the very predictable delivery that we demand as a company.

But if you look at the high level, we release the OS to our customers every two weeks now. We also have regular official operating system releases (about three a year compared to the original model of one huge release every 18 months). The shorter release cycle delivers regular improvements to the product and gets faster feedback from the customer.

Possibly there is no way back to the kind of flexible and autonomous development model of the early days of the company. At the end of the EPOC Release 5 project, 48 or so senior developers including architects and team leads spent two days at an off-site debriefing with the company management. Management had mapped out a two-day program, but on the first morning it was overturned by the engineering participants, who had a different view of where the obstacles to progress were and what should be done about them. It is unlikely that such a thing could happen now – if only because the likelihood of engineers and managers having sufficient time to spend two days out at the same debriefing session is remote.

Keith de Mendonca:

I think that physically the tools needed for controlling a large organization tend to automatically restrain that kind of flexibility and fast movement. But to some extent you can build agility into even medium- or large-sized companies, and I suppose that's what we are currently planning to do, to put that agility back.

It may also be that perhaps the company was never really a 'creative zoo' at all.

Bob Dewolf:

Development methodology is a really tricky issue we have. But creative zoo, is that what we used to have? I don't know. I don't think so.

Symbian is practiced at innovation. The nature of the development model and software engineering organization remain open questions. Watch this space.

Appendix A: Symbian OS Component Reference

Introduction

This appendix attempts to provide a definitive component reference for Symbian OS, based on Symbian OS v9.3, the most current available release at the time of publication. Although associated with a specific release, for the most part the component information will be useful for developers working on any release.

The goal is to make available to external developers (including those working with licensee or partner companies, as well as independent third-party developers), a minimum level of information about the system at component level.

Note that the component set should be interpreted as a superset of all possible components, and not as a definitive guide to the components present on a given Symbian OS device.

A.1 Explanation of Fields

Each component is documented in a simple format. The meaning of the fields is explained below.

Development Name

The development name is the short name by which the component is commonly known. For example, the Plug-In Framework component is commonly known as ECOM; the Data Comms Server is commonly known

as C32, and so on. Typically, the short name is either identical to or a derivation from the name of the binary that the component builds (where it builds to a single binary), the principal significant binary (where it builds several binaries), or the name of the directory in which the component source is located in the source tree.

Build File Location

The build file location is the path at which the build file for the component can be found in the OS source tree. Note that external developers do not have access to the full source tree.

System Model Location

Where the component appears in the System Model, its location is given as *Layer* (in all cases), *Block* (where applicable), *Sub-block* (where applicable), and *Component collection* (in all cases).

Licence Categorization

In the license that defines the terms of use of Symbian OS by licensees, all components intended for production deployment are categorized as either *Common* or *Optional*, and as either *Symbian* or *Replaceable* components, thus creating four possible categories. Components which are not intended for production deployment are categorized as *Reference* or *Test* components.

From an external developer perspective, the most meaningful interpretation of these categories is as follows:

	Symbian	**Replaceable**
Common	Always present in a vendor platform. Symbian supplied, defines Symbian OS APIs.	Always present in a vendor platform. Symbian or licensee supplied, preserves (but may extend) the Symbian OS APIs.
Optional	Optionally present in a vendor platform. Symbian supplied, if present, defines Symbian OS APIs.	Optionally present in a vendor platform. Symbian or licensee supplied, if present. Not guaranteed to preserve the Symbian OS APIs.

Reference and Test components are not intended for production deployment, although licensees may choose to deploy reference components. Test components must not be deployed on licensee devices.

Exposes Third-Party APIs

If this is YES, the component exposes public APIs which any developer can use. They are classified as `publishedAll` in the source code. If it is NO, the APIs are restricted.

Restricted APIs are classified as `publishedPartner`, `internalTechnology` or `internalAll`. Although partner APIs are not supported in public SDKs, they are supported in the kits shipped to partner developers and can be used by them. Internal APIs are reserved for use by Symbian development teams.

It is important to note that header files available either in kits or SDKs may contain mixed categories of APIs. The classification rules are intended to help guide developers towards the APIs they can safely use.

Present in OS Releases

Data is provided for all releases from Symbian OS v7.0 to v9.3 as an aid to developers migrating code from earlier releases.

Description

This is a brief plain text description of the functionality provided by the component and its role in the system.

A.2 Agenda Model

Development Name: AGNMODEL

Build File Location: `/common/generic/app-engines/agnmodel/group/`

System Model Location:

Layer: Application Services

Component collection: PIM Application Services

License Classification: Optional Symbian

Exposes Third-Party APIs: YES

Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: Public APIs are limited to enumerations in `calnotification.h`. Legacy API, replaced in Symbian OS v9 with the new Calendar API which is more suitable for a phone. Agenda Model is maintained for compatibility.

A.3 Alarm Server

Development Name: ALARMSERVER
Build File Location: `/common/generic/app-services/alarmserver/group/`
System Model Location:
Layer: Application Services
Component collection: PIM Application Support
License Classification: Optional Replaceable
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Server that manages a queue of system-wide, time-based alarms that provide APIs for client applications to set, modify, query and notify alarms.

A.4 Animation

Development Name: ANIMATION
Build File Location: `/common/generic/app-framework/animation/group/`
System Model Location:
Layer: UI Framework
Component collection: UI Support
License Classification: Optional Replaceable
Exposes Third-Party APIs: YES
Present in OS Releases: 9.1, 9.2, 9.3
Description: Framework that supports window-, sprite-, and bitmap-based animation, enabling animation plug-ins to be created and loaded.

A.5 Application Architecture

Development Name: APPARC
Build File Location: `/common/generic/app-framework/apparc/group/`
System Model Location:
Layer: Application Services
Component collection: Application Framework
License Classification: Common Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: Framework that defines basic application responsibilities and the interactions between core application classes, broadly following the MVC pattern. Abstracted via Uikon, and ultimately by a vendor-specific variant UI.

A.6 Application Utilities

Development Name: BAFL
Build File Location: `/common/generic/syslibs/bafl/group/`
System Model Location:
Layer: Base Services
Component collection: Low-Level Libraries and Frameworks
License Classification: Common Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Essential application utilities organized as a single library DLL. Includes system sounds, the clipboard, utility classes for resource-file handling and file finding, and implementations of string pools and descriptor arrays.

A.7 Audio Driver

Development Name: SOUNDDEV
Build File Location: `/common/generic/Multimedia/MMF/sounddev/group/`
System Model Location:
Layer: Kernel Services and Hardware Interface
Block: Kernel Architecture
Component collection: Logical Device Drivers
License Classification: Common Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: Hardware-abstraction layer for digital audio acceleration.

A.8 Backup and Restore Notification

Development Name: BACKUPRESTORENOTIFICATION
Build File Location: `/common/generic/app-services/BackupRestoreNotification/group/`
System Model Location:
Layer: Application Services

Component collection: PIM Application Support

License Classification: Optional Replaceable

Exposes Third-Party APIs: NO

Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3

Description: Intended for internal use only. An alert service to notify backup–restore progress to PIM applications. All other applications should use the Publish and Subscribe service to achieve similar functionality.

A.9 Baseband Channel Adaptor

Development Name: BCA

Build File Location: `/common/generic/networking/BasebandAdaptation/bca/group/`

System Model Location:

Layer: OS Services

Block: Comms Services

Sub-block: Networking Services

Component collection: Link Layer Control

License Classification: Optional Replaceable

Exposes Third-Party APIs: NO

Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3

Description: Abstracts the channel used to communicate with the baseband processor and defines a plug-in interface for a hardware-specific interface-implementation module.

A.10 Baseband Channel Adaptor for C32

Development Name: C32BCA

Build File Location: `/common/generic/networking/BasebandAdaptation/c32bca/group/`

System Model Location:

Layer: OS Services

Block: Comms Services

Sub-block: Networking Services

Component collection: Link Layer Control

License Classification: Optional Replaceable

Exposes Third-Party APIs: NO

Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3

Description: Intended for partner use only. Plug-in providing a serial comms implementation of the Baseband Channel Adapter interface.

A.11 Bearer Abstraction Layer

Development Name: BALSERVER MROUTER-PLUGIN

Build File Location: `/common/generic/connectivity/BAL/group/`

System Model Location:

Layer: OS Services

Block: Connectivity Services

Component collection: Device Connection

License Classification: Optional Replaceable

Exposes Third-Party APIs: NO

Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3

Description: Intended for internal use only. Abstraction framework for connectivity plug-ins that encapsulate actual bearers (e.g., m-Router), providing a connection management API for use by PC link-type applications.

A.12 BIO Messaging Framework

Development Name: MSG_BIOMSG

Build File Location: `/common/generic/messaging/biomsg/group/`

System Model Location:

Layer: Application Services

Component collection: Messaging Application Support

License Classification: Optional Symbian

Exposes Third-Party APIs: YES

Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: Messaging extension supporting 'smart' message types (Bearer Independent Objects), for example vCard or vCalendar messages, network setup messages and so on, which are not intended for end-user action but for system components or applications. Provides a mechanism for creating application-specific, bespoke message types.

A.13 BIO Messaging Parsers

Development Name: CBCP, ENP, GFP, IACP, WAPP

Build File Location: `/common/generic/messaging/biomsg/group/`

System Model Location:

Layer: Application Services

Component collection: Content Handling

License Classification: Optional Symbian

Exposes Third-Party APIs: YES

Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3

Description: Plug-ins to the BIO Messaging Framework for parsing specific message types including compact business card, email notification, Nokia Smart Message Internet Access, Nokia and Ericsson OTA, as well as a general file parser.

A.14 BIO Watchers

Development Name: BIOMSG, NBSWATCHER, WAPWATCHER, BIOWATCHERSCDMA

Build File Location: `/common/generic/messaging/biomsg/BioWatchers/bld.inf`, `/common/generic/messaging/biomsg/BioWatchersCdma/group/`

System Model Location:

Layer: Application Services

Component collection: Messaging Application Support

License Classification: Optional Symbian

Exposes Third-Party APIs: NO

Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3

Description: Watcher framework and service for notifying applications of message arrival.

A.15 Bit GDI

Development Name: BITGDI

Build File Location: `/common/generic/graphics/bitgdi/group/`

System Model Location:

Layer: OS Services

Block: Multimedia and Graphics Services

Component collection: Graphics and Printing Services

License Classification: Common Symbian

Exposes Third-Party APIs: YES

Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: Device- and display-mode-independent implementation of the concrete graphics context for bitmaps.

A.16 Bluetooth 1.0

Development Name: BLUETOOTH
Build File Location: `/common/generic/ME/group/`
System Model Location:
Layer: Java ME
Component collection: MIDP 2.0 Packages
License Classification: Optional Replaceable
Exposes Third-Party APIs: YES
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: MIDP 2.0 Bluetooth 1.0 (JSR-082) APIs that support Bluetooth messaging including Push support.

A.17 Bluetooth 1.0 Push Plug-in

Development Name: BLUETOOTH
Build File Location: `/common/generic/ME/group/`
System Model Location:
Layer: Java ME
Component collection: Bluetooth and SMS Push
License Classification: Optional Replaceable
Exposes Third-Party APIs: YES
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: Forms part of JTWI. Plug-in that binds the Bluetooth 1.0 package to the underlying system.

A.18 Bluetooth CSY

Development Name: BTCOMM
Build File Location: `/common/generic/bluetooth/`
System Model Location:
Layer: OS Services
Block: Comms Services
Sub-block: Short Link Services
Component collection: Serial Comms Server Plug-ins
License Classification: Common Replaceable
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: CSY plug-in to C32 Serial Server providing serial port emulation over Bluetooth.

A.19 Bluetooth HCI

Development Name: HCI
Build File Location: `/common/generic/bluetooth/`
System Model Location:
Layer: OS Services
Block: Comms Services
Sub-block: Short Link Services
Component collection: Short Link Protocol Plug-ins
License Classification: Reference/Test
Exposes Third-Party APIs: YES
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: Host Controller Interface (HCI) firmware driver implementation for reference only, so must be replaced by the licensee. The HCI framework, for example, includes public APIs.

A.20 Bluetooth Manager

Development Name: BLUETOOTH_MANAGER
Build File Location: `/common/generic/bluetooth/`
System Model Location:
Layer: OS Services
Block: Comms Services
Sub-block: Short Link Services
Component collection: Short Link
License Classification: Optional Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Information store for managing details of the local and remote Blueooth devices, implemented over DBMS.

A.21 Bluetooth PAN Profile

Development Name: BLUETOOTH_PAN
Build File Location: `/common/generic/bluetooth/`
System Model Location:

Layer: OS Services
Block: Comms Services
Sub-block: Networking Services
Component collection: Networking Plug-ins

License Classification: Optional Symbian

Exposes Third-Party APIs: YES

Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3

Description: Support for Bluetooth Personal Area Networking (PAN) Profile, which is analogous to a network interface agent for Bluetooth. Implements the Bluetooth Network Encapsulation Protocol (BNEP) as an Ethernet Packet Driver module, enabling PAN to behave like a regular Internet access provider.

A.22 Bluetooth Profiles

Development Name: BLUETOOTH_AVRCP, BLUETOOTH_GAVDP

Build File Location: `/common/generic/bluetooth/`

System Model Location:

Layer: OS Services
Block: Comms Services
Sub-block: Short Link Services
Component collection: Short Link

License Classification: Optional Symbian

Exposes Third-Party APIs: YES

Present in OS Releases: 9.1, 9.2, 9.3

Description: Support for Bluetooth Audio/Video Remote Control Profile (AVRCP) and Generic Audio/Visual Distribution Profile (GAVDP). AVRCP is implemented as a bearer plug-in to the remote-control framework. GAVDP is implemented as a thin layer over the Socket Server client APIs and allows clients to configure, send and receive data over the AVDTP protocol running inside the Bluetooth protocol plug-in.

A.23 Bluetooth Protocol Client APIs

Development Name: USER

Build File Location: `/common/generic/bluetooth/`

System Model Location:

Layer: OS Services
Block: Comms Services
Sub-block: Short Link Services

Component collection: Short Link

License Classification: Optional Symbian

Exposes Third-Party APIs: YES

Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: Bluetooth-specific APIs for use by Bluetooth socket clients, providing support for low-level control of protocol parameters (e.g., packet sizes) and hardware (e.g., power modes).

A.24 Bluetooth SDP

Development Name: BLUETOOTH_SDP

Build File Location: `/common/generic/bluetooth/`

System Model Location:

Layer: OS Services

Block: Comms Services

Sub-block: Short Link Services

Component collection: Short Link

License Classification: Optional Symbian

Exposes Third-Party APIs: YES

Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3

Description: Service Discovery agent and database, used by connected Bluetooth devices to query each other and exchange and store information about the Bluetooth services they support. Note that the SDP database is not persistent.

A.25 Bluetooth Stack PRT

Development Name: BLUETOOTH_STACK

Build File Location: `/common/generic/bluetooth/`

System Model Location:

Layer: OS Services

Block: Comms Services

Sub-block: Short Link Services

Component collection: Short Link Protocol Plug-ins

License Classification: Optional Symbian

Exposes Third-Party APIs: NO

Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3

Description: Bluetooth stack implementation in the form of a PRT protocol plug-in to Socket Server.

A.26 BMP Animation

Development Name: BMPANIM
Build File Location: `/common/generic/app-framework/bmpanim/group/`
System Model Location:
Layer: UI Framework
Component collection: UI Support
License Classification: Optional Replaceable
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Plug-in utility to Window Server implementing support for bitmap-based frame sequence animation. Forms part of the Animation framework.

A.27 Bookmark Support

Development Name: BOOKMARKS
Build File Location: `/common/generic/application-protocols/bookmarks/group/`
System Model Location:
Layer: Application Services
Component collection: Internet and Web Application Support
License Classification: Optional Replaceable
Exposes Third-Party APIs: NO
Present in OS Releases: 9.1, 9.2, 9.3
Description: Bookmark database support for Web browsers.

A.28 Bootstrap

Development Name: BOOTSTRAP
Build File Location: `/cedar/generic/base/e32/`
System Model Location:
Layer: Kernel Services and Hardware Interface
Block: Kernel Architecture
Component collection: Variant
License Classification: Optional Replaceable
Exposes Third-Party APIs: NO
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: Bootstraps the system by preparing the hardware, including memory and peripherals, mapping the virtual address space if an MMU is present and starting the kernel.

A.29 Broadcast Tuner

Development Name: TUNER

Build File Location: `/common/generic/Multimedia/Tuner/group/`

System Model Location:

Layer: OS Services

Block: Multimedia and Graphics Services

Component collection: Multimedia

License Classification: Optional Replaceable

Exposes Third-Party APIs: YES

Present in OS Releases: 9.1, 9.2, 9.3

Description: Integrated broadcast tuner API for analog or digital broadcast channel reception.

A.30 C Standard Library

Development Name: STDLIB

Build File Location: `/common/generic/syslibs/stdlib/group/`

System Model Location:

Layer: OS Services

Block: Generic OS Services

Component collection: Generic Libraries

License Classification: Optional Replaceable

Exposes Third-Party APIs: YES

Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: Port of a subset of the POSIX/C programming language Standard Library. Maps C function calls to native Symbian OS C++ APIs, allowing ported C applications to interface to native services. Orginally written to support porting of binary-only Java VM.

A.31 C32 Serial Server

Development Name: C32

Build File Location: `/common/generic/ser-comms/c32/group/`

System Model Location:

Layer: OS Services
Block: Comms Services
Sub-block: Comms Framework
Component collection: Data Comms Server

License Classification: Common Symbian

Exposes Third-Party APIs: YES

Present in OS Releases: 7.0, 7.1, 8.0, 8.1, 9.1, 9.2, 9.3

Description: Provides the client-session APIs for serial communications and the framework for creating and loading CSY plug-in modules which implement serial port abstractions, enabling clients to access virtual serial ports independently of the underlying hardware.

A.32 Calendar

Development Name: CALINTERIMAPI

Build File Location: `/common/generic/app-engines/calinterimapi/group/`

System Model Location:

Layer: Application Services
Component collection: PIM Application Services

License Classification: Optional Symbian

Exposes Third-Party APIs: YES

Present in OS Releases: 9.2, 9.3

Description: Calendar API replacement for Agenda Model, partially supports the iCalendar standard.

A.33 Camera

Development Name: ECAM

Build File Location: `/common/generic/Multimedia/Ecam/group/`

System Model Location:

Layer: Multimedia and Graphics Services
Component collection: Multimedia

License Classification: Optional Replaceable

Exposes Third-Party APIs: YES

Present in OS Releases: 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: Onboard camera API to provide compatibility for camera client applications between devices.

A.34 CDMA MTM

Development Name: CDMASMSMTM
Build File Location: `/common/generic/messaging/sms/multimode/group/`
System Model Location:
Layer: Application Services
Component collection: Messaging Application Support
License Classification: Optional Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: MTM plug-in to Messaging Store framework that implements messaging support for CDMA.

A.35 CDMA SMS Plug-ins

Development Name: CDMASMSSTACK
Build File Location: `/common/generic/nbprotocols/cdmasmsstack/group/`
System Model Location:
Layer: OS Services
Block: Comms Services
Sub-block: Telephony Services
Component collection: SMS Protocol Plug-ins
License Classification: Optional Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: CDMA SMS protocol implementation supporting CDMA teleservices.

A.36 CDMA TSY

Development Name: CDMATSY
Build File Location: `/common/generic/telephony/cdmatsy/group/`
System Model Location:
Layer: OS Services
Block: Comms Services
Sub-block: Telephony Services

Component collection: Telephony Server Plug-ins

License Classification: Reference/Test

Exposes Third-Party APIs: NO

Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: Reference-only TSY implementation of CDMA telephony extensions, replaced by licensees with a hardware specific implementation. Plug-in to ETel Telephony Server framework.

A.37 Central Repository

Development Name: CENTRALREPOSITORY

Build File Location: `/common/generic/syslibs/centralrepository/group/`

System Model Location:

Layer: Base Services

Component collection: Low-Level Libraries and Frameworks

License Classification: Common Symbian

Exposes Third-Party APIs: YES

Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3

Description: Persistent store for global settings that provides a notification mechanism allowing clients to register interest when settings change.

A.38 Certificate and Key Management

Development Name: CERTMAN

Build File Location: `/common/generic/security/certman/group/`

System Model Location:

Layer: OS Services

Block: Generic OS Services

Component collection: Generic Libraries

License Classification: Optional Replaceable

Exposes Third-Party APIs: YES

Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: Framework for managing and storing security certificates and keys supporting storage and retrieval, assignment of trust status, certificate chain construction, and certificate validation and revocation.

A.39 Certificate Store

Development Name: CERTSTORE
Build File Location: `/common/generic/security/certman/certstore/`
System Model Location:
Layer: OS Services
Block: Generic OS Services
Component collection: Generic Libraries
License Classification: Optional Replaceable
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Unified certificate store that provides clients with a single point of access to certificates stored on the device.

A.40 Character Encoding and Conversion Framework

Development Name: CHARCONV
Build File Location: `/common/generic/syslibs/charconv/`
System Model Location:
Layer: Base Services
Component collection: Low-Level Libraries and Frameworks
License Classification: Optional Replaceable
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Extensible framework supporting text conversion between Unicode and non-Unicode character sets (Symbian OS native text formats are Unicode).

A.41 Character Encoding and Conversion Plug-ins

Development Name: CHARCONV
Build File Location: `/common/generic/syslibs/charconv/`
System Model Location:
Layer: Base Services
Component collection: Low-Level Libraries and Frameworks
License Classification: Optional Replaceable
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: Converter plug-ins to the Character Encoding and Conversion Framework that support conversion to and from a variety of ASCII, UTF-7 and UTF-8 text formats, including JIS/ShiftJis.

A.42 Chinese Calendar Converter

Development Name: CALCON
Build File Location: `/common/generic/app-services/calcon/`
System Model Location:
Layer: Application Services
Component collection: PIM Application Support
License Classification: Optional Replaceable
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: API for converting between Gregorian and Chinese lunar calendar dates.

A.43 CLDC HI 1.1

Development Name: CLDCHI
Build File Location: `/common/generic/ME/group/`
System Model Location:
Layer: Java ME
Component collection: Virtual Machine
License Classification: Common Replaceable
Exposes Third-Party APIs: YES
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: Symbian OS port of the Sun CLDC HotSpot Implementation VM (CLDC HI) that forms part of the CLDC 1.1 specification (JSR-139).

A.44 Client Provisioning Adaptors

Development Name: DEVPROV_CLIENTPROV_ADAPTERS
Build File Location: `/common/generic/DevProv/Adapters/ClientProv/group/`
System Model Location:
Layer: Application Services
Component collection: Client Provisioning

License Classification: Optional Replaceable
Exposes Third-Party APIs: NO
Present in OS Releases: 9.1, 9.2, 9.3
Description: Adaptor plug-ins to the Client Provisioning Framework.

A.45 Client Provisioning Framework

Development Name: DEVPROV_CLIENTPROV_FRAMEWORK
Build File Location: `/common/generic/DevProv/ClientProv/group/`
System Model Location:
Layer: Application Services
Component collection: Client Provisioning
License Classification: Common Replaceable
Exposes Third-Party APIs: NO
Present in OS Releases: 9.1, 9.2, 9.3
Description: Framework implementing the OMA Client Provisioning standards and supporting provisioning of devices by network operators.

A.46 Clock

Development Name: CLOCK
Build File Location: `/common/generic/app-framework/clock/group/`
System Model Location:
Layer: UI Framework
Component collection: UI Support
License Classification: Optional Replaceable
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Shared library plug-in to the Window Server that supports creation of animation-based digital and analog clocks, used by UIs and applications.

A.47 Color Palette

Development Name: PALETTE
Build File Location: `/common/generic/graphics/palette/group/`

System Model Location:
Layer: OS Services
Block: Multimedia and Graphics Services
Component collection: Graphics Device Interface
License Classification: Common Replaceable
Exposes Third-Party APIs: YES
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: Low-level graphics support for licensee implementations of color-scheme switching.

A.48 Comms Database

Development Name: COMMDB
Build File Location: `/common/generic/comms-infras/commdb/group/`
System Model Location:
Layer: OS Services
Block: Comms Services
Sub-block: Comms Framework
Component collection: Comms Configuration Utilities
License Classification: Common Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Legacy repository for storing communications settings. It is replaced in Symbian OS v9 by the CommsDat interface to the Central Repository, although the original CommDB API is preserved for compatibility.

A.49 Comms Debug Utility

Development Name: COMMSDEBUGUTILITY
Build File Location: `/common/generic/comms-infras/commsdebugutility/group/`
System Model Location: N/A
License Classification: Test/Reference
Exposes Third-Party APIs: YES
Present in OS Releases: N/A
Description: Reimplementation of File Logger intended only for use by communications programs.

A.50 Comms Framework

Development Name: COMMSFW
Build File Location: `/common/generic/comms-infras/commsfw/group/`
System Model Location:
Layer: OS Services
Block: Comms Services
Sub-block: Comms Framework
Component collection: Comms Framework Utilities
License Classification: Common Symbian
Exposes Third-Party APIs: NO
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: Framework providing the base classes used to create communications servers, and the communication mechanisms used to communicate between communications server threads.

A.51 Comms Root Server

Development Name: ROOTSERVER
Build File Location: `/common/generic/comms-infras/rootserver/group/`
System Model Location:
Layer: OS Services
Block: Comms Services
Sub-block: Comms Framework
Component collection: Comms Process and Settings
License Classification: Common Symbian
Exposes Third-Party APIs: NO
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: Provides the main thread in the communications process that is responsible for starting and managing all other communications server threads (i.e. for starting communications servers as threads within the root server process). From Symbian OS v8, communications servers are started when the device boots up, rather than on demand.

A.52 Connection Provider Plug-in

Development Name: IPCPR
Build File Location: `/common/generic/networking/iprpr/group/`

System Model Location:
Layer: OS Services
Block: Comms Services
Sub-block: Networking Services
Component collection: Networking Plug-ins
License Classification: Optional Replaceable
Exposes Third-Party APIs: NO
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: Plug-in providing IP connections to clients, supporting bearer mobility.

A.53 Contacts Model

Development Name: CNTMODEL
Build File Location: `/common/generic/app-engines/cntmodel/group/`
System Model Location:
Layer: Application Services
Component collection: PIM Application Services
License Classification: Optional Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Application model (i.e. data store plus APIs) for a common contact or address-book application.

A.54 Content-Access Framework for DRM

Development Name: CAF2
Build File Location: `/common/generic/syslibs/caf2/group/`
System Model Location:
Layer: Application Services
Component collection: Content Handling
License Classification: Optional Symbian
Exposes Third-Party APIs: NO
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: Framework for brokering DRM-protected content between agents (DRM applications) and consumers (e.g. media players). Includes a Reference DRM Agent implementation.

A.55 Content-Handling Framework

Development Name: CONTENT_HANDLING
Build File Location: `/common/generic/content-handling/framework/group/`
System Model Location:
Layer: Application Services
Component collection: Application Framework
License Classification: Optional Replaceable
Exposes Third-Party APIs: YES
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: Provides a framework for content handlers that implements finding, loading, processing, and displaying of typed content on behalf of applications.

A.56 Control Environment (CONE)

Development Name: CONE
Build File Location: `/common/generic/app-framework/cone/group/`
System Model Location:
Layer: UI Framework
Component collection: UI Application Framework
License Classification: Common Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Control hierarchy and environment, providing UI-policy-free abstract controls, i.e. interactive screen elements and control context. Derived concrete controls are provided by the variant UI.

A.57 Core IPSec PRT

Development Name: IPSEC6
Build File Location: `/common/generic/networking/ipsec/ipsec6/group/`
System Model Location:
Layer: OS Services
Block: Comms Services
Sub-block: Networking Services

Component collection: Network Protocol Plug-ins

License Classification: Optional Replaceable

Exposes Third-Party APIs: YES

Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: IPSec for IP v4 and v6 including Authentication Header (AH) and Encapsulating Security Payload (ESP) cryptographic protocols. Implemented as a sockets server PRT plug-in module.

A.58 Cryptographic Token Framework

Development Name: CRYPTOTOKENS, FILETOKENS

Build File Location: `/common/generic/security/cryptotokens/group/`

System Model Location:

Layer: OS Services

Block: Generic OS Services

Component collection: Generic Libraries

License Classification: Optional Symbian

Exposes Third-Party APIs: YES

Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3

Description: Framework supporting use of secure hardware tokens (or their equivalent software emulations). Defines certificate and key storage and authentication APIs for secure hardware tokens, for example SD memory cards.

A.59 Cryptography Library

Development Name: CRYPTOGRAPHY

Build File Location: `/common/generic/security/crypto/group/`

System Model Location:

Layer: Base Services

Component collection: Low-Level Libraries and Frameworks

License Classification: Optional Symbian

Exposes Third-Party APIs: YES

Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: Non-RSA-based cryptographic algorithms including symmetric and asymmetric ciphers, hash functions and a cryptographic random-number generator. Supersedes RSA-based implementations. Implemented in 'strong' and 'weak' versions, of which the strong version is export-restricted.

A.60 CSD AGT

Development Name: CSDAGT

Build File Location: `/common/generic/networking/csdagt/group/`

System Model Location:

Layer: Comms Services

Sub-block: Networking Services

Component collection: Networking Plug-ins

License Classification: Optional Replaceable

Exposes Third-Party APIs: YES

Present in OS Releases: 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: AGT agent plug-in to Connection Agent framework that negotiates circuit-switched connections e.g. to GSM and CDMA networks, supporting dial-up networking services.

A.61 Data Engine

Development Name: DAMODEL

Build File Location: `/common/generic/app-engines/damodel/group/`

System Model Location:

Layer: Application Services

Component collection: Office Application Engines

License Classification: Optional Replaceable

Exposes Third-Party APIs: YES

Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: Legacy application model for a free-form database application, originating from the early EPOC built-in application set.

A.62 DBMS

Development Name: DBMS

Build File Location: `/common/generic/syslibs/dbms/group/`

System Model Location:

Layer: Base Services

Component collection: Low-Level Libraries and Frameworks

License Classification: Optional Symbian

Exposes Third-Party APIs: YES

Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: Relational database manager and server that defines database-access APIs and implements a proprietary database format. Supports both exclusive-access and shared-access databases.

A.63 Device Management Adaptors

Development Name: DEVPROV_DEVMAN_ADAPTERS
Build File Location: `/common/generic/SyncML/DevMan/group/`
System Model Location:
Layer: Application Services
Component collection: Device Management
License Classification: Optional Replaceable
Exposes Third-Party APIs: NO
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: Adaptor plug-ins to Device Management Framework, supporting application management, browser bookmarks, data synchronization and Accounts, Device Information, Device Management Accounts, Email, MMS, Network Access Points, SMS, WAP Proxies.

A.64 Device Management Framework

Development Name: DEVPROV_DEVMAN_FRAMEWORK
Build File Location: `/common/generic/SyncML/DevMan/group/`
System Model Location:
Layer: Application Services
Component collection: Device Management
License Classification: Optional Replaceable
Exposes Third-Party APIs: NO
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: Framework that implements OMA Device Management based on SyncML and supports remote provisioning of devices by network operators.

A.65 DHCP

Development Name: DHCP
Build File Location: `/common/generic/networking/dhcp/group/`
System Model Location:
Layer: OS Services
Block: Comms Services

Sub-block: Networking Services
Component collection: TCP/IP Utilities

License Classification: Optional Replaceable

Exposes Third-Party APIs: NO

Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3

Description: For internal use only, this Dynamic Host Configuration Protocol (DHCP) implementation is used by networking components.

A.66 Dial

Development Name: Dial

Build File Location: `/generic/telephony/dial/group/`

System Model Location:

Layer: OS Services
Block: Comms Services
Sub-block: Telephony Services
Component collection: Telephony Utilities

License Classification: Optional Replaceable

Exposes Third-Party APIs: N/A

Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: Deprecated utility APIs for phone-number manipulation.

A.67 Dialog

Development Name: DIALOG

Build File Location: `/common/generic/networking/DIALOG/group/`

System Model Location:

License Classification: Optional Replaceable

Exposes Third-Party APIs: YES

Present in OS Releases:

Description: Generic connection-agent dialog server that allows user interaction where appropriate when setting up a connection to the Internet.

A.68 DND

Development Name: DND

Build File Location: `/common/generic/networking/dnd/group/`

System Model Location:
Layer: OS Services
Block: Comms Services
Sub-block: Networking Services
Component collection: TCP/IP Utilities
License Classification: Common Symbian
Exposes Third-Party APIs: NO
Present in OS Releases: 7.0, 7.0s, 9.1, 9.2, 9.3
Description: For internal use only, this Domain Name Service (DNS) implementation is used by networking components.

A.69 Emulator

Development Name: WINS_VARIANT_EKA2
Build File Location: `/cedar/generic/base/wins/`
System Model Location:
Layer: Kernel Services and Hardware Interface
Block: Kernel Architecture
Component collection: Variant
License Classification: Optional Replaceable
Exposes Third-Party APIs: NO
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Symbian OS emulator for Microsoft Windows platforms. In EKA2, this is implemented as a hardware target variant.

A.70 ESock Socket Server

Development Name: ESOCK
Build File Location: `/common/generic/comms-infras/esock/group/`
System Model Location:
Layer: OS Services
Block: Comms Services
Sub-block: Networking Services
Component collection: Sockets Server
License Classification: Common Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Server and framework for sockets-based communications. Loads socket implementations from PRT protocol plug-in modules.

A.71 ETel Third-Party API

Development Name: ETEL3RDPARTY
Build File Location: `/common/generic/telephony/etel3rdparty/group/`
System Model Location:
Layer: OS Services
Block: Comms Services
Sub-block: Telephony Services
Component collection: Telephony Server
License Classification: Common Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Telephony API subset intended for third-party developer use.

A.72 ETel CDMA

Development Name: ETELCDMA
Build File Location: `/common/generic/telephony/etelcdma/group/`
System Model Location:
Layer: OS Services
Block: Comms Services
Sub-block: Telephony Services
Component collection: Telephony Server
License Classification: Optional Symbian
Exposes Third-Party APIs: NO
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: CDMA extensions to the ETel Telephony Server.

A.73 ETel Multimode

Development Name: ETELMM
Build File Location: `/common/generic/telephony/etelmm/group/`
System Model Location:
Layer: OS Services
Block: Comms Services
Sub-block: Telephony Services

Component collection: Telephony Server

License Classification: Common Symbian

Exposes Third-Party APIs: YES

Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: ETel Telephony Server API extensions that provide network-agnostic APIs for voice, fax, data and multimedia calls and that, therefore, abstract the differences between GSM and CDMA networks.

A.74 ETel Packet Data

Development Name: ETELPCKT

Build File Location: `/common/generic/telephony/etelpckt/group/`

System Model Location:

Layer: OS Services

Block: Comms Services

Sub-block: Telephony Services

Component collection: Telephony Server

License Classification: Common Symbian

Exposes Third-Party APIs: NO

Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: ETel Telephony Server API extensions that provide access to packet services on GPRS, UMTS and CDMA/CDMA2000 networks.

A.75 ETel Server and Core

Development Name: ETEL

Build File Location: `/common/generic/telephony/etel/group/`

System Model Location:

Layer: OS Services

Block: Comms Services

Sub-block: Telephony Services

Component collection: Telephony Server

License Classification: Common Symbian

Exposes Third-Party APIs: YES

Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3

Description: ETel Telephony Server and core APIs.

A.76 ETel SIM Toolkit

Development Name: ETELSAT
Build File Location: `/common/generic/telephony/etelsat/group/`
System Model Location:
Layer: OS Services
Block: Comms Services
Sub-block: Telephony Services
Component collection: Telephony Server
License Classification: Common Symbian
Exposes Third-Party APIs: NO
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: ETel Telephony Server API extension that provides access to the GSM/WCDMA (U)SIM Application Toolkit.

A.77 Ethernet Driver

Development Name: ETHERDRV
Build File Location: `/common/generic/networking/etherdrv/group/`
System Model Location:
Layer: Kernel Services and Hardware Interface
Block: Kernel Architecture
Component collection: Logical Device Drivers
License Classification: Common Replaceable
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: LDD and PDD implementations for Ethernet cards and emulators.

A.78 Ethernet NIF

Development Name: ETHINT
Build File Location: `/common/generic/networking/ether802/group/`
System Model Location:
Layer: OS Services
Block: Comms Services

Sub-block: Networking Services
Component collection: Link Layer Control

License Classification: Optional Replaceable

Exposes Third-Party APIs: NO

Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: Ethernet protocol NIF plug-in to Network Interface Manager supporting wired Ethernet.

A.79 Ethernet Over IR Packet Driver

Development Name: IRLANPACKETDRIVERS

Build File Location: `/common/generic/networking/ether802/group/`

System Model Location:

Layer: OS Services
Block: Comms Services
Sub-block: Networking Services
Component collection: Link Layer Control

License Classification: Optional Replaceable

Exposes Third-Party APIs: NO

Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: Logical and physical device drivers providing Ethernet-framing services for networking over infrared.

A.80 Ethernet Packet Driver

Development Name: ETHER802

Build File Location: `/common/generic/networking/ether802/group/`

System Model Location:

Layer: Kernel Services and Hardware Interface
Block: Kernel Architecture
Component collection: Logical Device Drivers

License Classification: Optional Replaceable

Exposes Third-Party APIs: YES

Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: Logical and physical LAN packet drivers providing Ethernet framing to generic-networking services.

A.81 Event Logger

Development Name: LOGENG
Build File Location: `/common/generic/syslibs/logeng/group/`
System Model Location:
Layer: OS Services
Block: Generic OS Services
Component collection: Generic Services
License Classification: Optional Replaceable
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Interface that supports logging events to a logging engine and retrieval, filtering and viewing of logged events by clients.

A.82 FAT Filename Conversion Plug-ins

Development Name: FATCHARSETCONV
Build File Location: `/common/generic/syslibs/FATCharsetConv/group/`
System Model Location:
Layer: Base Services
Component collection: User Library and File Server
License Classification: Common Replaceable
Exposes Third-Party APIs: NO
Present in OS Releases: 9.1, 9.2, 9.3
Description: File Server plug-ins that support converting FAT file names from and to Unicode.

A.83 Fax Client and Server

Development Name: FAX
Build File Location: `/common/generic/telephony/FAX/group/`
System Model Location:
Layer: OS Services
Block: Comms Services
Sub-block: Telephony Services
Component collection: Telephony Server
License Classification: Optional Replaceable
Exposes Third-Party APIs: NO
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: Fax server and protocol stack together with client-side APIs. Extension to ETel Telephony Server that manages fax transmission and reception requests from application clients.

A.84 Feature Registry

Development Name: FEATREG
Build File Location: `/common/generic/syslibs/featreg/group/`
System Model Location:
Layer: Base Services
Component collection: Low-Level Libraries and Frameworks
License Classification: Common Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 9.2, 9.3
Description: APIs for run-time discovery of supported 'features' on a given platform.

A.85 FEP Base

Development Name: FEPBASE
Build File Location: `/common/generic/app-framework/fepbase/group/`
System Model Location:
Layer: UI Framework
Component collection: UI Application Framework
License Classification: Common Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Framework for front-end processors (FEPs) that enable application-independent input preprocessing to support keyboard mapping, multitap-keyboard input, handwriting recognition, voice recognition, and so on.

A.86 File Converter Framework

Development Name: CONARC
Build File Location: `/common/generic/app-framework/conarc/group/`
System Model Location:
Layer: Application Services

Component collection: Application Framework

License Classification: Optional Replaceable

Exposes Third-Party APIs: YES

Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: Framework for converter plug-ins that enable file conversions based on MIME type.

A.87 File Converter Plug-ins

Development Name: CHTMLTOCRTCONVERTER, CONVERT, RICHTEXTTOHTMLCONV

Build File Location: `/common/generic/app-services/chtmltocrtconv/group/`, `/common/generic/app-engines/convert/group/`, `/common/generic/app-services/richtexttohtmlconv/group/`

System Model Location:

Layer: Application Services

Component collection: PIM Application Support

License Classification: Optional Replaceable

Exposes Third-Party APIs: NO

Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: Plug-ins to File Converter Framework that support conversions between Symbian OS rich-text objects, HTML files and Microsoft Office file formats.

A.88 File Logger

Development Name: FLOGGER

Build File Location: `/common/generic/comms-infras/flogger/group/`

System Model Location:

Layer: OS Services

Block: Generic OS Services

Component collection: Generic Services

License Classification: Optional Replaceable

Exposes Third-Party APIs: YES

Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: Legacy utility that enables logging of events to file.

A.89 File Server

Development Name: F32_EKA2
Build File Location: `/cedar/generic/base/f32/group/`
System Model Location:
Layer: Base Services
Component collection: User Library and File Server
License Classification: Common Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Generic file-system server and framework providing an extension interface that supports the implementation of custom file systems. File systems are implemented as loadable FSY plug-ins. All file-system access is managed through client sessions with the file server.

A.90 File Systems

Development Name: FILESYS
Build File Location: `/cedar/generic/base/f32/group/`
System Model Location:
Layer: Base Services
Component collection: User Library and File Server
License Classification: Common Replaceable
Exposes Third-Party APIs: NO
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: File system plug-in extensions implementing LFFS and FAT file systems.

A.91 Flash Translation Layer

Development Name: UNISTORE2_DRIVERS
Build File Location: `/cedar/generic/base/omap/`
System Model Location:
Layer: Kernel Services and Hardware Interface
Block: Kernel Architecture
Component collection: Variant
License Classification: Optional Replaceable
Exposes Third-Party APIs: NO
Present in OS Releases: 9.1, 9.2, 9.3
Description: File-system plug-in implementation of flash driver support.

A.92 Font and Bitmap Server

Development Name: FBSERV
Build File Location: `/common/generic/graphics/fbserv/group/`
System Model Location:
Layer: OS Services
Block: Multimedia and Graphics Services
Component collection: Graphics and Printing Services
License Classification: Common Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Server that manages system-wide shared access to single-instance fonts and bitmaps, providing bitmap and font services for native bitmap fonts and vector fonts through its client-side APIs.

A.93 Font Store

Development Name: FNTSTORE
Build File Location: `/common/generic/graphics/fntstore/group/`
System Model Location:
Layer: OS Services
Block: Multimedia and Graphics Services
Component collection: Graphics and Printing Services
License Classification: Common Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Provides font storage and font-file loading, including closest-fit matching of font requests. Supports the Open Font specification for vector fonts as well as proprietary Symbian OS bitmap fonts.

A.94 FreeType Font Rasterizer

Development Name: FREETYPE
Build File Location: `/common/generic/graphics/freetype/group/`
System Model Location:
Layer: OS Services
Block: Multimedia and Graphics Services
Component collection: Graphics and Printing Services

License Classification: Optional Replaceable
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Reference implementation port of the FreeType TrueType font rasterizer, supporting FreeType 2 TrueType font descriptions and the Open Font interface.

A.95 FTP Engine

Development Name: FTP_E
Build File Location: `/common/generic/networking/ftp_e/group/`
System Model Location:
Layer: Application Services
Component collection: Internet and Web Application Support
License Classification: Optional Replaceable
Exposes Third-Party APIs: NO
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: For internal use only. Symbian OS File Transfer Protocol (FTP) daemon implementation used by networking components.

A.96 GDI

Development Name: GDI
Build File Location: `/common/generic/graphics/gdi/group/`
System Model Location:
Layer: OS Services
Block: Multimedia and Graphics Services
Component collection: Graphics Device Interface
License Classification: Common Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Device-independent graphics context abstraction that supports drawing to various devices including screens and printers, which are treated as specialized graphics contexts.

A.97 GPRS/UMTS QoS PRT

Development Name: GUQOS
Build File Location: `/common/generic/networking/guqos/group/`

System Model Location:
Layer: OS Services
Block: Comms Services
Sub-block: Networking Services
Component collection: Networking Plug-ins
License Classification: Optional Replaceable
Exposes Third-Party APIs: NO
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: PRT socket protocol module used by the QoS Framework and network-interface components to implement 3GPP parameters.

A.98 Graphics Effects

Development Name: GFXTRANSEFFECT
Build File Location: `/common/app-framework/gfxtranseffect/group/`
System Model Location:
Layer: UI Framework
Component collection: UI Support
License Classification: Optional Replaceable
Exposes Third-Party APIs: NO
Present in OS Releases: 9.1, 9.2, 9.3
Description: Support for flicker-free window and window-contents animation and graphics-composition effects, for example to support animated-menu 'transition effects'.

A.99 Grid

Development Name: GRID
Build File Location: `/common/generic/app-framework/grid/group/`
System Model Location:
Layer: UI Framework
Component collection: UI Support
License Classification: Optional Replaceable
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Layout engine for spreadsheet-style grid layout, presentation, print preview and printing. Now considered a legacy component.

A.100 GSM Utilities

Development Name: GSMU
Build File Location: `/common/generic/nbprotocols/smsstackv2/gsmu/`
System Model Location:
Layer: OS Services
Block: Comms Services
Sub-block: Telephony Services
Component collection: SMS Utilities
License Classification: Common Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Utilities for processing GSM SMS messages, including encoding and decoding routines, used by SMS PRT and its clients.

A.101 HCI Framework

Development Name: HCI_V2_FRAMEWORK
Build File Location: `/common/generic/bluetooth/`
System Model Location:
Layer: OS Services
Block: Comms Services
Sub-block: Short Link Services
Component collection: Short Link
License Classification: Optional Symbian
Exposes Third-Party APIs: NO
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: Bluetooth Host Controller Interface (HCI) implementation.

A.102 Help

Development Name: HLPMODEL
Build File Location: `/common/generic/app-services/hlpmodel/group/`
System Model Location:
Layer: Application Services
Component collection: Other Application Services
License Classification: Optional Symbian
Exposes Third-Party APIs: YES

Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: Context-sensitive help engine providing a read-only interface to all help files on a Symbian OS device.

A.103 HTTP Filter Plug-ins

Development Name: HTTP

Build File Location: `/common/generic/application-protocols/http/group/`

System Model Location:

Layer: Application Services

Component collection: Other Application Services

License Classification: Optional Replaceable

Exposes Third-Party APIs: YES

Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: Plug-ins to HTTP Transport Framework that are dynamically loaded to configure a transport session before use. Filters encapsulate responses to session events, for example, client authentication, message validation, and message redirection.

A.104 HTTP Protocol Plug-ins

Development Name: HTTP

Build File Location: `/common/generic/application-protocols/http/group/`

System Model Location:

Layer: Application Services

Component collection: Internet and Web Application Support

License Classification: Optional Replaceable

Exposes Third-Party APIs: YES

Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: Plug-ins to HTTP Transport Framework that are dynamically loaded. Application and network-protocol handlers including TCP/IP, HTTP, and WSP.

A.105 HTTP Transport Framework

Development Name: HTTP

Build File Location: `/common/generic/application-`

`protocols/http/group/`

System Model Location:

Layer: Application Services

Component collection: Internet and Web Application Support

License Classification: Optional Replaceable

Exposes Third-Party APIs: YES

Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: Framework that enables clients to establish a transport session for HTTP-like protocols providing core APIs for Transport Sessions, Transactions and Messages.

A.106 HTTP Utilities Library

Development Name: INETPROTUTIL

Build File Location: `/common/generic/application-protocols/inetprotutil/group/`

System Model Location:

Layer: Application Services

Component collection: Internet and Web Application Support

License Classification: Optional Replaceable

Exposes Third-Party APIs: YES

Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3

Description: Convenience component for storing utility classes commonly used by Internet-protocol parsing components.

A.107 Image Conversion Library

Development Name: ICL, ICL_IMAGEDISPLAY, IMAGETRANSFORM

Build File Location: `/common/generic/Multimedia/ICL/group/`

System Model Location:

Layer: OS Services

Block: Multimedia and Graphics Services

Component collection: Multimedia

License Classification: Common Symbian

Exposes Third-Party APIs: YES

Present in OS Releases: 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: Extensible framework that integrates image conversion into the Multimedia Framework.

A.108 Image Conversion Library Plug-ins

Development Name: IMAGETRANSFORM, GIFSCALER
Build File Location: `/common/generic/Multimedia/ICL/ImageTransform/group/`, `/common/generic/Multimedia/ICL/GIFSCALER/group/bld.inf`
System Model Location:
Layer: OS Services
Block: Multimedia and Graphics Services
Component collection: Multimedia
License Classification: Common Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Default reference codecs for common still image formats including Gif, Jpeg, Png, Bmp, Mbm, and others.

A.109 IMAP4 MTM

Development Name: IMAPSERVERMTM
Build File Location: `/common/generic/messaging/email/group/`
System Model Location:
Layer: Application Services
Component collection: Messaging Application Support
License Classification: Optional Symbian
Exposes Third-Party APIs: NO
Present in OS Releases: 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: MTM plug-in to Messaging Store framework supporting sending, receiving and editing of IMAP4 (HTML mail) email messages.

A.110 Internet Sockets

Development Name: INSOCK
Build File Location: `/common/generic/networking/insock/group/`
System Model Location:
Layer: OS Services
Block: Comms Services
Sub-block: Networking Services
Component collection: ESock API Extensions

License Classification: Common Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Plug-in library to Socket Server that specializes generic socket-server address classes for TCP/IP v4 or v6 protocols to implement sockets over TCP/IP.

A.111 IP Event Notifier

Development Name: IPEVENTNOTIFIER
Build File Location: `/common/generic/networking/IPEventNotifier/group/`
System Model Location:
Layer: OS Services
Block: Comms Services
Sub-block: Networking Services
Component collection: Network Protocol Plug-ins
License Classification: Common Symbian
Exposes Third-Party APIs: NO
Present in OS Releases: 9.1, 9.2, 9.3
Description: Catches events occuring within the IP stack and publishes them to registered subscribers. Implemented as an IP Hook.

A.112 IP Hook

Development Name: INHOOK6
Build File Location: `/common/generic/networking/inhook6/group/`
System Model Location:
Layer: OS Services
Block: Comms Services
Sub-block: Networking Services
Component collection: Network Protocol Plug-ins
License Classification: Common Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Implements a TCP/IP Hook interface to which modules bind to perform transformations on inbound and outbound IP packets.

A.113 IPSec

Development Name: IPSEC
Build File Location: `/common/generic/networking/ipsec/group/`
System Model Location:
Layer: OS Services
Block: Comms Services
Sub-block: Networking Services
Component collection: TCP/IP Security
License Classification: Optional Replaceable
Exposes Third-Party APIs: NO
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: IPSec key negotiation for IPv4/v6 that enables policy-based secure networks, for example virtual private networks (VPNs).

A.114 IrDA CSY

Development Name: IRCOMM
Build File Location: `/common/generic/infra-red/irda/group/`
System Model Location:
Layer: OS Services
Block: Comms Services
Sub-block: Short Link Services
Component collection: Serial Comms Server Plug-ins
License Classification: Common Replaceable
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: CSY plug-in to C32 Serial Server providing serial port emulation over an IrDA link.

A.115 IrDA PRT

Development Name: IRDA
Build File Location: `/common/generic/infra-red/irda/`
System Model Location:
Layer: OS Services
Block: Comms Services
Sub-block: Short Link Services

Component collection: Short Link Protocol Plug-ins

License Classification: Optional Symbian

Exposes Third-Party APIs: YES

Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: IrDA protocol stack implemented as a PRT Socket Server protocol plug-in.

A.116 Java IO

Development Name: JAVA.IO

Build File Location: `/common/generic/ME/group/`

System Model Location:

Layer: Java ME

Component collection: CLDC 1.1

License Classification: Common Replaceable

Exposes Third-Party APIs: YES

Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3

Description: CLDC 1.1 Java I/O libraries that define the data-stream-based input and output APIs and APIs for reading and writing bytes and basic Java types.

A.117 Java Lang

Development Name: JAVA.LANG

Build File Location: `/common/generic/ME/group/`

System Model Location:

Layer: Java ME

Component collection: CLDC 1.1

License Classification: Common Replaceable

Exposes Third-Party APIs: YES

Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3

Description: CLDC 1.1 language libraries that define basic Java types and objects, including Byte, Integer, Object and Thread.

A.118 Java MIDlet Installer

Development Name: JAVAMIDLETINSTALLER

Build File Location: `/common/generic/security/JavaMIDletInstaller/group/`

System Model Location:
Layer: Application Services
Component collection: Application Framework
License Classification: Optional Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Responsible for installation, removal and management of MIDP JAR files and MIDlets, including OTA support.

A.119 Java Utilities

Development Name: JAVA.UTIL
Build File Location: `/common/generic/ME/group/`
System Model Location:
Layer: Java ME
Component collection: CLDC 1.1
License Classification: Common Replaceable
Exposes Third-Party APIs: YES
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: CLDC 1.1 utilities library that supplies basic utility classes, including Date and Time, and collection classes, including Hashtable, Stack and Vector.

A.120 JTWI 1.0

Development Name: J2ME9.2
Build File Location: `/common/generic/ME/group/`
System Model Location:
Layer: Java ME
Component collection: MIDP 2.0 Packages
License Classification: Optional Replaceable
Exposes Third-Party APIs: YES
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: Wireless Messaging API for JTW 1.0 (JSR185).

A.121 Kernel Architecture 2

Development Name: E32_EKA2
Build File Location: `/cedar/generic/base/e32/`

System Model Location:
Layer: Kernel Services and Hardware Interface
Block: Kernel Architecture
Component collection: Kernel Services
License Classification: Common Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 8.1b, 9.1, 9.2, 9.3
Description: Symbian OS EKA2 (real-time) kernel, delivered in Symbian OS v8.1b and in all releases from Symbian OS v9.

A.122 Key Store

Development Name: KEYSTORE
Build File Location: `/common/generic/security/certman/certstore/`
System Model Location:
Layer: OS Services
Block: Generic OS Services
Component collection: Generic Libraries
License Classification: Optional Replaceable
YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Provides a repository of private PKI keys and APIs for storing and retrieving keys and for managing the store itself.

A.123 LCDUI Plug-in

Development Name: LCDUIB
Build File Location: `/common/generic/ME/group/`
System Model Location:
Layer: Java ME
Component collection: Low-Level Plug-ins
License Classification: Common Replaceable
Exposes Third-Party APIs: YES
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: Low-level graphics APIs with direct screen access, implemented as a plug-in that may be replaced with an alternative implementation.

A.124 Locale Support

Development Name: LOCE32, ELOCL, EKTRAN
Build File Location: `/common/generic/base/loce32/`
System Model Location:
Layer: Kernel Services and Hardware Interface
Component collection: Localization
License Classification: Optional Replaceable
Exposes Third-Party APIs: NO
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Customizable plug-in that implements a library of locale-specific settings and standard strings including currency symbol and date format, used by both the Kernel and the User Library.

A.125 Lubbock Variant

Development Name: LUBBOCK_EKA2
Build File Location: `/cedar/generic/base/lubbock/`
System Model Location:
Layer: Kernel Services and Hardware Interface
Block: Kernel Architecture
Component collection: Variant
License Classification: Optional Replaceable
Exposes Third-Party APIs: NO
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: Variant code for the Intel Lubbock development board.

A.126 MBuf Manager

Development Name: MBUFMAN
Build File Location: `/common/generic/comms-infras/mbufmgr/group/`
System Model Location:
Layer: OS Services
Block: Comms Services
Sub-block: Comms Framework
Component collection: Comms Framework Utilities
License Classification: Common Symbian
Exposes Third-Party APIs: NO
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: MBuf implementation enabling efficient inter-thread communications within the communications process.

A.127 Media Drivers

Development Name: MEDUSII, MEDUSII_CRASHLOG, MEDUSIIS
Build File Location: `/cedar/generic/base/e32/drivers/unistore2/`, `/cedar/generic/base/integrator/logic/lmnand2/`
System Model Location:
Layer: Kernel Services and Hardware Interface
Block: Kernel Architecture
Component collection: Logical Device Drivers
License Classification: Common Symbian
Exposes Third-Party APIs: NO
Present in OS Releases: 9.1, 9.2, 9.3
Description: NAND-flash-media driver libraries, replacing the Flash Translation Layer implementation.

A.128 Message Store

Development Name: MSG, MSG_FRAMEWORK
Build File Location: `/common/generic/messaging/group/`, `/common/generic/messaging/framework/group/`
System Model Location:
Layer: Application Services
Component collection: Messaging Application Support
License Classification: Optional Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: The message server, store and framework. Supports standard message types (e.g. email, SMS) and defines the plug-in interface for Message Type Modules (MTMs) that implement message handling.

A.129 MIDI Driver

Development Name: MMF_DEVMIDI
Build File Location: `/common/generic/Multimedia/MMF/MIDI/group/`
System Model Location:

Layer: Kernel Services and Hardware Interface
Block: Kernel Architecture
Component collection: Logical Device Drivers

License Classification: Common Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: API to support hardware-accelerated MIDI engines.

A.130 MIDP Device Control

Development Name: MIDP
Build File Location: `/common/generic/ME/group/`
System Model Location:
Layer: Java ME
Component collection: MIDP 2.0 Profile

License Classification: Common Replaceable
Exposes Third-Party APIs: YES
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: Enabler for MIDP device-control APIs (JSR-118), for example, to control device vibration and backlight, and to enable platform requests (e.g. opening a browser page), triggered by device events.

A.131 MIDP File GCF

Development Name: GCF
Build File Location: `/common/generic/ME/group/`
System Model Location:
Layer: Java ME
Component collection: MIDP 2.0 Packages

License Classification: Optional Replaceable
Exposes Third-Party APIs: YES
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: MIDP File Connection APIs (JSR-075) implemented through the GCF-communications framework.

A.132 MIDP GSM Security Recommended Policy

Development Name: MIDP2 SECURITY RP
Build File Location: `/common/generic/ME/midp2security/`
System Model Location:

Layer: Java ME

Component collection: MIDP 2.0 Profile

License Classification: Common Replaceable

Exposes Third-Party APIs: YES

Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3

Description: Enabler for MIDP 2.0 Security Recommended Policy enabling domain-based protection.

A.133 MIDP IO

Development Name: JAVAX.MICROEDITION.IO

Build File Location: `/common/generic/ME/group/`

System Model Location:

Layer: Java ME

Component collection: MIDP 2.0 Profile

License Classification: Common Replaceable

Exposes Third-Party APIs: YES

Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3

Description: MIDP high-level input/output APIs, including networking support and HTTP connections.

A.134 MIDP LCDUI

Development Name: JAVAX.MICROEDITION.LCDUI

Build File Location: `/common/generic/ME/group/`

System Model Location:

Layer: Java ME

Component collection: MIDP 2.0 Profile

License Classification: Common Replaceable

Exposes Third-Party APIs: YES

Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3

Description: MIDP-graphics APIs that use UIKON native controls to acquire platform-specific look and feel through the UI Application Framework LAF implementation, which is customized by the UI variant.

A.135 MIDP MIDlet

Development Name: MIDP2

Build File Location: `/common/generic/ME/group/`

System Model Location:

Layer: Java ME
Block: Java
Sub-block: ME
Component collection: MIDP 2.0 Profile
License Classification: Common Replaceable
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: MIDlet lifecycle implementation.

A.136 MIDP PIM

Development Name: MIDP
Build File Location: `/common/generic/ME/group/`
System Model Location:
Layer: Java ME
Component collection: MIDP 2.0 Packages
License Classification: Optional Replaceable
Exposes Third-Party APIs: YES
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: MIDP Personal Information Management (PIM) APIs (JSR-075).

A.137 MIDP RMS

Development Name: JAVAX.MICROEDITION.RMS
Build File Location: `/common/generic/ME/group/`
System Model Location:
Layer: Java ME
Component collection: MIDP 2.0 Profile
License Classification: Common Replaceable
Exposes Third-Party APIs: YES
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: MIDP persistence APIs implemented over native DBMS.

A.138 MIME Recognizer Framework

Development Name: EMIME
Build File Location: `/common/generic/app-framework/emime/group/`
System Model Location:

Layer: Application Services
Component collection: Content Handling
License Classification: Common Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Recognizer framework for MIME data types.

A.139 MMF Recognizers

Development Name: RECMMF
Build File Location: `/common/generic/Multimedia/MMF/group/`
System Model Location:
Layer: Application Services
Component collection: Content Handling
License Classification: Common Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Recognizer framework plug-ins to recognize specific multimedia data and document types.

A.140 MMS MTM

Development Name: MMS
Build File Location: `/common/generic/messaging/group/`
System Model Location:
Layer: Application Services
Component collection: Messaging Application Support
License Classification: Optional Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: MTM plug-in to Messaging Store framework supporting sending, receiving and editing of MMS messages. From Symbian OS v9, legacy component and licensees provide their own implementations, if any.

A.141 MMS Settings

Development Name: MMSSETTINGS
Build File Location: `/common/generic/messaging/mmsettings/group/`

System Model Location:
Layer: Application Services
Component collection: Messaging Application Support
License Classification: Optional Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: Encapsulation of MMS settings that are stored in the message store.

A.142 Mobile 3D 1.0

Development Name: M3GIO
Build File Location: `/common/generic/ME/group/`
System Model Location:
Layer: Java ME
Component collection: MIDP 2.0 Packages
License Classification: Optional Replaceable
Exposes Third-Party APIs: YES
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: 3D-graphics APIs for scalable, small-footprint, devices (JSR-184).

A.143 Mobile Media API 1.1

Development Name: MMAPI11
Build File Location: `/common/generic/ME/group/`
System Model Location:
Layer: Java ME
Component collection: MIDP 2.0 Packages
License Classification: Optional Replaceable
Exposes Third-Party APIs: YES
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: Mobile Media APIs (JSR-135).

A.144 m-Router

Development Name: MROUTERSECURE
Build File Location: `/common/generic/connectivity/BAL/Plugins/mRouter3/group/`

System Model Location:
Layer: OS Services
Block: Connectivity Services
Component collection: Device Connection
License Classification: Optional Replaceable
Exposes Third-Party APIs: NO
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Plug-in to Bearer Abstraction Layer component providing m-Router connectivity link layer.

A.145 Multimedia Framework

Development Name: MMF, COMMON
Build File Location: `/common/generic/Multimedia/MMF/group/`
System Model Location:
Layer: Multimedia and Graphics Services
Component collection: Multimedia
License Classification: Common Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: Lightweight multi-threaded plug-in framework for handling multimedia data that provides client APIs for audio playback, recording and conversion, tone playback, video playback and recording, MIDI playback, and speech recognition. Supports hardware acceleration via Media Device Framework.

A.146 Multimedia Framework Plug-ins

Development Name: MMFAUDIOCONTROLLER, MMFSTDSOURCEANDSINKPLUGIN, MMFLINEARAUDIOCODECS, GSM610, MMFAUDIOOUTPUT, MMFAUDIOINPUT, MMFFORMATBASECLASSES, MMFWAVFORMAT, MMFRAWFORMAT, MMFAUFORMAT
Build File Location: `/common/generic/Multimedia/MMF/MMPfiles/plugin/`
System Model Location:
Layer: Multimedia and Graphics Services
Component collection: Multimedia
License Classification: Common Symbian
Exposes Third-Party APIs: YES

Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: Reference audio controller plug-ins.

A.147 MultiMode TSY

Development Name: MMTSY
Build File Location: `/common/generic/telephony/mmtsy/group/`
System Model Location:
Layer: OS Services
Block: Comms Services
Sub-block: Telephony Services
Component collection: Telephony Server Plug-ins
License Classification: Reference/Test
Exposes Third-Party APIs: NO
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Reference ETel Telephony Server TSY plug-in implementing GSM and GPRS specific extensions. Replaced on an actual device by a hardware-specific licensee TSY.

A.148 Network Controller

Development Name: NETCON
Build File Location: `/common/generic/networking/netcon/group/`
System Model Location:
Layer: OS Services
Block: Comms Services
Sub-block: Comms Framework
Component collection: Data Comms Server
License Classification: Optional Replaceable
Exposes Third-Party APIs: NO
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Used by Network Interface Manager to select an appropriate network interface agent to create an outgoing network interface.

A.149 Network Interface Manager

Development Name: NIFMAN
Build File Location: `/common/generic/comms-infras/nifman/group/`

System Model Location:
Layer: OS Services
Block: Comms Services
Sub-block: Comms Framework
Component collection: Data Comms Server
License Classification: Common Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Framework for creating, loading and managing interface agent (AGT) and interface (NIF) plug-ins to create bearer-level network connections.

A.150 Null AGT

Development Name: NULLAGT
Build File Location: `/common/generic/networking/NULLAGT/group/`
System Model Location:
Layer: OS Services
Block: Comms Services
Sub-block: Networking Services
Component collection: Networking Plug-ins
License Classification: Optional Replaceable
Exposes Third-Party APIs: NO
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Minimal connection AGT agent plug-in, enables connection to an Ethernet LAN.

A.151 OBEX Extension API

Development Name: OBEX_EXTENSIONAPIS
Build File Location: `/common/generic/obex/obexextensionapis/group/`
System Model Location:
Layer: OS Services
Block: Comms Services
Sub-block: Short Link Services
Component collection: OBEX
License Classification: Optional Symbian
Exposes Third-Party APIs: NO

Present in OS Releases: 9.1, 9.2, 9.3
Description: Packet extensions for the OBEX implementation.

A.152 OBEX MTMs

Development Name: MSG_OBEXMTM
Build File Location: `/common/generic/messaging/obex/group/`
System Model Location:
Layer: Application Services
Component collection: Messaging Application Support
License Classification: Optional Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: MTM plug-ins to Messaging Store framework supporting OBEX messages over Bluetooth and infrared.

A.153 OBEX Protocol

Development Name: OBEX
Build File Location: `/common/generic/obex/group/`
System Model Location:
Layer: OS Services
Block: Comms Services
Sub-block: Short Link Services
Component collection: OBEX
License Classification: Optional Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: OBEX-session protocol implementation for IrDA, Bluetooth and USB transports, supporting connections from simple beaming all the way to fully fledged synchronization technologies such as SyncML.

A.154 OMA Data Sync

Development Name: SYNCMLDSCLIENT
Build File Location: `/common/generic/SyncML/group/`
System Model Location:
Layer: Application Services
Component collection: Data Sync Services
License Classification: Optional Replaceable

Exposes Third-Party APIs: NO
Present in OS Releases: 9.1, 9.2, 9.3
Description: SyncML client that manages the device side of a SyncML data-exchange session.

A.155 OMA SyncML DM Interface

Development Name: SYNCMLDMCLIENT
Build File Location: `/common/generic/SyncML/group/`
System Model Location:
Layer: Application Services
Component collection: Data Sync Services
License Classification: Optional Replaceable
Exposes Third-Party APIs: NO
Present in OS Releases: 9.1, 9.2, 9.3
Description: Supplies an interface to the SyncML Framework for use during a SyncML device-management session.

A.156 OMA SyncML Framework

Development Name: SYNCMLCLIENT
Build File Location: `common/generic/SyncML/group/`
System Model Location:
Layer: Application Services
Component collection: Data Sync Services
License Classification: Optional Replaceable
Exposes Third-Party APIs: YES
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: Client–server-based framework that supports SyncML data synchronization and device management over HTTP, WSP and OBEX. Follows the SyncML v1.1.2 specification including large object support, Server Alerted Notification and transactional behavior. Clients may provide plug-ins to manage device-management settings.

A.157 OMAP 2420

Development Name: OMAP2420
Build File Location: `/cedar/generic/base/omap_hrp/assp/`
System Model Location:
Layer: Kernel Services and Hardware Interface
Block: Kernel Architecture

Component collection: ASSP

License Classification: Optional Replaceable

Exposes Third-Party APIs: NO

Present in OS Releases: 9.2, 9.3

Description: ASSP support for the Texas Instruments H4 development board with OMAP 2420 (ARMv6-based core). Hardware reference platform for Symbian OS releases from Symbian OS v9.2.

A.158 OMAP H2

Development Name: OMAP_H2

Build File Location: `/cedar/generic/base/omap/h2/`

System Model Location:

Layer: Kernel Services and Hardware Interface

Block: Kernel Architecture

Component collection: Variant

License Classification: Optional Replaceable

Exposes Third-Party APIs: NO

Present in OS Releases: 9.1, 9.2, 9.3

Description: Variant code for the Texas Instruments H2 development board.

A.159 OMAP H4

Development Name: OMAPH4HRP

Build File Location: `/cedar/generic/base/omap_hrp/h4/`

System Model Location:

Layer: Kernel Services and Hardware Interface

Block: Kernel Architecture

Component collection: Variant

License Classification: Optional Replaceable

Exposes Third-Party APIs: NO

Present in OS Releases: 9.1, 9.2, 9.3

Description: Variant code for the Texas Instruments H4 development board.

A.160 OpenGL ES

Development Name: OPENGLES9.X

Build File Location: `/common/generic/graphics/OpenGLES/group/`

System Model Location:
Layer: OS Services
Block: Multimedia and Graphics Services
Component collection: OpenGL ES
License Classification: Reference/Test
Exposes Third-Party APIs: NO
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: Reference implementation of OpenGL ES, replaced by licensees. Provides multi-client access to screen, keyboard, and pointer or digitizer for GUI applications.

A.161 OpenGL ES Display Properties

Development Name: OPENGLESDISPLAYPROPERTY
Build File Location: `/common/generic/graphics/OpenGLESDisplayProperty/group/`
System Model Location:
Layer: OS Services
Block: Multimedia and Graphics Services
Component collection: OpenGL ES
License Classification: Optional Symbian
Exposes Third-Party APIs: NO
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: Encapsulates display-drawing properties (e.g. display rectangles and clipping regions), enabling window-surface access, i.e. drawing, to clients from non-window-owning threads.

A.162 OpenGL ES Headers

Development Name: OPENGLSHEADERS
Build File Location: `/common/generic/graphics/OpenGLESHeaders/group/`
System Model Location:
Layer: OS Services
Block: Multimedia and Graphics Services
Component collection: OpenGL ES
License Classification: Optional Symbian
Exposes Third-Party APIs: YES

Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: Standard OpenGL ES headers and binary definition files to encourage compatibility between OpenGL ES implementations for Symbian OS.

A.163 Other LDDs

System Model Location:
Layer: Kernel Services and Hardware Interface
Block: Kernel Architecture
Component collection: Logical Device Drivers
License Classification: Common Replaceable
Exposes Third-Party APIs: NO
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Drivers supporting hardware devices, implemented as Symbian OS logical device drivers (LDDs).

A.164 Peripheral Bus Controllers

Development Name: EPBUS
Build File Location: `/cedar/generic/base/e32/`
System Model Location:
Layer: Kernel Services and Hardware Interface
Block: Kernel Architecture
Component collection: Variant
License Classification: Common Symbian
Exposes Third-Party APIs: NO
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Peripheral-bus controllers for supported variants implemented as a kernel-side DLL interfacing media and I/O device drivers to PC-card or MMC-card-socket hardware.

A.165 Phonebook Sync

Development Name: PHBKSYNC
Build File Location: `/generic/telephony/phbksync/group/`
System Model Location:
Layer: OS Services
Block: Comms Services

Sub-block: Telephony Services

Component collection: Telephony Utilities

License Classification: Optional Replaceable

Exposes Third-Party APIs: YES

Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: Server enabling synchronization of contacts between a phonebook application and entries stored in the Integrated Circuit Card (ICC), or SIM card, of a device.

A.166 PLP Variant

Development Name: PLPVARIANT, PLP, BRDCST

Build File Location: `/common/generic/connectivity/legacy/plp/PLPVARIANT/`

System Model Location:

Layer: OS Services

Block: Connectivity Services

Component collection: Service Providers

License Classification: Optional Replaceable

Exposes Third-Party APIs: N/A

Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: Deprecated legacy component previously used as the bearer for connectivity services. Retained only for compatibility with third-party components that use some of its APIs.

A.167 Plug-in Framework (ECOM)

Development Name: ECOM

Build File Location: `/common/generic/syslibs/ecom/`

System Model Location:

Layer: Base Services

Component collection: Low-Level Libraries and Frameworks

License Classification: Optional Symbian

Exposes Third-Party APIs: YES

Present in OS Releases: 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: Framework and server for plug-in interface implementations. Defines the base classes used by conforming plug-ins and a client-side API used by framework clients to locate and load plug-ins on demand, in conformance with security-policy mechanisms.

A.168 POP3 MTM

Development Name: MSG_EMAIL
Build File Location: `/common/generic/messaging/email/popservermtm/group/`
System Model Location:
Layer: Application Services
Component collection: Messaging Application Support
License Classification: Optional Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: MTM plug-in to Messaging Store framework supporting send/receive/edit of POP3 (dial-up) email messages.

A.169 Power and Shutdown Management

Development Name: PWRCLI
Build File Location: `/common/generic/syslibs/pwrcli/group/`
System Model Location:
Layer: Base Services
Component collection: Low-Level Libraries and Frameworks
License Classification: Common Symbian
Exposes Third-Party APIs: NO
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: Customizable user-side power manager supporting policy-driven power management via power-domain 'profiles' at device switch-on and switch-off.

A.170 PPP Compression Plug-ins

Development Name: PREDCOMP, STACCOMP, MSCOMP
Build File Location: `/common/generic/networking/predcomp/group/`, `/common/generic/networking/staccomp/group/`, `/common/generic/networking/mscomp/group/`
System Model Location:
Layer: OS Services
Block: Comms Services
Sub-block: Networking Services
Component collection: Link Layer Control
License Classification: Optional Replaceable

Exposes Third-Party APIs: NO
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: PPP NIF (Point to Point Protocol) plug-ins that implement Predictor, Stac and Microsoft compression algorithms.

A.171 PPP NIF

Development Name: PPP
Build File Location: `/common/generic/networking/ppp/group/`
System Model Location:
Layer: OS Services
Block: Comms Services
Sub-block: Networking Services
Component collection: Link Layer Control
License Classification: Optional Replaceable
Exposes Third-Party APIs: NO
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Point-to-Point Protocol plug-in to Network Interface Manager. Supports TCP/IP over serial communications.

A.172 Printer Drivers

Development Name: PRINTDRV
Build File Location: `/common/generic/graphics/printdrv/group/`
System Model Location:
Layer: OS Services
Block: Multimedia and Graphics Services
Component collection: Graphics and Printing Services
License Classification: Optional Replaceable
Exposes Third-Party APIs: NO
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Reference implementation for concrete printer driver plug-ins. Considered legacy for most mobile phones.

A.173 Printing Services

Development Name: PRINT
Build File Location: `/common/generic/app-framework/print/group/`

System Model Location:

Layer: Application Services

Component collection: Printing Support

License Classification: Optional Replaceable

Exposes Third-Party APIs: YES

Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: Framework providing standard print dialogs for print-job setup and control to application clients. Considered legacy for most mobile phones.

A.174 Printing Support

Development Name: PDRSTORE

Build File Location: `/common/generic/graphics/pdrstore/group/`

System Model Location:

Layer: OS Services

Block: Multimedia and Graphics Services

Component collection: Graphics and Printing Services

License Classification: Optional Replaceable

Exposes Third-Party APIs: YES

Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: Framework that manages and loads printer drivers as bitmapped-device-context implementations and manages access to printer ports. Considered legacy for most mobile phones.

A.175 PSD AGT

Development Name: PSDAGT

Build File Location: `/common/generic/networking/psdagt/group/`

System Model Location:

Layer: OS Services

Block: Comms Services

Sub-block: Networking Services

Component collection: Networking Plug-ins

License Classification: Optional Replaceable

Exposes Third-Party APIs: N/A

Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3

Description: Deprecated functionality replaced by other components. AGT agent plug-in to Connection Agent framework that negotiates packet-switched connection, for example to GPRS networks.

A.176 QoS Framework PRT

Development Name: QOS

Build File Location: `/common/generic/networking/common/generic/networking/qoslib/group/`

System Model Location:

Layer: OS Services

Block: Comms Services

Sub-block: Networking Services

Component collection: Network Protocol Plug-ins

License Classification: Optional Replaceable

Exposes Third-Party APIs: NO

Present in OS Releases: 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: QoS Framework and library modules, implemented as a PRT protocol plug-in to Socket Server.

A.177 Raw IP NIF

Development Name: RAWIPNIF

Build File Location: `/common/generic/networking/rawipnif/group/`

System Model Location:

Layer: OS Services

Block: Comms Services

Sub-block: Networking Services

Component collection: Link Layer Control

License Classification: Optional Replaceable

Exposes Third-Party APIs: NO

Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3

Description: NIF plug-in to Network Interface Manager that supports multiple primary-PDP contexts, i.e. multi-homing over GPRS, on the telephony-reference platform.

A.178 Reference DRM Agent

Development Name: DRMAGENT
Build File Location: `/common/generic/syslibs/caf2/group/`
System Model Location:
Layer: Application Services
Component collection: Content Handling
License Classification: Reference/Test
Exposes Third-Party APIs: NO
Present in OS Releases: 9.1, 9.2, 9.3
Description: Reference implementation of DRM-agent plug-in to Content Access Framework for DRM.

A.179 Reference Fonts

Development Name: FONTS
Build File Location: `/common/generic/graphics/fonts/group/`
System Model Location:
Layer: OS Services
Block: Multimedia and Graphics Services
Exposes Third-Party APIs: NO
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: Reference native fonts for Symbian OS. Replaced by licensees.

A.180 Remote Control Framework

Development Name: BLUETOOTH_REMOTECONTROL
Build File Location: `/common/generic/bluetooth/`
System Model Location:
Layer: OS Services
Block: Comms Services
Sub-block: Short Link Services
Component collection: Short Link
License Classification: Optional Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 9.1, 9.2, 9.3
Description: Bearer-agnostic remote-control framework that enables sending and receiving of remote-control commands to and from remote Bluetooth devices using bearers provided as plug-ins to the framework.

A.181 Remote File Server

Development Name: SCREMOTEFILESERVER
Build File Location: `/common/generic/connectivity/SCRemoteFileServer/group/`
System Model Location:
Layer: OS Services
Block: Connectivity Services
Component collection: Service Providers
License Classification: Optional Replaceable
Exposes Third-Party APIs: YES
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: Named service providing on-device file-management functions to a remote (off-device) client over TCP/IP including access to backup and restore functions provided by other system components.

A.182 RTP

Development Name: RTP
Build File Location: `/common/generic/mm-protocols/rtp/group/`
System Model Location:
Layer: Application Services
Component collection: Multimedia Protocols
License Classification: Optional Replaceable
Exposes Third-Party APIs: YES
Present in OS Releases: 9.1, 9.2, 9.3
Description: Server and user-side API providing socket-based access to Real-Time Transport Protocol (RTP) services, providing an IP-based real-time-network transport service.

A.183 Runtime Plug-in

Development Name: MIDP2RUNTIME
Build File Location: `/common/generic/ME/group/`
System Model Location:
Layer: Java ME
Component collection: Low-Level Plug-ins
License Classification: Common Replaceable

Exposes Third-Party APIs: YES
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: Licensee-customizable MIDP 2.0 runtime plug-in module.

A.184 Scheduled Send MTM

Development Name: MSG_SCHEDULEDSEND
Build File Location: `/common/generic/messaging/schedulesend/group/`
System Model Location:
Layer: Application Services
Component collection: Messaging Application Support
License Classification: Optional Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: MTM plug-in to Messaging Store framework that supports scheduled sending of any available message type including SMS and fax and defines the scheduling parameters.

A.185 Screen Driver

Development Name: SCREENDRIVER
Build File Location: `/common/generic/graphics/screendriver/group/`
System Model Location:
Layer: Kernel Services and Hardware Interface
Component collection: Screen Driver
License Classification: Common Replaceable
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Device-dependent component that implements the generic operations defined by the Bit GDI to manipulate the physical memory map of the device display or bitmap memory map. Note that this is not implemented as a standard Symbian OS device driver.

A.186 SD Card Driver

Development Name: SDCARD4C
Build File Location: `/cedar/generic/base/e32/drivers/`

System Model Location:
Layer: Kernel Services and Hardware Interface
Block: Kernel Architecture
Component collection: Logical Device Drivers
Component collection: Logical Device Drivers
License Classification: Common Replaceable
Exposes Third-Party APIs: NO
Present in OS Releases: 9.1, 9.2, 9.3
Description: Logical and physical device drivers supporting Secure Digital flash-memory cards.

A.187 Secondary PDP context UMTS Driver

Development Name: SPUD
Build File Location: `/common/generic/networking/Spud/group/`
System Model Location:
Layer: OS Services
Block: Comms Services
Sub-block: Networking Services
Component collection: Networking Plug-ins
License Classification: Optional Replaceable
Exposes Third-Party APIs: NO
Present in OS Releases: 9.1, 9.2, 9.3
Description: Network-interface-adapter component that supports primary and secondary PDP contexts (multi-homing over GPRS) on the telephony-reference platform only.

A.188 Secure Backup Engine

Development Name: SECUREBACKUPENGINE
Build File Location: `/common/generic/connectivity/SecureBackupEngine/group/`
System Model Location:
Layer: OS Services
Block: Connectivity Services
Component collection: Service Providers
License Classification: Optional Replaceable
Exposes Third-Party APIs: YES
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3

Description: Manages backup and restore of device-side data, as controlled by the Secure Backup Socket Server.

A.189 Secure Backup Socket Server

Development Name: SBSSERVER
Build File Location: `/common/generic/connectivity/SBSocketServer/group/`
System Model Location:
Layer: OS Services
Block: Connectivity Services
Component collection: Service Providers
License Classification: Optional Replaceable
Exposes Third-Party APIs: NO
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: Named service providing backup–restore functions to a remote (off-device) client over TCP/IP. Communicates with a connected PC and with the Secure Backup Engine to carry out backup and restore operations to a PC.

A.190 Secure Software Install

Development Name: SECURESOFTWAREINSTALL
Build File Location: `/common/generic/security/swi/group/`
System Model Location:
Layer: Application Services
Component collection: Application Framework
License Classification: Optional Symbian
Exposes Third-Party APIs: NO
Present in OS Releases: 9.1, 9.2, 9.3
Description: Installers for native and Java apps.

A.191 Security Policy Reference Plug-in

Development Name: MIDP2SECURITY
Build File Location: `/common/generic/ME/group/`
System Model Location:
Layer: Java ME
Component collection: MIDP 2.0 Profile
License Classification: Optional Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3

Description: Reference implementation of Java security policy, implemented as a replaceable plug-in.

A.192 Send As

Development Name: SENDASV2

Build File Location: `/common/generic/messaging/sendas2/group/`

System Model Location:

License Classification: Optional Symbian

Exposes Third-Party APIs: YES

Present in OS Releases: 9.1, 9.2, 9.3

Description: Not shown explicitly in the model and forming part of the messaging framework, from Symbian OS v9, a client–server architecture enabling message sending from within applications.

A.193 Serial Port CSY

Development Name: ECUART

Build File Location: `/common/generic/ser-comms/c32/group/`

System Model Location:

Layer: OS Services

Block: Comms Services

Sub-block: Short Link Services

License Classification: Common Replaceable

Exposes Third-Party APIs: YES

Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: CSY plug-in to C32 Serial Server providing virtual serial port.

A.194 Server Socket

Development Name: SERVERSOCKET

Build File Location: `/common/generic/connectivity/ServerSocket/group/`

System Model Location:

Layer: OS Services

Block: Connectivity Services

Component collection: Service Providers

License Classification: Optional Replaceable

Exposes Third-Party APIs: YES

Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3

Description: Helper library that communicates service-port numbers and manages messages and commands, for use by the Service Broker.

A.195 Service Broker

Development Name: SERVICEBROKER

Build File Location: `/common/generic/connectivity/ServiceBroker/group/`

System Model Location:

Layer: OS Services

Block: Connectivity Services

Component collection: Service Framework

License Classification: Optional Symbian

Exposes Third-Party APIs: YES

Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3

Description: Configuration-file-based service and port registration that enables device-side services to register a port number for use by PC-side clients.

A.196 Sheet Engine

Development Name: SHENG

Build File Location: `/common/generic/app-engines/sheng/group/`

System Model Location:

Layer: Application Services

Component collection: Office Application Engines

License Classification: Optional Symbian

Exposes Third-Party APIs: YES

Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: Legacy application engine supporting a spreadsheet application, originating from the early EPOC built-in applications set.

A.197 SIM TSY

Development Name: SIMTSY

Build File Location: `/common/generic/telephony/simtsy/group/`

System Model Location:

Layer: OS Services

Block: Comms Services

Sub-block: Telephony Services

Component collection: Telephony Server Plug-ins

License Classification: Reference/Test

Exposes Third-Party APIs: NO

Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3

Description:TSY simulator module that uses static configuration data and dynamic system-agent notifications to simulate the presence of phone hardware. Implemented as a TSY plug-in to the ETel Telephony Server.

A.198 SIP Connection Provider Plug-ins

Development Name: SIPCPRT, SIPDUMMYPRT, SIPSTATEMAC, SIPPARAMS, SIPSCPRT

Build File Location: `/common/generic/mm-protocols/connprov/sipcpr/group/`

System Model Location:

Layer: Application Services

Component collection: Multimedia Protocols

License Classification: Optional Replaceable

Exposes Third-Party APIs: YES

Present in OS Releases: 9.2, 9.3

Description: Network-layer connection provision for Session Initiation Protocol (SIP).

A.199 SIP Framework

Development Name: SIP_COM

Build File Location: `/common/generic/mm-protocols/sip/group/`

System Model Location:

Layer: Application Services

Component collection: Multimedia Protocols

License Classification: Optional Replaceable

Exposes Third-Party APIs: YES

Present in OS Releases: 9.1, 9.2, 9.3

Description: Framework providing Session Initiation Protocol (SIP) support and integration into the networking infrastructure, but not the protocol implementation (which is provided as a plug-in by licensees).

A.200 SLIP NIF

Development Name: SLIP

Build File Location: `/common/generic/networking/slip/group/`

System Model Location:

Layer: OS Services

Block: Comms Services

Sub-block: Networking Services

Component collection: Link Layer Control

License Classification: Reference/Test

Exposes Third-Party APIs: NO

Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: Reference implementation of Serial Line Internet Protocol (SLIP) NIF plug-in to Network Interface Manager providing TCP/IP over serial communications via modem dialup.

A.201 SMIL Parser

Development Name: GMXML

Build File Location: `/common/generic/messaging/gmxml/group/`

System Model Location:

Layer: Application Services

Component collection: Content Handling

License Classification: Optional Replaceable

Exposes Third-Party APIs: YES

Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3

Description: Parser for SMIL (and other simple XML) content based on a generic XML Parser/Composer with 'mini-DOM' API. Replaces the SMIL Translator implementation of Symbian OS v7.0s.

A.202 SMS MTM

Development Name: MSG_SMS8.1

Build File Location: `/common/generic/messaging/sms/`

System Model Location:

Layer: Application Services

Component collection: Messaging Application Support

License Classification: Optional Symbian

Exposes Third-Party APIs: YES

Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: MTM plug-in to Messaging Store framework that implements SMS-messaging support.

A.203 SMS PRT

Development Name: SMSSTACK

Build File Location: `/common/generic/nbprotocols/smsstackv2/smsprot/group/`

System Model Location:

Layer: OS Services
Block: Comms Services
Sub-block: Telephony Services
Component collection: SMS Protocol Plug-ins
License Classification: Common Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Socket Server PRT plug-in that implements SMS protocol.

A.204 SMS Utilities

Development Name: SMSU
Build File Location: `/common/generic/nbprotocols/smsstack/smsu/group/`
System Model Location:
Layer: OS Services
Block: Comms Services
Sub-block: Telephony Services
Component collection: SMS Utilities
License Classification: Common Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Utilities for processing SMS messages, includes streaming classes, logging support and interface to the backup server, used by SMS PRT and its clients.

A.205 SMTP MTM

Development Name: SMTPSERVERMTP
Build File Location: `/common/generic/messaging/email/SMTPSERVERMTm/group/`
System Model Location:
Layer: Application Services
Component collection: Messaging Application Support
License Classification: Optional Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: MTM plug-in to Messaging Store framework supporting sending, receiving and editing of SMTP (Internet) mail.

A.206 Software Install Server

Development Name: SWINSTALLSERVER
Build File Location: `/common/generic/connectivity/SWInstallServer/group/`
System Model Location:
Layer: OS Services
Block: Connectivity Services
Component collection: Service Providers
License Classification: Optional Replaceable
Exposes Third-Party APIs: YES
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: Named service that interacts with the software-installation components on the device to enable remote installation of SIS, JAR and JAD files over TCP/IP or OBEX.

A.207 Speech Driver

Development Name: DEVASR
Build File Location: `/common/generic/Multimedia/MMF/DEVASR/group/`
System Model Location:
Layer: Kernel Services and Hardware Interface
Block: Kernel Architecture
Component collection: Logical Device Drivers
License Classification: Common Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: Hardware-acceleration API for Automatic Speech Recognition that allows the computationally-intensive speech recognition algorithms to be performed in hardware, where present.

A.208 Subconnection Parameters

Development Name: UMTSIF
Build File Location: `/common/generic/networking/`
System Model Location:
Layer: OS Services

Block: Comms Services
Sub-block: Networking Services
Component collection: Subconnection Interface

License Classification: Common Replaceable

Exposes Third-Party APIs: NO

Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: QoS parameters and Traffic Flow Templates for GPRS.

A.209 Store

Development Name: STORE

Build File Location: `/common/generic/syslibs/store/group/`

System Model Location:

Layer: Base Services
Component collection: Low-Level Libraries and Frameworks

License Classification: Common Replaceable

Exposes Third-Party APIs: YES

Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: Defines the Symbian OS persistence model based on the streams and stores abstractions, providing an application data-storage model that shields applications from the underlying file-server implementation.

A.210 Sync Initiation

Development Name: SYNCMLINITSERVER

Build File Location: `/common/generic/connectivity/SyncMLInitServer/group/`

System Model Location:

Layer: Application Services
Component collection: Data Sync Services

License Classification: Optional Replaceable

Exposes Third-Party APIs: YES

Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3

Description: Allows a synchronization SyncML operation to be initiated from the PC. Note that, although Symbian does not supply a PC-side SyncML server, this service allows the creation of such a service by partners.

A.211 System Agent

Development Name: SYSAGENT2
Build File Location: `/common/generic/syslibs/sysagent2/group/`
System Model Location:
Layer: OS Services
Block: Generic OS Services
Component collection: Generic Services
License Classification: Optional Replaceable
Exposes Third-Party APIs: NO
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Legacy component whose functionality is largely replaced by Publish and Subscribe.

A.212 System Starter

Development Name: SYSSTART
Build File Location: `/common/generic/app-framework/SysStart/group/`
System Model Location:
Layer: Application Services
Component collection: Application Launch Services
License Classification: Optional Replaceable
Exposes Third-Party APIs: NO
Present in OS Releases: 9.1, 9.2, 9.3
Description: Server and framework enabling policy-based startup of system servers at boot time.

A.213 Task Scheduler

Development Name: SCHSVR_ONGOING
Build File Location: `/common/generic/syslibs/schsvr/`
System Model Location:
Layer: OS Services
Block: Generic OS Services
Component collection: Generic Services
License Classification: Optional Replaceable
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: Task or executable launching service for time-based and condition-based task triggers.

A.214 TCP/IPv4/v6 PRT

Development Name: TCPIP6
Build File Location: `/common/generic/networking/tcpip6/group/`
System Model Location:
Layer: OS Services
Block: Comms Services
Sub-block: Networking Services
Component collection: Network Protocol Plug-ins
License Classification: Common Symbian
Exposes Third-Party APIs: NO
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: IPv4/v6 protocol implementation and TCP/IP stack and plug-in extension architecture, implemented as a PRT protocol plug-in to ESock Socket Server.

A.215 Telephony Watchers

Development Name: TELEPHONY_WATCHERS
Build File Location: `/common/generic/telephony/Watchers/group/`
System Model Location:
Layer: OS Services
Block: Comms Services
Sub-block: Telephony Services
Component collection: Telephony Utilities
License Classification: Optional Replaceable
Exposes Third-Party APIs: NO
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: Watcher Framework plug-ins that monitor telephony conditions and report them as Publish and Subscribe properties.

A.216 Telnet Engine

Development Name: TELNET_E
Build File Location: `/common/generic/networking/telnet_e/group/`

System Model Location:
Layer: Application Services
Component collection: Internet and Web Application Support
License Classification: Optional Replaceable
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Symbian OS Telnet daemon implementation that supports client sessions for communicating with a specified host.

A.217 Text Formatting (FORM)

Development Name: FORM
Build File Location: `/common/generic/app-framework/form/group/`
System Model Location:
Layer: Application Services
Component collection: Text Rendering
License Classification: Optional Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Text view and layout classes that support the separation of display attributes (layout and drawing) from logical text attributes (styles).

A.218 Text Handling (ETEXT)

Development Name: ETEXT
Build File Location: `/common/generic/app-framework/etext/group/`
System Model Location:
Layer: Application Services
Component collection: Text Rendering
License Classification: Optional Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Text-content framework that supports storing of editable text and its logical attributes, for example paragraph alignment and character fonts.

A.219 Text Shaper Plug-in

Development Name: ICULAYOUTENGINE
Build File Location: `/common/generic/graphics/iculayoutengine/group/`
System Model Location:
Layer: OS Services
Block: Multimedia and Graphics Services
Component collection: Graphics and Printing Services
License Classification: Optional Replaceable
Exposes Third-Party APIs: NO
Present in OS Releases: 9.1, 9.2, 9.3
Description: Plug-in to Font and Bitmap Server that supports rendering text in Devanagari script.

A.220 Text Shell

Development Name: ESHELL
Build File Location: `/cedar/generic/base/f32/etshell/group/`
System Model Location:
Layer: Base Services
Component collection: Text Mode Shell
License Classification: Test/Reference
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Text shell providing a command-line interface to the base system.

A.221 Text Window Server

Development Name: EWSRV
Build File Location: `/cedar/generic/base/e32/ewsrv/`
System Model Location:
Layer: Base Services
Component collection: Text Mode Shell
License Classification: Optional Replaceable
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: Text-mode window server supporting the Text Shell. Uses a text-mode display driver providing VT100 terminal emulation over a serial line and VGA/LCD implementations for reference hardware.

A.222 Timezone

Development Name: TZ, TZLOCALIZATIONRSCFACTORY
Build File Location: `/common/generic/app-services/tz/group/`
System Model Location:
Layer: Application Services
Component collection: Other Application Services
License Classification: Optional Replaceable
Exposes Third-Party APIs: YES
Present in OS Releases: 9.1, 9.2, 9.3
Description: Localization support including time-zone database for Standard, Daylight, Short Standard and Short Daylight names for time zones. Associated components not explicitly shown in the model include a time-zone compiler, database and localization tools.

A.223 TLS

Development Name: TLS
Build File Location: `/common/generic/networking/tls/group/`
System Model Location:
Layer: OS Services
Block: Comms Services
Sub-block: Networking Services
Component collection: TCP/IP Security
License Classification: Common Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Implements Secure Sockets Layer (SSL 3.0) and Transport Level Security (TLS 1.0) protocols, enabling secure network connections.

A.224 TRP CSY

Development Name: TRP
Build File Location: `/common/generic/telephony/trp/csy/csy27010/group/`
System Model Location:
Layer: OS Services

Block: Comms Services
Sub-block: Telephony Services
Component collection: Telephony Reference Platform

License Classification: Common Replaceable

Exposes Third-Party APIs: YES

Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: CSY implementation for the Telephony Reference Platform that manages the internal channel between the telephony and application hardware as a standard serial port.

A.225 TRP TSY

Development Name: TRP

Build File Location: `/common/generic/telephony/trp/tsy/group/`

System Model Location:

Layer: OS Services
Block: Comms Services
Sub-block: Telephony Services
Component collection: Telephony Reference Platform

License Classification: Common Replaceable

Exposes Third-Party APIs: NO

Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: TSY implementation for the Telephony Reference Platform.

A.226 Tunnel NIF

Development Name: TUNNELNIF

Build File Location: `/common/generic/networking/tunnelnif/group/`

System Model Location:

Layer: OS Services
Block: Comms Services
Sub-block: Networking Services
Component collection: Link Layer Control

License Classification: Optional Replaceable

Exposes Third-Party APIs: NO

Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3

Description: Ethernet protocol NIF plug-in to Network Interface Manager supporting IPSec tunneling capability. Forms part of the VPN client and is used when running IPSec in tunnel mode.

A.227 UI Graphics Utilities

Development Name: EGUL, NUMBERCONVERSION
Build File Location: `/common/generic/app-framework/egul/group/`
System Model Location:
Layer: UI Framework
Component collection: UI Support
License Classification: Common Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Libraries used by UI-framework components as well as the variant UI and applications, providing color, font, icon, text, drawing and number-conversion utilities.

A.228 UI Look and Feel

Development Name: UIKLAFGT
Build File Location: `/common/generic/app-framework/uiklafGT/group/`
System Model Location:
Layer: UI Framework
Component collection: UI Application Framework
License Classification: Common Symbian
Exposes Third-Party APIs: NO
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: Reference-implementation plug-in to Uikon framework that determines the look and feel of Control Environment controls by defining standard methods for which UI customizers provide a custom implementation.

A.229 Uikon

Development Name: UIKON
Build File Location: `/common/generic/app-framework/uikon/group/`
System Model Location:
Layer: UI Framework
Component collection: UI Application Framework
License Classification: Common Symbian
Exposes Third-Party APIs: YES

Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Concrete framework for UI and application creation, providing the foundation for licensee UI customization.

A.230 Uikon Error Resolver Plug-in

Development Name: ERRORRESGT
Build File Location: `/common/generic/app-framework/errorresgt/group/`
System Model Location:
Layer: UI Framework
Component collection: UI Application Framework
License Classification: Common Symbian
Exposes Third-Party APIs: NO
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Resource file that maps system error numbers to helpful error-text strings, extended and customized by variant UIs.

A.231 USB CSY

Development Name: ECACM
Build File Location: `/generic/ser-comms/usb/CSY/group/`
System Model Location:
Layer: OS Services
Block: Comms Services
Sub-block: Short Link Services
Component collection: Serial Comms Server Plug-ins
License Classification: Common Replaceable
Exposes Third-Party APIs: NO
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: CSY plug-in to C32 Serial Server providing serial-port emulation over USB.

A.232 USB Driver

Development Name: USBC
Build File Location: `/cedar/generic/base/e32/`
System Model Location:
Layer: Kernel Services and Hardware Interface

Block: Kernel Architecture

Component collection: Logical Device Drivers

License Classification: Common Replaceable

Exposes Third-Party APIs: YES

Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: Logical Device Driver for USB supporting dynamically configurable USB 2.0 Full Speed device functionality.

A.233 USB Manager

Development Name: USB

Build File Location: `/common/generic/ser-comms/usb/usbman/group/`

System Model Location:

Layer: OS Services

Block: Comms Services

Sub-block: Comms Framework

Component collection: Data Comms Server

License Classification: Optional Symbian

Exposes Third-Party APIs: NO

Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: USB class support that enables a Symbian OS device to both use and serve as a USB host.

A.234 User HAL

Development Name: HAL_EKA2

Build File Location: `/cedar/generic/base/hal/`

System Model Location:

Layer: Base Services

Block: User-Side Hardware Abstraction

Sub-block: User-Side Hardware Abstraction

Component collection: User-Side Hardware Abstraction

License Classification: Common Symbian

Exposes Third-Party APIs: NO

Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: User-side access to hardware via hardware abstraction, deprecated for application use.

A.235 User Library

Development Name: EUSER
Build File Location: `/cedar/generic/base/e32/group/`
System Model Location:
Layer: Base Services
Component collection: User Library and File Server
License Classification: Common Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Symbian OS user library, used by all user-side code to provide fundamental basic services.

A.236 vCal Plug-in

Development Name: AGNVERSIT
Build File Location: `/common/generic/app-engines/agnversit/group/`
System Model Location:
Layer: Application Services
Component collection: PIM Application Services
License Classification: Optional Replaceable
Exposes Third-Party APIs: Yes
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: Limited public APIs, for example in `vcal.h`. Plug-in library used by Agenda Model to interact with the vCard and vCal component.

A.237 vCard and vCal

Development Name: VERSIT
Build File Location: `/common/generic/app-services/versit/group/`
System Model Location:
Layer: Application Services
Component collection: PIM Application Support
License Classification: Optional Replaceable
Exposes Third-Party APIs: YES
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3

Description: Parsers that convert between vCard and vCalendar entries and Symbian OS native formats.

A.238 Video Driver

Development Name: DEVVIDEO
Build File Location: `/common/generic/Multimedia/MMF/DevVideo/group/`
System Model Location:
Layer: Kernel Services and Hardware Interface
Block: Kernel Architecture
Component collection: Logical Device Drivers
License Classification: Common Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: Hardware-abstraction layer for video decoding and encoding acceleration.

A.239 View Server

Development Name: VIEWSRV
Build File Location: `/common/generic/app-framework/viewsrv/group/`
System Model Location:
Layer: Application Services
Component collection: Application Framework
License Classification: Common Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Provides a mechanism for sharing and switching views between applications. A running application can switch into and use a view belonging to another application.

A.240 VPN

Development Name: VPNAPI, VPNCONNAGT, VPNMANAGER
Build File Location: `/common/generic/networking/ipsec/vpnapi/group/`, `/common/generic/ipsec/vpnconnagt/group/`, `/common/generic/ipsec/vpnmanager/group/`
System Model Location:
Layer: OS Services

Block: Comms Services
Sub-block: Networking Services
Component collection: TCP/IP Security

License Classification: Optional Replaceable

Exposes Third-Party APIs: YES

Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3

Description: Key negotiation and tunneling using IPSec for VPN connections, enabling users to connect to VPNs.

A.241 WAP Message API

Development Name: WAPMESSAGE

Build File Location: `/common/generic/wap-stack/wapmessage/group/`

System Model Location:

Layer: OS Services
Block: Comms Services
Sub-block: Networking Services
Component collection: WAP Stack

License Classification: Optional Replaceable

Exposes Third-Party APIs: YES

Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: APIs for WAP Push, connectionless WSP and WDP datagrams.

A.242 WAP Push Framework

Development Name: WAPPUSH

Build File Location: `/common/generic/wap-browser/wappush/group/`

System Model Location:

Layer: Application Services
Component collection: Internet and Web Application Support

License Classification: Optional Replaceable

Exposes Third-Party APIs: NO

Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: Provides an interface between the WAP stack and the messaging infrastructure to support WAP as a messaging transport. WAP Push embeds links to WAP addresses within SMS messages.

A.243 WAP Push Handlers

Development Name: WAPPUSHSUPPORT
Build File Location: `/common/generic/wap-browser/WapPushSupport/group/`
System Model Location:
Layer: Application Services
Component collection: Content Handling
License Classification: Optional Replaceable
Exposes Third-Party APIs: NO
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: Plug-in handlers, including Several Interfaces Single Logic (SISL), that use the WAP message API.

A.244 WAP Push MTM

Development Name: WAP-BROWSER
Build File Location: `/common/generic/wap-browser/wappush/pushmtm/group/`
System Model Location:
Layer: Application Services
Component collection: Messaging Application Support
License Classification: Optional Replaceable
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: MTM plug-in to Messaging Store framework supporting WAP messaging.

A.245 WAP Short Stack

Development Name: WAPSTACK
Build File Location: `/common/generic/wap-stack/wapmessage/group/`
System Model Location:
Layer: OS Services
Block: Comms Services
Sub-block: Networking Services
Component collection: WAP Stack
License Classification: Optional Replaceable

Exposes Third-Party APIs: NO
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: Shortened WAP stack supporting WDP, WAP Push and Connectionless WSP over IP or SMS, implemented as an ESock Socket Server plug-in. May be replaced by licensee implementation.

A.246 WBXML Parser

Development Name: WBXMLPARSER
Build File Location: `/common/generic/syslibs/xml/group/`
System Model Location:
Layer: Base Services
Component collection: Low-Level Libraries and Frameworks
License Classification: Optional Replaceable
Exposes Third-Party APIs: YES
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: Plug-in to XML Parser Framework that parses WAP Binary XML.

A.247 Web Recognizers

Development Name: RECOGNISERS
Build File Location: `/common/generic/application-protocols/recognisers/group/`
System Model Location:
Layer: Application Services
Component collection: Content Handling
License Classification: Optional Replaceable
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Web URL and bookmark recognizers implemented as plug-ins to the MIME Recognizer Framework.

A.248 Window Server

Development Name: WSERV8.1
Build File Location: `/common/generic/graphics/wserv/group/`
System Model Location:
Layer: OS Services

Block: Multimedia and Graphics Services
Component collection: Windowing Framework

License Classification: Common Symbian

Exposes Third-Party APIs: YES

Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3

Description: Server that owns and manages access to the screen as a drawable resource, making it available to applications through the abstraction of windowed screen areas, and access to the keyboard and pointer or digitizer for GUI applications. Includes the keyclick reference plug-in that produces key or pointer clicks.

A.249 Wireless LAN

Development Name: WIFI_802_11

Build File Location: `/common/generic/networking/802.11/group/`

System Model Location:

Layer: OS Services
Block: Comms Services
Sub-block: Networking Services
Component collection: Link Layer Control

License Classification: Optional Replaceable

Exposes Third-Party APIs: NO

Present in OS Releases: 9.3

Description: Support for wireless LAN based on the IEEE 802.11 specifications.

A.250 WMA 1.1

Development Name: WMA

Build File Location: `/common/generic/ME/group/`

System Model Location:

Layer: Java ME
Component collection: MIDP 2.0 Packages

License Classification: Optional Replaceable

Exposes Third-Party APIs: YES

Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3

Description: MIDP Wireless Messaging 1.1 (JSR-120) APIs supporting SMS and MMS, including Push support.

A.251 WMA 1.1 Push Plug-in

Development Name: WMA
Build File Location: `/common/generic/ME/group/`
System Model Location:
Layer: Java ME
Component collection: Bluetooth and SMS Push
License Classification: Optional Replaceable
Exposes Third-Party APIs: YES
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: Plug-in that binds the WMA 1.1 package to the underlying system.

A.252 Word Engine

Development Name: WPENG
Build File Location: `/common/generic/app-engines/wpeng/group/`
System Model Location:
Layer: Application Services
Component collection: Office Application Engines
License Classification: Optional Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Legacy application engine supporting a word-processor application, originating from the early EPOC built-in applications suite.

A.253 World Server

Development Name: WORLDSERVER
Build File Location: `/common/generic/app-services/worldserver/group/`
System Model Location:
Layer: Application Services
Component collection: Other Application Services
License Classification: Optional Replaceable
Exposes Third-Party APIs: NO
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Legacy server providing application access to country and city information, including country, capital city, international and city dialing codes, latitude, longitude, and UTC offset.

A.254 XML Framework

Development Name: XML
Build File Location: `/common/generic/syslibs/xml/group/`
System Model Location:
Layer: Base Services
Component collection: Low-Level Libraries and Frameworks
License Classification: Optional Replaceable
Exposes Third-Party APIs: YES
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: Extensible framework for XML parsing based on a SAX-2.0-like parser model and supporting DTD and processing plug-ins (e.g. validators and auto correctors) as well as parser plug-ins.

A.255 XML Parser

Development Name: XMLPARSERPLUGIN
Build File Location: `/common/generic/syslibs/xml/group/`
System Model Location:
Layer: Base Services
Component collection: Low-Level Libraries and Frameworks
License Classification: Optional Replaceable
Exposes Third-Party APIs: YES
Present in OS Releases: 8.0, 8.1, 9.1, 9.2, 9.3
Description: Non-validating parser plug-in for XML 1.0.

A.256 Zip Compression Library

Development Name: EZLIB
Build File Location: `/common/generic/syslibs/ezlib/group/`
System Model Location:
Layer: Base Services
Component collection: Low-Level Libraries and Frameworks
License Classification: Optional Symbian
Exposes Third-Party APIs: YES
Present in OS Releases: 7.0, 7.0s, 8.0, 8.1, 9.1, 9.2, 9.3
Description: Symbian OS port of the zlib compression library, implementing ZLIB, DEFLATE and GZIP for ZIP compression and decompression.

Appendix B: Interviewee Biographies

Geert Bollen

Before joining Symbian, Geert was involved in a startup in the then emerging Electronic Document Management field, where he integrated databases with optical storage technology and made them sit up and perform tricks

At Symbian, Geert initially led the design and implementation of persistent data services in Symbian OS and went on to create its first Java implementation – possibly the first for mobile phones. Following that, he held a wide range of engineering management roles. He is currently VP System Management with overall technical responsibility for Symbian OS.

Martin Budden

Before joining Symbian (then Psion), Martin worked in a variety of computer-related jobs. His first, in 1980, was as a 'Student Scientist' at the Royal Signals and Radar Establishment (now QinetiQ) where he worked on MASCOT (a parallel-processing system) and was exposed to email and the ARPANET. Martin holds an MA in Mathematics from the University of Cambridge.

Martin joined Symbian (then Psion) in 1990 and started working on the MC400 laptop. He worked on a variety of Psion products, including the Series 3 and Series 5 palmtops and then was Technical Lead on an early Philips project – the first mobile phone project to earn Symbian money. He spent a year in Symbian's Swedish offices in Ronneby helping