Commercial Mobile Operating Systems

Unit 5

Classification based on ownership

- Manufacturer-built proprietary operating systems (both device and OS by the same organization)
  - Apple iOS
  - RIM BlackBerry OS
  - HP WebOS
- Third party proprietary operating systems (organizations license their OS to device manufactures)
  - Microsoft Windows Phone 7
  - Microsoft Windows Mobile
- Free & open source operating systems (by a community of developers)
  - Android
  - MeeGo
  - Symbian

Examples of mobile operating systems

1. Windows Mobile
2. Palm OS (deprecated)
3. Symbian OS
4. iOS
5. Android
6. Blackberry Operating system

Windows Mobile

- **Windows Mobile is a compact operating system designed for mobile devices and based on Microsoft Win32.**
- **Windows CE (Compact Edition) - designed specifically for handheld devices, based on Win32 API.**
- **PDA (personal digital assistant), palmtop computer, PocketPC were original intended platform for the Windows Mobile OS.**
- **The OS provides multitasking and has the ability to navigate a file system.**
- **Internet Explorer Mobile is the default web browser, and Windows Media Player is the default media player used for playing digital media. The mobile version of Microsoft Office, is the default office suite.**
- **Most early Windows Mobile devices came with a stylus.**
- **There are three main versions of Windows Mobile for various hardware devices:**
  1. **Windows Mobile Professional** (runs on smartphones with touchscreens)
  2. **Windows Mobile Standard** (runs on mobile phones without touchscreens)
  3. **Windows Mobile Classic** (which runs on personal digital assistant or Pocket PCs.)

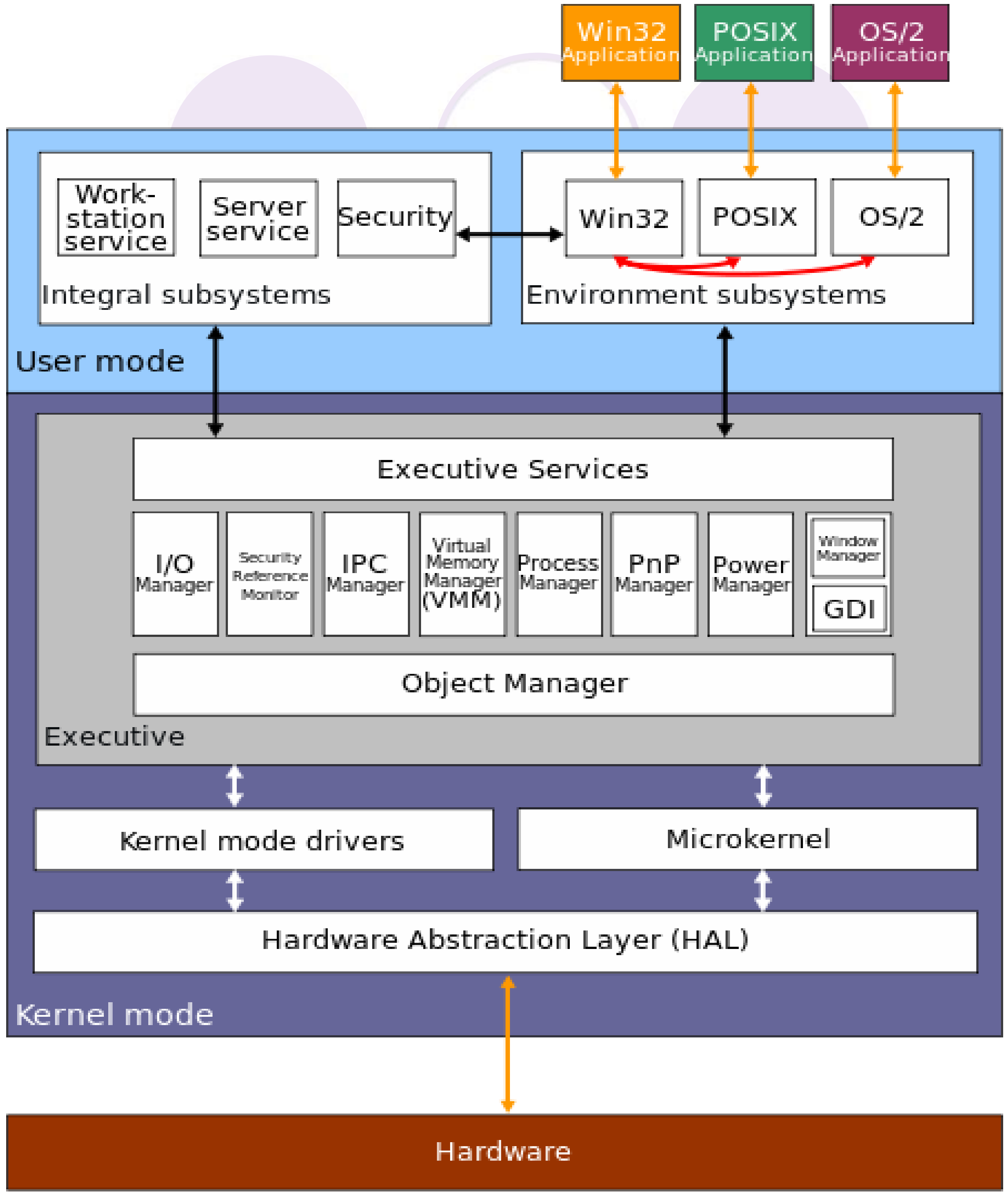
History of Windows Mobile OS

- 1996 – Windows CE 1.0
- 1997 – Windows CE 2.0 (ATM, games consoles, Handheld PC's, kitchen utensils)
- 2000 - Windows CE 3.0 - Pocket PC 2000 - (became the os of choice on many Pocket PCs, looked and worked like Windows 98, no phone feature)
- 2001 - CE 3.0 - Smartphone
- 2002– used for Pocket PC phones and Smartphones, UI reflect the new Windows XP
- 2003 – Windows Mobile 2003 (Windows CE 4.2) - first release under the Windows Mobile banner - name changed form PocketPC to Windows Mobile
- 2005 - WM5 (CE5.0) - new standard API created for a simplified programming of 3D apps and games with Direct3Dmobile. It use .Net Compact Framework environment

History of Windows Mobile OS

- 2007 – WM6 (CE 5.2) – (also year of introducing iPhone) similar in design to the Vista, works much like WM5, but with much better stability
- 2008 – WM 6.1 – (year of releasing Android)
- 2009 – WM6.5, vertically scrollable labels, Windows Marketplace announced
- Feb 2010 – WM6.5.3, was officially announced as first Windows Phone 6.5.3 smartphone

Windows phone 8 architecture



Palm OS

- Palm OS is an embedded operating system designed for ease of use with a touch screen-based graphical user interface.
- It has been implemented on a wide variety of mobile devices such as smart phones, barcode readers, and GPS devices.
- It is run on Arm architecture-based processors. It is designed as a 32-bit architecture.
- ARM – is a family of reduced instruction set computing (RISC) architectures for computer processors. A RISC-based computer design approach means processors require fewer transistors than typical complex instruction set computing (CISC) processors in most personal computers. This approach reduces costs, heat and power use.

PalmOS

- Palm OS does not use a traditional flat file system. Data is stored in memory chunks called “records”, which are grouped into “databases”. A database is analogous to a file. The difference is that data is broken down into multiple records instead of being stored in one contiguous chunk.

- Palm OS applications are generally single-threaded, event-driven programs. Only one program runs at a time.

- Each application has a PilotMain function that is equivalent to main in C programs. To launch an application, Palm OS calls PilotMain and sends it a launch code. The launch code may specify that the application is to become active and display its user interface (called a normal launch), or it may specify that the application should simply perform a small task and exit without displaying its user interface. The sole purpose of the PilotMain function is to receive launch codes and respond to them.

- Applications can send launch codes to each other, so an application might be launched from another application or it might be launched from the system. An application can use a launch code to request that another application perform an action or modify its data.

Palm OS

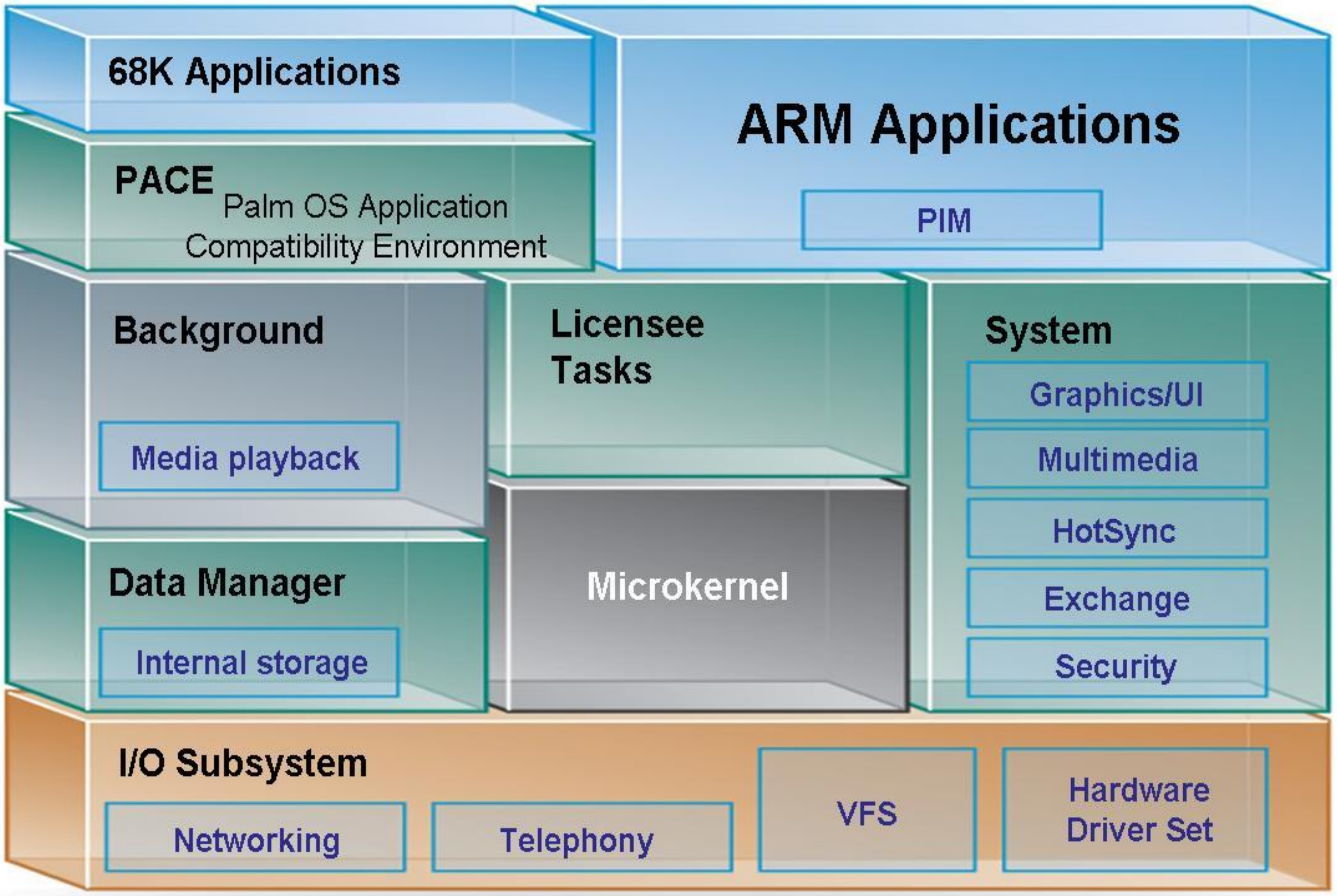
- The key features of Palm OS
- A single-tasking OS:

1. Palm OS Garnet (5.x) uses a kernel developed at Palm, but it does not expose tasks or threads to user applications. In fact, it is built with a set of threads that can not be changed at runtime.
2. Palm OS Cobalt (6.0 or higher) does support multiple threads but does not support creating additional processes by user applications.

Palm OS

- Palm OS has a pre-emptive multitasking kernel that provides basic tasks but it does not expose this feature to user applications.
- Memory Management: The Memory, RAM and ROM, for each Palm resides on a memory module known as card. In other words, each memory card contains RAM, ROM or both. Palms can have no card, one card or multiple cards. Handwriting recognition input.
- Support of serial port, USB, Infrared, Bluetooth and Wi-Fi connections Defined standard data format for PIM (Personal Information Management) applications to store calendar, address, task and note entries, accessible by third-party applications

PalmOS 6 architecture



Symbian OS

- Symbian OS is 32 bit, little-endian operating system, running on different flavours of ARM architecture
- It is a multitasking operating system and very less dependence on peripherals.
- Kernel runs in the privileged mode and exports its service to user applications via user libraries.
- User libraries include networking, communication, I/O interfaces and etc.
- Access to these services and resources is coordinated through a client-server framework.
- Clients use the service APIs exposed by the server to communicate with the server. The client-server communication is conducted by the kernel.

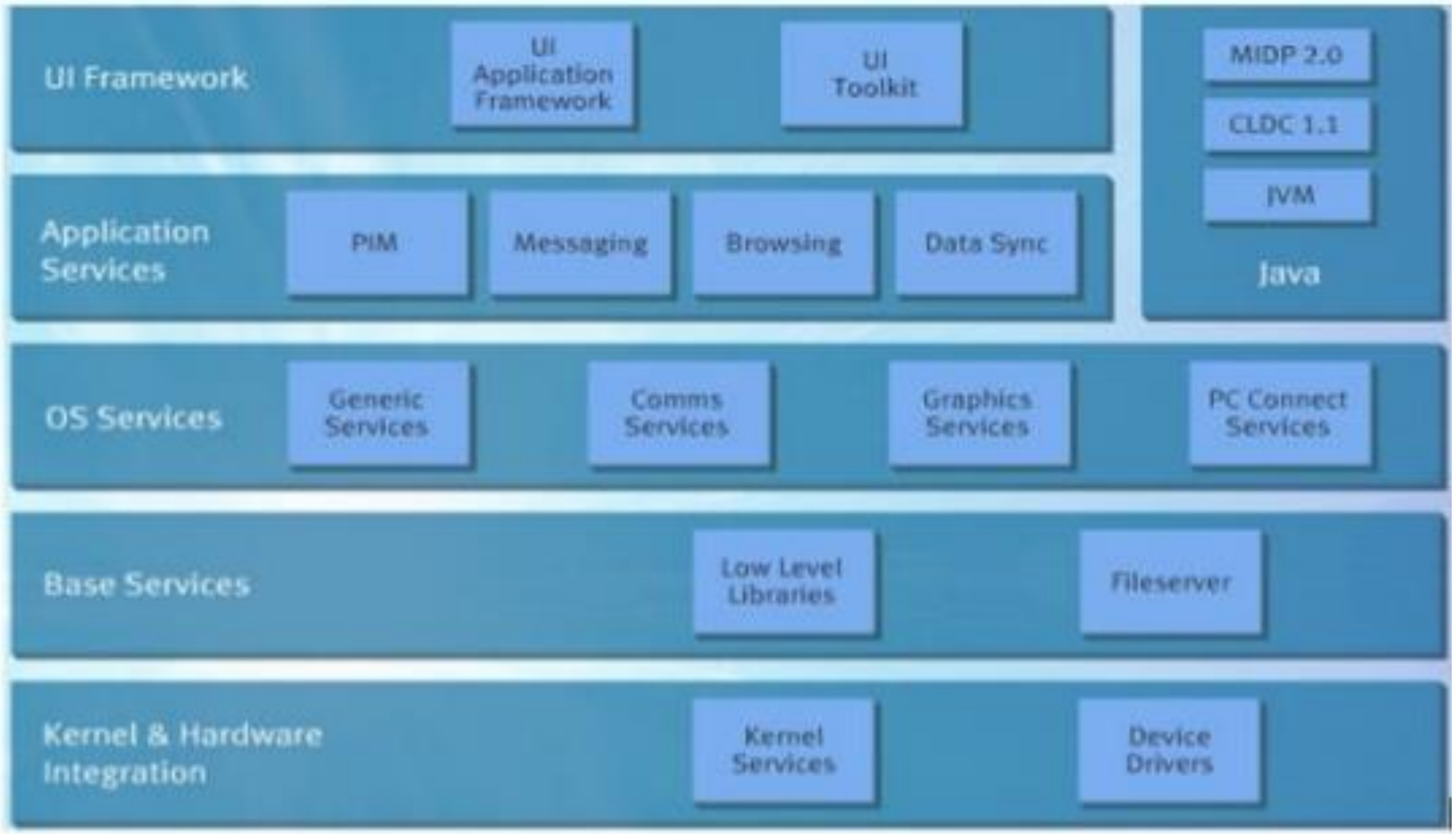
Symbian OS

- Real-time: it has a real-time, multithreaded kernel.
- **Data Caging :**
  A. it allows applications to have their own private data partition. This feature allows for applications to guarantee a secure data store.
  B. It can be used for e-commerce applications, location aware applications and etc.
- **Platform Security :** Symbian provides a security mechanism against malware.
  A. It allows sensitive operations can be accessed by applications which have been certified by a signing authority.
  B. In addition, it supports full encryption and certificate management, secure protocols (HTTPS, TLS and SSL) and WIM framework.

Symbian OS

- Multimedia : it supports audio, video recording, playback and streaming, and Image conversion.
- Internationalization support : it supports Unicode standard. Fully object-oriented and component- based. Optimized memory management
- Client-server architecture . it provides simple and high-efficient inter process communication. This feature also eases porting of code written for other platforms to Symbian OS.
- A Hardware Abstraction Layer (HAL)
  1. **This layer provides a consistent interface to hardware and supports device-independency**
  2. **Kernel offers hard real-time guarantees to kernel and user mode threads**

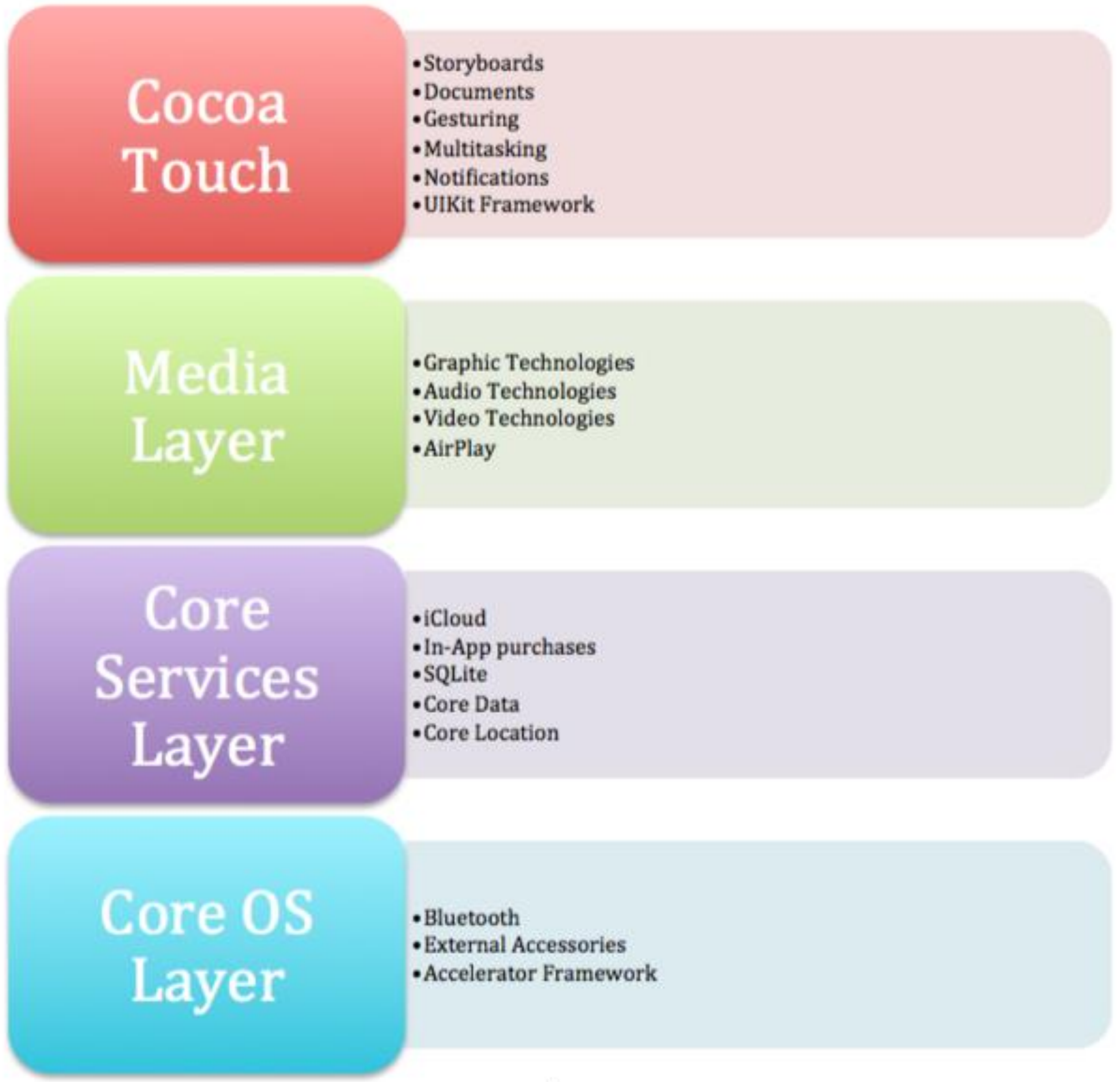
Layered Architecture Of Symbian OS



iOS

- Cocoa Touch is a UI framework for building software programs to run on the iOS operating system (for the iPhone, iPod Touch, and iPad) from Apple Inc.
- Cocoa Touch mainly contains the classes implemented in Objective-C. Because Objective-C is a superset of C, it is easy to mix C and even C++ into your Cocoa Touch applications.

iOS architecture



Core OS Layer Frameworks
1. Accelerate Framework 2. Core Bluetooth Framework 3. External Accessory Framework 4. Generic Security Services Framework 5. Security Framework 6. System 7. 64-Bit Support

Core OS Layer

- The Core OS layer holds the low level features that most other technologies are built upon.
- Core Bluetooth Framework.
- Accelerate Framework.
- External Accessory Framework.
- Security Services framework.
- Local Authentication framework.
- 64-Bit support from IOS7 supports the 64 bit app development and enables the application to run faster.

Cocoa touch layer

- EventKit framework – gives view controllers for showing the standard system interfaces for seeing and altering calendar related events
- GameKit Framework – implements support for Game Center which allows users share their game related information online
- iAd Framework – allows you deliver banner-based advertisements from your app.
- MapKit Framework – gives a scrollable map that you can include into your user interface of app.
- PushKitFramework – provides registration support for VoIP apps.
- Twitter Framework – supports a UI for generating tweets and support for creating URLs to access the Twitter service.
- UIKit Framework – gives vital infrastructure for applying graphical, event-driven apps in iOS.

Core Services Layer

- **Address book framework** – Gives programmatic access to a contacts database of user.
- **Cloud Kit framework** – Gives a medium for moving data between your app and iCloud.
- **Core data Framework –** Technology for managing the data model of a Model View Controller app.
- **Core Foundation framework –** Interfaces that gives fundamental data management and service features for iOS apps.
- **Core Location framework –** Gives location and heading information to apps.
- **Core Motion Framework –** Access all motion based data available on a device. Using this core motion framework Accelerometer based information can be accessed.
- **Foundation Framework –** Objective C covering too many of the features found in the Core Foundation framework
- **Healthkit framework –** New framework for handling health-related information of user
- **Homekit framework –** New framework for talking with and controlling connected devices in a user's home.
- **Social framework –** Simple interface for accessing the user's social media accounts.
- **StoreKit framework –** Gives support for the buying of content and services from inside your iOS apps, a feature known asIn-App Purchase.

Key Technologies in the cocoa layer are:	**Cocoa Touch Frameworks:**
1.Air Drop. 2.TextKit. 3.UIKit Dynamics . 4.Multitasking . 5.Auto Layout. 6.Story Boards. 7. UI State Preservation. 8.Apple Push Notification. 9.Local Notifications. 10.Gesture Recognizers. 11.Standard System View Controllers.	1.Address Book UI Framework 2.Event Kit UI Framework 3.Game Kit Framework 4.MapKit Framework 5.iAd Kit Frame Work 6.Message UI Framework 7.Twitter Frame Work

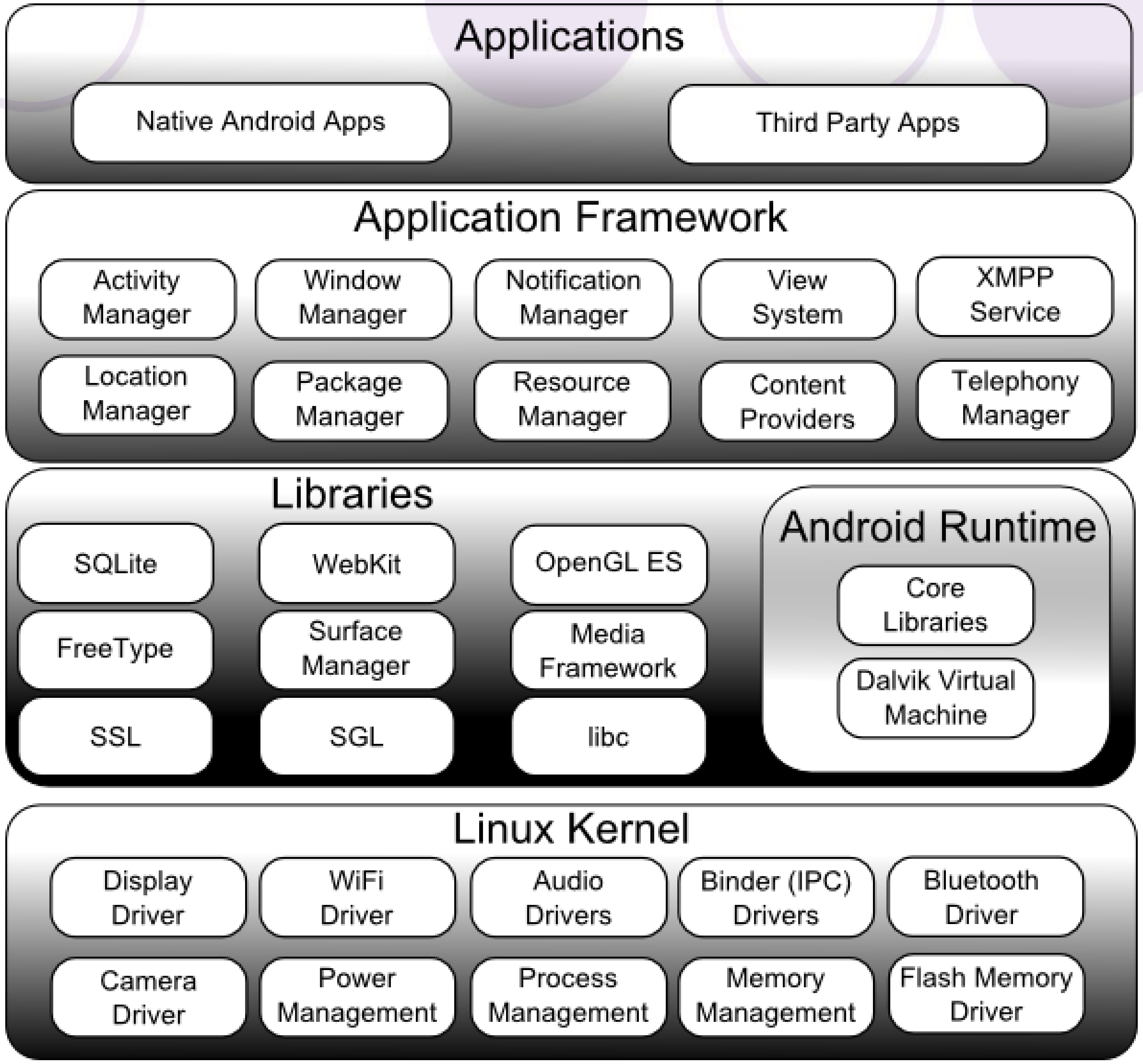
Graphics Technologies :	**Audio Technologies:**
1.UIKit Graphics 2.Core Animation 3.OpenGL ES (2D/3D) 4.TextKit and Core Text 5.Image I/O(to read/ write any image format)	1.Media Player 2.AV-Foundation 3.OpenAL 4.Core Audio
Video Technologies:	**Air Play:**
1.UIImagePickerController 2.Media Player 3.AV-Foundation 4.Core Media	This Technology lets your app stream audio &amp; video content to Apple TV and stream audio content to third party airplay speakers and recievers.

Features in Core Services Layer	Core Services Frameworks
1. Peer-to-Peer Services 2. iCloud Storage 3. Automatic Reference Counting 4. Block Objects 5. Data Protection 6. File-Sharing Support 7. Grand Central Dispatch 8. In-App Purchase 9. SQLite 10.XML Support	1. Accounts Framework 2. Address Book Framework 3. Ad Support Framework 4. CFNetwork Framework 5. Core Data Framework 6. Core Foundation Framework 7. Core Location Framework 8. Core Media Framework 9. Core Motion Framework 10.Core Telephony Framework 11.Event Kit Framework 12.Foundation Framework 13.JavaScript Core Framework 14.Mobile Core Services Framework 15.Multipeer Connectivity Framework 16.Newsstand Kit Framework 17.Pass Kit Framework 18.Quick Look Framework 19.Safari Services Framework 20.Social Framework 21.Store Kit Framework 22.System Configuration Framework

Android OS

- Linux kernel
  - At the bottom of the layers is Linux - Linux 3.6 with approximately 115 patches. This provides a level of abstraction between the device hardware and it contains all the essential hardware drivers like camera, keypad, display etc. Also, the kernel handles all the things that Linux is really good at such as networking and a vast array of device drivers, which take the pain out of interfacing to peripheral hardware.
- Libraries
  - On top of Linux kernel there is a set of libraries including open-source Web browser engine WebKit, well known library libc, SQLite database which is a useful repository for storage and sharing of application data, libraries to play and record audio and video, SSL libraries responsible for Internet security etc.

Android OS Architecture



- **Android Libraries**
  1. This category encompasses those Java-based libraries that are specific to Android development. Examples of libraries in this category include the application framework libraries in addition to those that facilitate user interface building, graphics drawing and database access. A summary of some key core Android libraries available to the Android developer is as follows –
  2. **android.app** – Provides access to the application model and is the cornerstone of all Android applications.
  3. **android.content** – Facilitates content access, publishing and messaging between applications and application components.
  4. **android.database** – Used to access data published by content providers and includes SQLite database management classes.
  5. **android.opengl** – A Java interface to the OpenGL ES 3D graphics rendering API.
  6. **android.os** – Provides applications with access to standard operating system services including messages, system services and inter-process communication.
  7. **android.text** – Used to render and manipulate text on a device display.
  8. **android.view** – The fundamental building blocks of application user interfaces.
  9. **android.widget** – A rich collection of pre-built user interface components such as buttons, labels, list views, layout managers, radio buttons etc.
  10. **android.webkit** – A set of classes intended to allow web-browsing capabilities to be built into applications.

- Also has C/C++ based libraries contained in the Android software stack.

Android OS

- This is the third section of the architecture and available on the second layer from the bottom. This section provides a key component called **Dalvik Virtual Machine** which is a kind of Java Virtual Machine specially designed and optimized for Android..
- The Android runtime also provides a set of core libraries which enable Android application developers to write Android applications using standard Java programming language.
- **Application Framework**

1. The Application Framework layer provides many higher-level services to applications in the form of Java classes. Application developers are allowed to make use of these services in their applications.
2. **The Android framework includes the following key services –**
3. **Activity Manager** − Controls all aspects of the application lifecycle and activity stack.
4. **Content Providers** − Allows applications to publish and share data with other applications.
5. **Resource Manager** − Provides access to non-code embedded resources such as strings, colour settings and user interface layouts.
6. **Notifications Manager** − Allows applications to display alerts and notifications to the user.
7. **View System** − An extensible set of views used to create application user interfaces.

4 main Components in Android Application

S.No	Components & Description
1	**Activities** They dictate the UI and handle the user interaction to the smart phone screen.
2	**Services** They handle background processing associated with an application.
3	**Broadcast Receivers** They handle communication between Android OS and applications.
4	**Content Providers** They handle data and database management issues.

REFERENCES

1. Evolution of mobile OS - https://www.tiki-toki.com/timeline/entry/356039/From-Symbian-to-Sailfish-the-Evolution-of-the-Smartphone-OS/
2. Windows phone 8 - https://andri102.wordpress.com/android-os/windows-phone-8/
3. Symbian OS - https://es.slideshare.net/PriyaPandharbale/symbian-os-presentation
4. PalmOS6 - http://mobile.osnews.com/printer.php?news_id=6148
5. For overall content - https://www.slideshare.net/ayyakathir/it6601-mobile-computing-54722943
6. iOS - https://tilakgondi.wordpress.com/2015/01/14/ios-architecture/
7. Android - https://www.tutorialspoint.com/android/android_overview.htm
8. Blackberry - https://de.slideshare.net/appinonline1/secured-mobile-application-development-in-android-blackberry-ios