

John Pagonis
Symbian Developer Network

Symbian OS v9.x

Architectural impact for developers

Overview of what is **so** important in v9.x ?

1. Real Time Kernel
2. Platform Security
3. Modern ISO C++ enablement
4. New and hugely improved compilers
5. Multithreaded sockets server
6. Multithreaded file-system server
7. Symbian Signed

plus ... more support for emerging standards

- Digital Rights Management (DRM)
 - ... New DRM framework – based on plug-in architecture
 - ... Compliant with OMA DRM v2.0
- Device Management (DM)
 - ... Compliant with OMA DM v1.1.2 services
 - ... Compliant with OMA Client Provisioning v1.1
- Instant Messaging Service (IMS)
 - ... support for 3GPP r5 networks with IMS support
- Java
 - ... Support for JTWI (JSR 185) & PIM (JSR 075)
 - ... Performance improvements
- E-mail
 - ... IMAP mail filtering & sorting

...and more multimedia support

- Bluetooth stereo headset profile - streaming audio from a phone to a headset (Bluetooth GAVDP)
- Remotely controlling a music player from a phone (AVRCP) and a new Remote Control framework
- USB mass storage
- Audio mixing
- Multi-megapixel camera support

Major impacts we'll focus on today...

- EKA2 architectural enhancements

...we will focus on the new IPC mechanisms and their implications to application and middleware developers rather than the kernel internals

- Capability-based platform security model

... we will focus on the technology impact to application development and deployment, but not on the Symbian Signed programme

- Few things about our ISO C++ enablement

Architectural Evolution – EKA2

- New multi-threaded, pre-emptible Real Time Kernel
- In-fact is a Symbian OS personality on top of a Nanokernel – **many** personalities can co-exist ;-)
- O(1) scheduler
- System calls are all pre-emptible as well → dual stacks
- Deterministic ISR, thread response, latencies etc
- Memory models and local allocator strategies can be plugged-in
- Many more IPC/ITC mechanisms such as local/global message queues, publish-subscribe, global anonymous queues, shared I/O buffers

Kernel – Nano Kernel and personalities

- EKA2 splits the kernel in two layers
- A Nano Kernel and a Symbian OS Kernel
- The Symbian OS Kernel is still a Micro Kernel
- The Symbian OS Kernel is a “personality” on top of the NK
- There can be many such personalities simultaneously running, thus many kernels can be run pre-emptively on top of the Nano Kernel !!!!!
 - ... For example, one for the GSM or UMTS stack and one for Symbian OS
- NK is responsible for the very basic synchronisation, timing, initial interrupt handling and scheduling services
- It is not depended on EUser and doesn't know about processes or memory models
- All offered services are deterministic with bound execution times

Kernel pre-emptibility

- The EKA2 Symbian OS Kernel is multi-threaded
 - ... Device drivers are much easier to write
- It is completely pre-emptible
 - ... even the memory allocations and context switch can be pre-empted
- User side threads have a user mode and supervisor mode stack
 - ... executive calls run on user thread's supervisor stack
 - ... executive calls can thus all be pre-empted !!!!

Kernel memory models and allocators

- In EKA2 memory models are replaceable
 - ... moving (like in EKA1 ARMv4/v5), direct (no MMU), emulator and multiple (like in x86,ARMv6)
 - ... thus supports physically tagged cache as well for faster context switching
- In EKA2 Symbian OS v9.x user side threads may install their own application optimised local allocator (in RHeap)
 - ... This means that for threads that do specific things can employ the best local allocator for their particular task (e.g games, multimedia, signal processing etc)

EKA2 emulator

- The EKA2 emulator does not rely on the host OS for the thread scheduling
 - ... It “freezes” all threads and uses it’s CPU time to schedule (in terms of priorities) exactly as the Nano Kernel scheduler would
 - ... Debugging is so much better this way, since the relevant priorities will always be correct !!
- DLL loading with writeable static data has a one attachment limitation though
 - ... Still better than EKA1 though :-)
- It has been redesigned to be easily portable to other host OSs :-)

More IPCs ?!!

A major event in any OS's evolution is that of Inter Process Communication mechanism addition. Let alone 3 of them !!

- ✓ Publish & Subscribe
- ✓ Message Queues
- ✓ Shared Buffer I/O – between driver and user space

...this evolution has started already and exists in some ways through back-porting in the latest v7.0s and v8.x products

www.symbian.com/developer/techlib/papers/cpp_sysarch.asp#ipc_mech

Beyond Client-Server IPC

- Connection-oriented
- Client initiates request, server responds
- Guaranteed delivery and request completion mechanism
- Connection set-up and teardown is always synchronous in EKA1
- Not anymore in EKA2 - Symbian OS v9.x
- Good paradigm for when many clients need to reliably access a service or shared resource concurrently
- Relation is one-to-one but not peer-to-peer

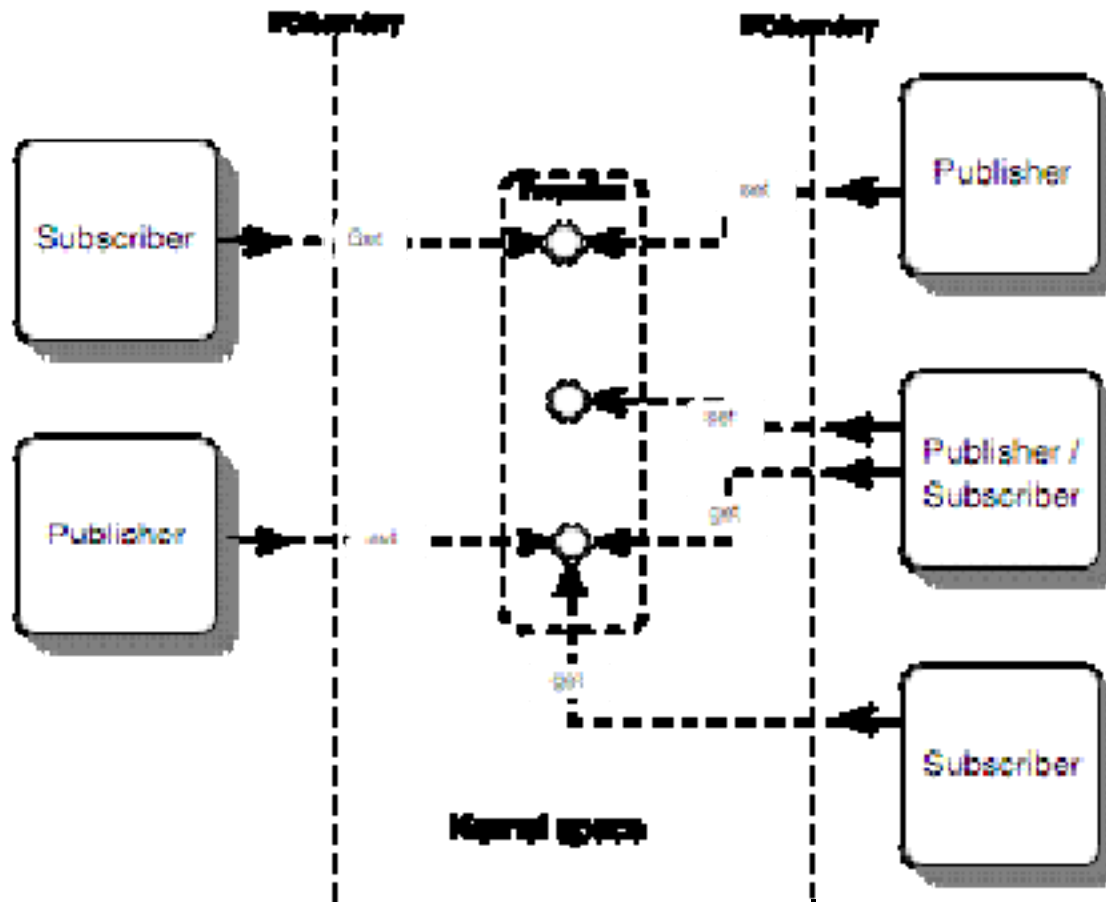
Client-Server IPC Limitations

- Not all I/O is user-initiated – system has evolved ...
- Clients must know which server provides the service they want
- It requires that permanent sessions between clients and server are maintained
- Deadlock potential between servers due to synchronicity of session creation/teardown
- Not really suitable for event multitasking
- Although delivery is guaranteed, there is no real time deterministic guarantee of message delivery.

Publish & Subscribe (a.k.a. Properties)

- Define and publish system-wide properties
- Properties are communicated to *many peers* asynchronously
- Both user and kernel side (via similar APIs)
- Properties are single data values, uniquely identified by an integral key
- Properties have *identity* and *type*
- The identity and type are the only things that need to be shared between publisher(s) and subscriber(s)
- Will be used by the new connectivity framework in v9.x

Publish & Subscribe



Publish & Subscribe operations

- Define: Create a property and define type and identity
- Delete: Remove a property from the system
- Publish: Change the value of a property
- Retrieve: Get the current value of a property
- Subscribe: Register for notification of changes
- Unsubscribe: Deregister for notification of changes

Can be used transiently or by prior 'attachment'

...attachment helps to do real time deterministic operations in
EKA2 – Symbian OS v9.x

Publish & Subscribe characteristics

- Definition and deletion coupled in the same thread
- Either publisher or subscriber may define a property !!
- Connection-less paradigm (between P – S)
- Publishers and subscribers don't need to know about each other or link to special client APIs etc.
- One to many and many to many communication
- Attached or transient operation
- Properties are read and written atomically
- Registration, change, notification, retrieval

Message Queues

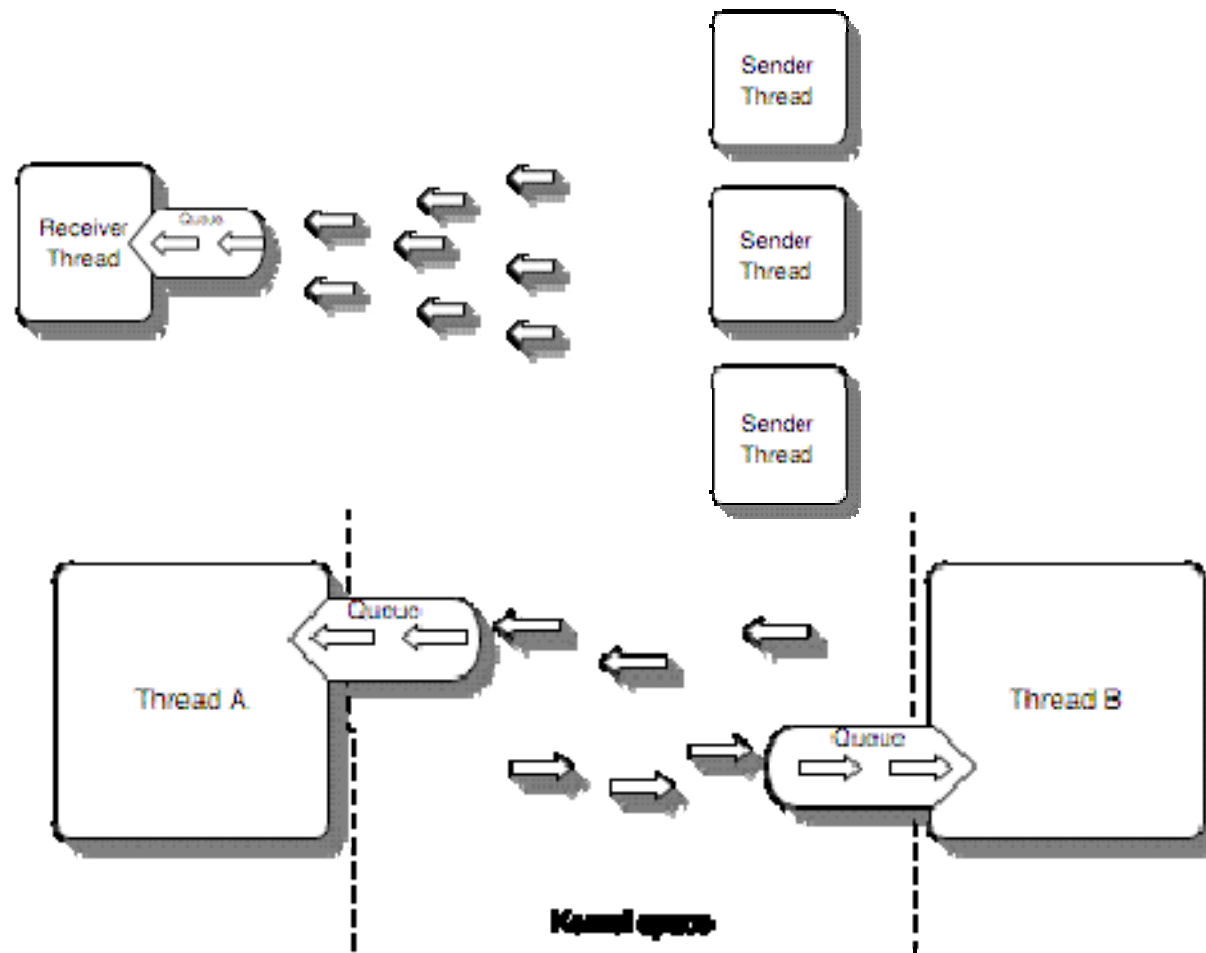
- Peer-to-peer
- Many-to-many
- Fire-and-forget communication semantics
- Guaranteed delivery of messages to queues
- ..but final delivery to reader isn't
- Queues are dimensioned at the point of creation, messages are short and fixed size
- Lowest overhead IPC in EKA2 - Symbian OS v9.x
- deterministic IPC mechanism

Message Queue Operations

There are five basic operations that can be performed on a message queue:

- Creating/opening a message queue
- Sending a message
- Receiving a message
- Waiting for space to become available in the queue
- Waiting for data to arrive in the queue

Message Queues - Data Flow



Message Queues - characteristics

- Kernel managed objects
- Queues may run out of space
- Senders can block on sending
- Senders can be notified of potential overflow thus retry
- Queues can be named and globally visible
- Queues can be process local only
- Queues can be anonymous and globally accessible in EKA2
- Queues may have multiple readers (with health warnings)
- Neither messages nor queues are persistent
- Symbian OS Queues can be *typed*
- *Message structures are <256 bytes*

Getting to the new IPCs now... pre v9.x

You can get to P&S and Message Queues API by using any C++ SDK based on v8.0 or higher

Nokia Series 60 2nd ed FP2

Can't get to EKA2-only features though without a v9.x SDK

They have been back-ported to the very latest 7.0s releases
.....like Series 60 2nd edition FP1 :-)

What is Symbian OS v9.x Platform Security ?

It is a fine grained way to efficiently restrict or completely prevent unauthorised access to sensitive APIs and data on the mobile phone while keeping the device open to developers

- ✓ It follows a per-process capability-based model
- ✓ It compartmentalises the system according to access capabilities to APIs and files
- ✓ It makes sure that the users can make policy decisions they understand
- ✓ It is Kernel mediated but server enforced

Why do this ?

- Why introduce a finer-grained, Platform Security model?
 - ... Phones are open, networked & data communication devices
 - ... Users expect their phones to be highly reliable
 - ... Users care about their privacy – and their phone bills
 - ... Mobile networks are not like the internet – they can restrict access
 - ... “Perimeter Security” model enables unrestricted access to all phone capabilities once installed

Platform Security – user centric view

Plat Sec means for users that:

- They have
 - ... No unexpected items in their phone bill
 - ... Their phone working when needed
 - ... No virus
 - ... Their private data staying private
- They do not have
 - ... To take security decisions they do not understand
 - ... To take security decisions too often

Scope

- Includes
 - ... Symbian OS & device drivers
 - ... User interface
 - ... Applications
- Excludes
 - ... Hardware
 - ... Network infrastructure
 - ... Remote servers

When we talk about Platform Security...

- It is about
 - ... Protecting phone integrity
 - ... Protecting sensitive data
 - ... Controlling access to sensitive operations
- It is not about
 - ... Encrypting data
 - ... Securing network protocols
 - ... Scanning for viruses
 - ... Managing public key infrastructure

Benefits

- For developers
 - ... Maintains network operator & user confidence in open phone environment
 - ... Grows opportunity for mass market applications, content & services
 - ... enables m-commerce applications & high value DRM content
- For network operators
 - ... Protects network & handsets from malware
 - ... Protects customer data & privacy

Don't Panic !



New Symbian OS Concept – Capabilities

- Every executable is tagged at build time with some capabilities, this applies for both EXEs and DLLs
- At run time, every process has a set of capabilities
- Capabilities of a process never change
- Capabilities are assigned based on which APIs a process needs and therefore is authorised to use
- Capabilities and policing of, is transparent to API users

New Symbian OS concept – Data Caging

- Separating code from data (API vs FS)
- File-system structure changes
 - ... \sys, \resource, \private\<process specific>, \<other>
 - ... Executables will be placed and only run from \sys\bin
- Processes are confined to their own part of the file-system
- Access rules based on directory path
 - ... Single user, no access control list required
 - ... No extra storage needed
- Support for removable media file systems
 - ... tamper evidence for binaries

New Symbian OS Concept - Process Identification

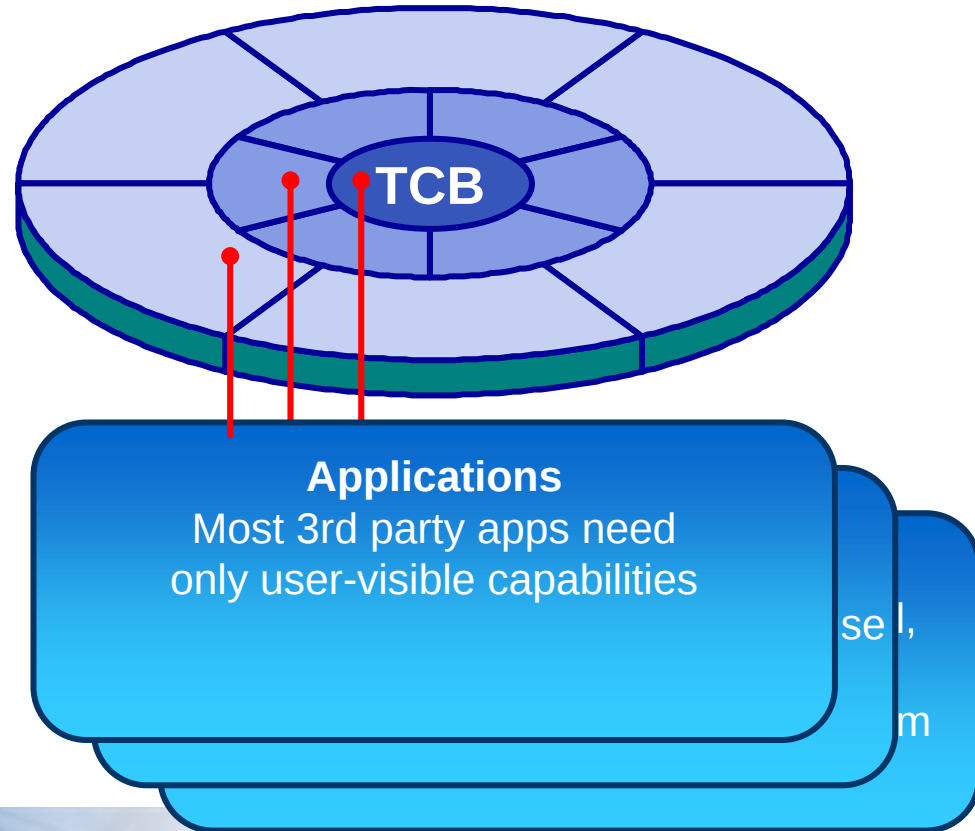
- Each executable now contains a Secure ID (SID)
- Secure IDs are guaranteed to be locally unique
 - ... Hence `\private\<Secure_ID>\`
- SIDs will come from the upper part of the UID range
- SID is specified by the SECUREID keyword in an .mmp file
 - ... If not given UID3 is used, otherwise KNullID
- Each executable now can contain a Vendor ID (VID)
- VIDs allow for unique identification of vendors
- VID is specified by the VENDORID keyword in an .mmp file

New Symbian OS concept - Trusted Computing

- Trusted Computing Base (TCB) → access all areas
 - ... New Kernel, EKA2
 - New Inter-Process communication protocol
 - New kernel memory model
 - ... New Software Install
 - Better rollback of interrupted or failed installation
 - Verification of application's access rights at install-time
 - ... File server & Loader
 - New file access control
 - New loading rules
- Trusted Computing Environment (TCE)
 - ... All important system servers (e.g, ETel, ESock, WServ etc)

Capabilities Model enables Compartmentalisation

- Based on their assigned capabilities, processes may access API calls over IPC or by DLL loading
- System servers will need to police such calls and grant access to callers
- The kernel passes ,like a token, to servers the capabilities of calling processes on each IPC
- The file server will police access to parts of the file-system based on the capabilities and identity of the caller process.



Capabilities categorisation

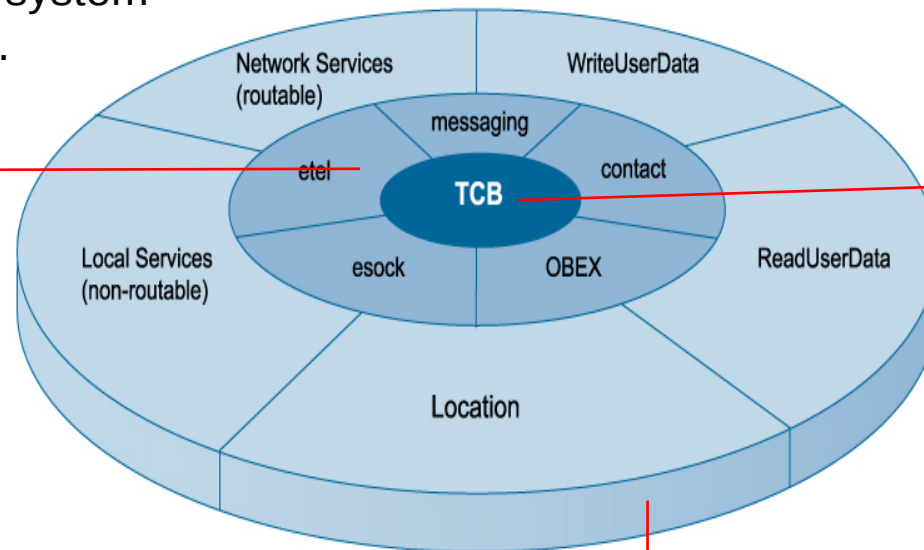
- Full file system privilege
 - ... Reserved for Trusted Computing Base
- System privileges
 - ... Reserved for the Trusted Computing Environment
 - ... Coarse-grained capabilities: CommDD, MultimediaDD, NetworkControl, DRM, DiskAdmin etc
- User privileges
 - ... NetworkServices, LocalServices
 - ... ReadUserData, WriteUserData
 - ... Location, UserEnvironment
- According to capabilities, service access is policed by the next level service providers

TCB → TCE → rest

Capabilities & Trusted Computing Platform

Trusted Computing Environment System servers: Run at different restricted system privileges.

Trusted Computing Base: Runs at full file system - permission to modify executables.



User Visible Range: User can grant these capabilities at install time OR applications can be signed for them.

How to assign capabilities to binaries

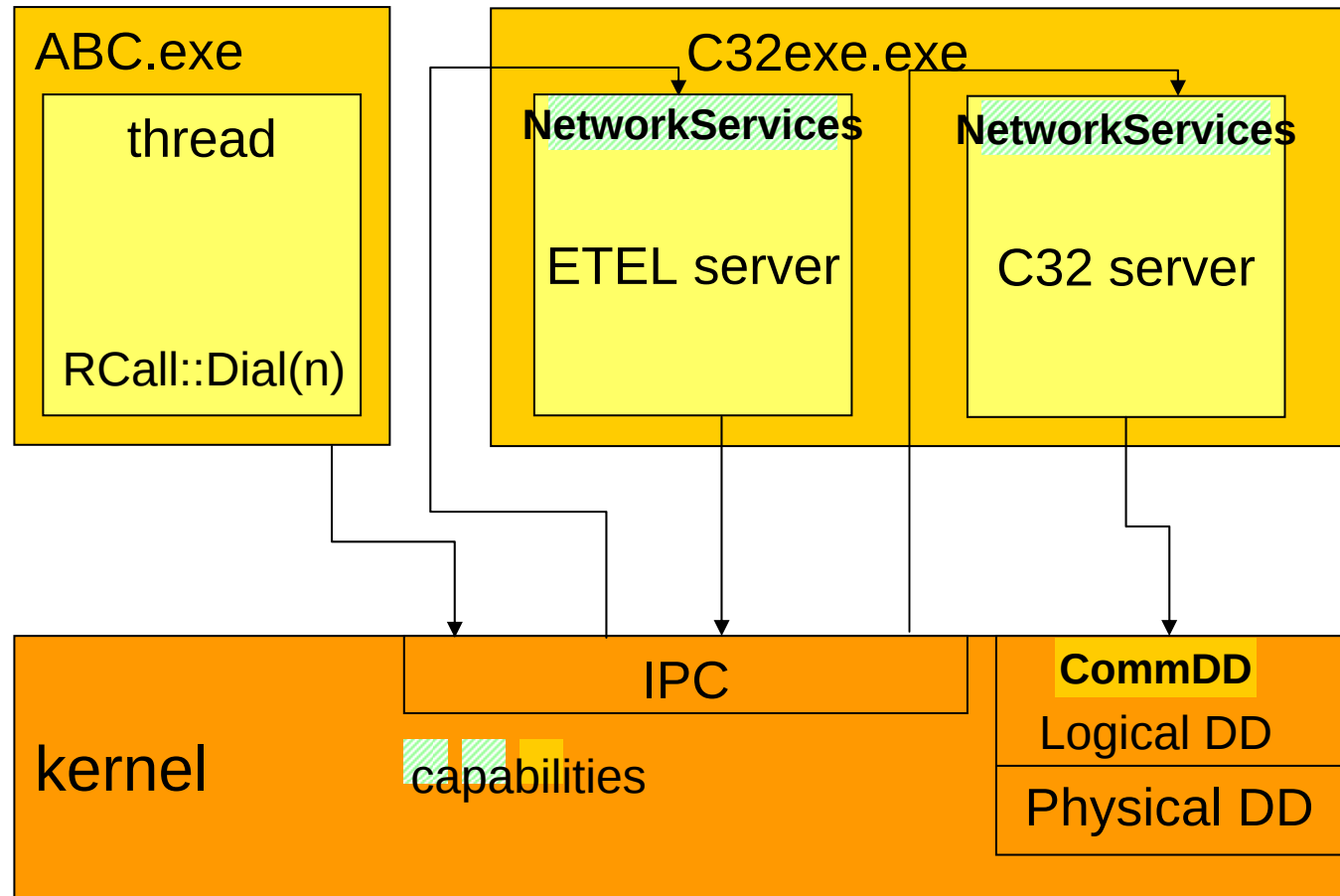
- Capabilities are stored in executables
 - ... They are part of the EKA2 executable file format
- Capabilities are defined in mmp files

```
// program123.mmp
TARGET                program123.exe
TARGETTYPE            exe
UID                   0x00000000 0x00000123
SOURCEPATH            ..\mysource
SOURCE                myfile.cpp
USERINCLUDE            ..\include
SYSTEMINCLUDE         \epoc32\include
....
CAPABILITY          ReadUserData,
                     WriteUserData
```

Capabilities at load time

- Rule 1: The capabilities of a process never change
 - ... No way to add or remove capabilities to a process
 - ... Loading a DLL never change the process capabilities
 - ... DLL code runs at process capabilities level
- Rule 2: A process cannot load a DLL with less capabilities than itself
 - ... DLL capabilities do *only* reflect a level of trust
 - ... DLL capabilities do not authorise anything

How do capabilities work at run-time?



They are worth checking only when a process boundary may be crossed

Data caging directory access rules

- \sys
 - ... Read/Write access reserved to TCB
 - ... All binaries under \sys\bin
 - \resource
 - ... Read access for all, Write access for TCB
 - ... Used for storing fonts, bitmaps, help files...
 - \private\<process_SecureId>\ul> - ... One private space per process
 - ... Process_SecureId == EXE's 3rd UID
 - ... Read/Write access reserved to process owner & TCB
- \<others>
 - ... Read/Write access for all

So what if you want to share ?

- Publish & Subscribe
 - ... New EKA2 IPC allows publisher to specify subscriber capabilities, SIDs or VIDs
- Central repository
 - ... Service for sharing persistent settings
- DBMS
 - ... Service for sharing relational databases
- Shared file handle between processes
 - ... New EKA2 – Symbian OS v9.x feature

What happens to applications then ?

- ABC.app becomes ABC.exe
 - ... To assign ABC.exe the capabilities it needs
 - ... To protect ABC's private data
 - ... Only a few code lines to change
- Application files need to be relocated

\System\Apps\ABC\ABC.app	\Sys\Bin\ABC.exe
\System\Apps\ABC\ABC.mbm	\Resource\Apps\LocalisableFiles\ABC.mbm
\System\Apps\ABC\ABC.rsc	\Resource\Apps\UIResourceFiles\ABC.rsc

What about polymorphic interface DLLs ?

- Plug-in DLLs limited to what the host process can do
 - ... Implementers do not have to implement capability checking
- Plug-in DLLs as trusted as the host process
 - ... Recognisers, same trust level as Apparac server, MTMs same trust level as Messaging server

What about static interface DLLs

- Shared libraries that export a static interface will need to have capabilities such that all its users may load them
- This means that even a simple DLL that does for example some signal processing calculations will need to have capabilities such that a telephony application may use it.
- A DLL that is loaded by another DLL will need to have the same or greater capabilities as the calling executable

..and what about servers ?

- Servers will need to police access to their resources accordingly (use of CPolicyServer)
- Policing must occur at IPC boundaries
- Servers which are trusted by the TCE and others, should be careful not to 'leak' such trust

Bluetooth PAN Profile (already in 8.1)

- BNEP , PAN-GN and PAN-U roles implemented
- No PAN-NAP role though (no NAT in the IP stack)
- Issues surrounding multi-profile operation – the ability to create PAN profile connections, and act in certain PAN profile roles, depends on hardware capabilities and on which other services are currently in use by a device. Developers and users need to be educated on these issues !
- No applications need to change as a result of the PAN profile implementation – PAN IAPs appear just like any other to applications

(Bluetooth) AV Remote Control Framework

- Allows a Symbian OS phone to be controlled by another device, or to act as a remote control for other devices.
- The framework can be extended to support new bearers for transmitting and receiving remote control messages.
- Specified in Bluetooth AVRCP (Audio Video Remote Control Profile) and AV/C Digital Interface Command Set General Specification.

Remote Control Framework

- To route commands to a target (be that remote, or on the phone) the licensee/partner/3rd party (?) must supply a Target Selector (actually depend on connection mode of user). This component accepts unaddressed commands and selects which targets to deliver them to. A reference implementation for this component is provided.
- - Core API providing media type control functions, play, stop etc. This can be used by controllers or targets. An application may act as both a target and controller. When acting as a controller the application may choose to address the commands themselves or delegate that to the Target Selector.
- Symbian also provides Track Info and Absolute Volume APIs, but without the lower-level implementations required to be able to send and receive such messages over Bluetooth
- - AVRCP bearer is provided (in Bluetooth)
- Licensees may add more client side APIs, or bearer plug-ins as they wish.

Bluetooth Stereo Headset Support

- Streaming stereo audio from a phone to a headset
- Remotely controlling a (CD) player from a phone
- GAVDP (Generic Audio Video Distribution Profile)
- Support, for licensees and partners, for writing implementations of A2DP (Advanced Audio Distribution Profile)— Interesting DRM issues for FN support

Bluetooth V2 APIs (since 8.0, but in 8.1 SDL)

- The Bluetooth stack publishes the values of its various settings in the category [KPropertyUidBluetoothCategory](#) and applications can request that settings be changed by setting the value of the equivalent property in the [KPropertyUidBluetoothControlCategory](#).
- It is recommended that where appropriate, application code migrate to using CBluetoothSocket objects in place of RSocketS.
CBluetoothSocket objects should be used where any new code is being written.
- Listening sockets should use auto Bind not SetLocalPort (deprecated)
- SCO connections API, new BTComm port config only etc

Enter RBuf8/16 (since 8.1)

- Introduced in 8.1 and didn't get much publicity due to backwards compatibility issues (?)
- In v9.1 it is time to educate people about them (simpler, safer, better code)

(Almost) Introducing Modern ISO Standard C++...

- TRAP and User::Leave() implemented internally in terms of catch and throw
- Ported 3rd party code can use standard C++ exception mechanisms
 - try/catch/throw
 - ... But not mix these with Symbian OS system APIs
 - ... “Leaving functions must not throw, unless they also catch internally” → “barriers”
- Standard C++ exception specifications are supported
- Writeable static data for DLLs is finally here !!
 - ... emulator has a caveat that allows only one DLL attachment though

..more

- The C++ spec is ISO/IEC 14882 1998/2003 and is enabled but not delivered
- RTTI and `dynamic_cast<>` is enabled, but not for Symbian OS APIs
- RTTI - `dynamic_cast<>` implies the use of `catch()` to handle the exception that is thrown when the cast fails, and this doesn't mesh well with the limited use of C++ exceptions for TRAP and `Leave()`.
- Current coding standards are to use `NONSHARABLE_CLASS` for internal component classes, which disables the RTTI info being emitted

...therefore remember

- Symbian OS is now compiled with exceptions turned on, i.e. exception tables and the like are included in the (x86 and EABI) binaries
- implemented User::Leave/TRAP in terms of exceptions
- It is possible to throw and catch exceptions within your own code.
- It is possible to catch leaves and exceptions in your own code....Don't
- You cannot, in general, TRAP an exception
- Code that leaves and code that throws should not intermingle
- nor should code that leaves have stuff on the stack with nontrivial destructors
- nor should code that throws use the cleanup stack OR depend on code that indirectly depends on objects on it.
- Code that leaves MAY NOT call code that throws, without some suitable barrier -> will not call the cleanup sequence

On Barriers

- Q: Why do we need a 'suitable' barrier?
- A: To protect the integrity of the cleanup stack.

```
#ifdef __LEAVE_EQUALS_THROW__  
EXPORT_C void User::Leave(TInt aReason)  
{  
    TTrapHandler* pH = Exec::TrapHandler();  
    if (pH)  
        pH->Leave(aReason); // causes things on the cleanup stack to be  
        cleaned up  
    throw XLeaveException(aReason);  
}  
#endif
```

A C++ teaser.... be prepared !

- Spot any problems ?

```
void SymbianFunc1L()  
{  
    Cx* obj1 = Cx::NewLC();  
    NonSymbianFuncL(obj1);  
    CleanupStack::PopAndDestroy(obj1);  
}  
void NonSymbianFuncL(Cx* aCx)  
{  
    Sx obj2(aCx); // automatic object with destructor  
    SymbianFunc2L(obj2.DoSomething());  
    obj2.DoSomethingElse();  
    // obj2 destroyed here  
}  
void SymbianFunc2L( aParam )  
    {User::Leave( leavecode );}
```

System Agent finally replaced with P&S

- e.g., `KUidBatteryStrength` in `SaCls.h`

Thank you :-)

Q & A

“Symbian OS Internals: Real Time Kernel Programming”, by Jane Sales

“Symbian OS Explained”, by Jo Stichbury

“Symbian OS for Mobile Phones vol2” , by R. Harrison

..and of course visit the Symbian Developer Network