Systemy operacyjne na urządzenia mobilne

Marek Białowąs, Wojciech Łobacz, Sławek Rudnicki, Marcin Sulikowski

Uniwersytet Warszawski 2009

Agenda

- ✔ Urządzenia mobilne - wprowadzenie
- ✔ Problematyka urządzeń mobilnych
- ✔ Symbian
- ✔ Windows Mobile
- ✔ Android
- ✔ Android LiveDemo
- ✔ Systemy na inne urządzenia mobilne
- ✔ Podsumowanie

Urządzenia mobilne

Nieformalnie:

„Możesz to oprogramować, wsadzić w kieszeń i zabrać ze sobą”

Formalniej:

„Urządzenie elektroniczne pozwalające na przetwarzanie, odbieranie oraz wysyłanie danych bez konieczności utrzymywania przewodowego połączenia z siecią. Urządzenie mobilne może być przenoszone przez użytkownika bez konieczności angażowania dodatkowych środków”

M. Macutkiewicz

Urządzenia mobilne przykłady

Tablet PC

Samsung Q1, Toshiba Portege, Fujitsu Lifebook, Motion Computing, IBM Thinkpad

PDA (Personal Digital Assistant)

Palm Pilot, Revo, Sony Clie, Hewlett-Packard Jornado, Casio Cassiopedia, Compaq iPaq, Toshiba Pocket PC

Smartphone

Sony Ericsson, Palm Treo, Blackberry, Nokia T-Mobile Sidekick, Torq, Motorola Q, E-Ten, HP iPaq, I-mate

Inne

wysokiej klasy aparaty fotograficzne
wysokiej klasy odtwarzacze multimedialne

Przykładowo - HTC Touch Pro

Procesor 528 MHz
QualcommMSM7501A

Wymiary 1016mm x 508mm x 168

Waga: 147gram

Rozdzielczość ekranu 480x640

Klawiatura QWERTY

Ekran dotykowy

Aparat: Tak

Pamięć: 288 MB RAM; 512 MB ROM

Dodatkowa pamięć: Micro SD

Wymagania, trudności

SDK

Modułowość

Stabilne jądro

Bezpieczeństwo

Energooszczędny

Szybkość połączenia internetowego

Wymagania, trudności

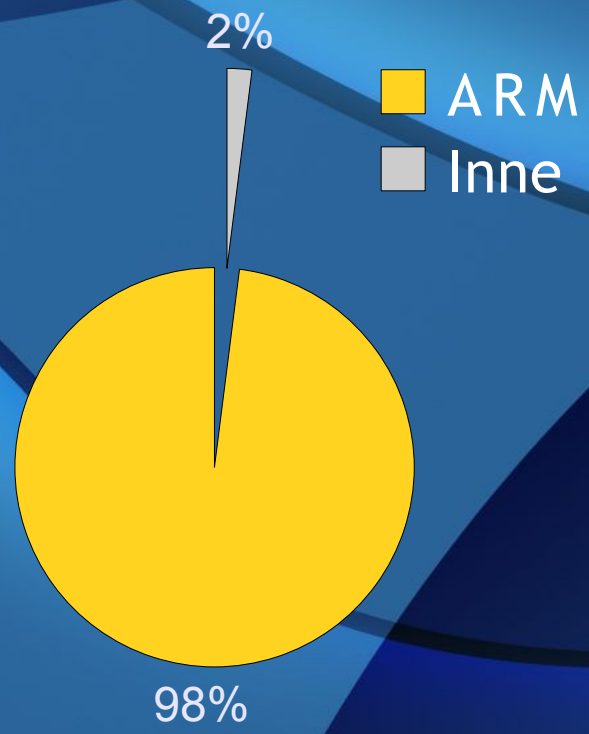
- Interfejs
- Mała pamięć
- Oszczędność energii
- Odporność na zakłócenia
- Nieszkodliwy zdrowotnie
- Nietechniczni użytkownicy

Problemy bezpieczeństwa

- Uwierzytelnianie
- Model STRIDE:
  - **S** poofing,
  - **T** ampering,
  - **R** epudiation,
  - **I** nformation Disclose,
  - **D** enial of Service,
  - **E** levation of Privilege

Architektury mobilne

- Advanced RISC Machine (ARM)
- Intel Atom
- ...



ARM – historia

- Projekt Acorn Computers Ltd.
- 1986 rok – ARM2
  - 30 000 tranzystorów
    - Motorola 68000 (1979) – 70 000
    - Pentium IV Prescott (2005) – 169 000 000
- 1994 – StrongARM
  - 233 MHz przy poborze mocy 1W.

ARM – zastosowania

- ARM7TDMI
  - iPod
  - GameBoy Advance
  - Lego Mindstorms
- ARM9E
  - Nintendo Wii
  - telefony SE, BenQ
- ARM11
  - smartfony Nokii
  - iPhone
  - iPod Touch
  - procesory Qualcomm

ARM – istotne własności

- RISC
- Instrukcje 32-bitowe wykonywane warunkowo

- Tryb Thumb – wykonywanie instrukcji skompresowanych do 16 bitów
  - wolniejsze, ale program zajmuje mniej pamięci

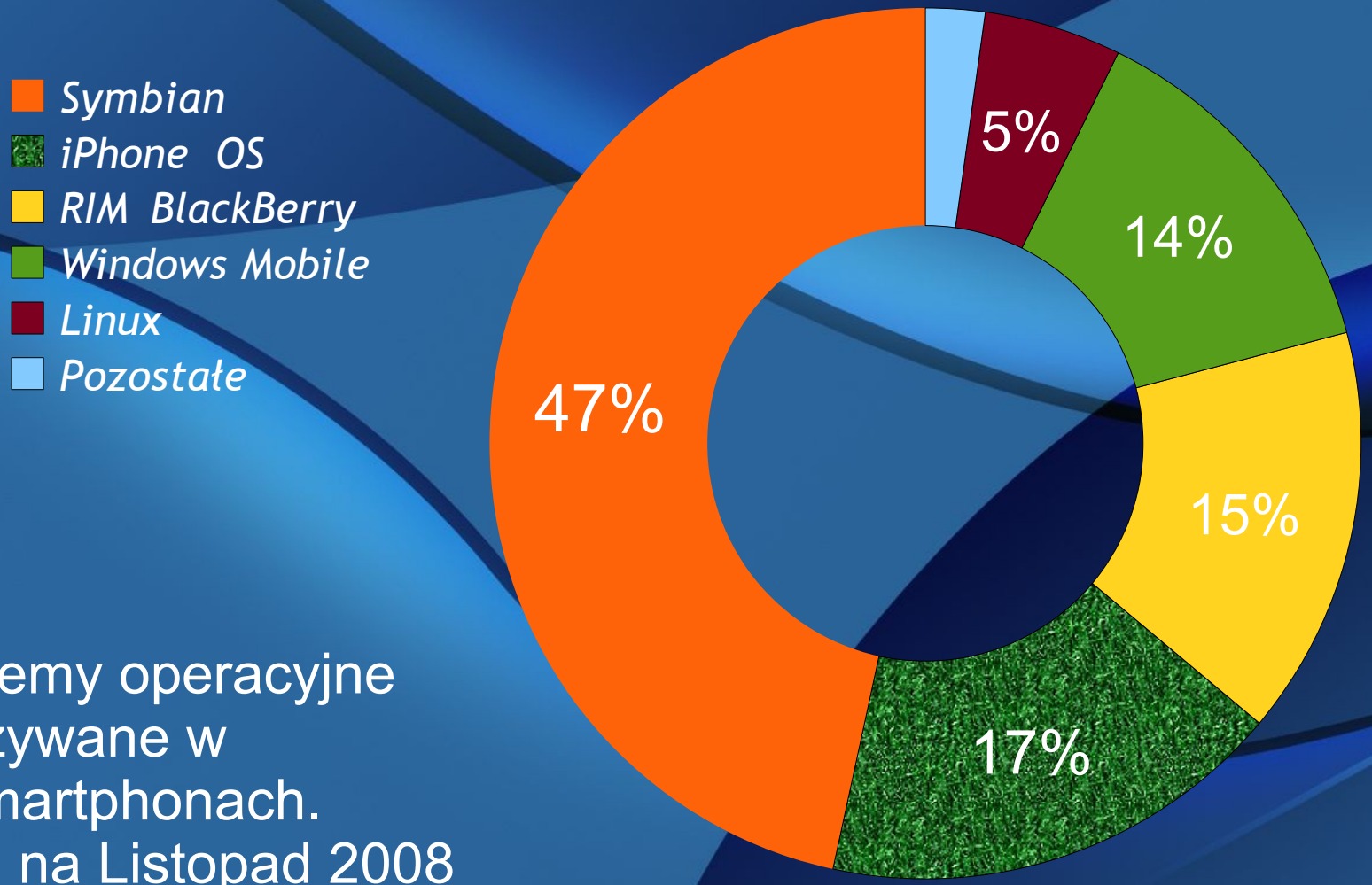
ARM – istotne własności

- Jazelle
  - uruchamianie bytekodu Javy bezpośrednio na procesorze
  - obsługa do 95% instrukcji przez sprzęt
  - tylko wybrane JVM
  - ważne ze względu na popularność aplikacji i gier Java wśród użytkowników urządzeń mobilnych

Wiele systemów - dobrze?

Nadmiar systemów operacyjnych dla urządzeń przenośnych ogranicza rozwój rynku. Według Aruna Sarin, prezesa Vodafone, platform mobilnych powinno być najwyżej pięć. Rozwój wielu - zależnych od platform mobilnych - rynków jest hamowany przez konieczność przenoszenia tego samego kodu na różne systemy operacyjne

Smartphone



Systemy operacyjne używane w smartphonach.
Stan na Listopad 2008

Kod aplikacji

	Maszynowy	Zarzadzany
Symbian	X	X
Windows Mobile	X	X
Android		X
iPhone	X	
BlackBerry		X

Maszynowy

zależny od architektury, nieczytelny, szybki

Zarządzany (np. JVM)

przenośność, wygoda, bezpieczeństwo

symbian OS

Założenia
Historia
Struktura
Tworzenie oprogramowania

Systemy Operacyjne 2009 – Urządzenia mobilne
Marek Białowąs, Wojciech Łobacz, Sławomir Rudnicki, Marcin Sulikowski

Symbian w założeniach

Z rodziny EPOC
'Electrionic Piece of Cheese' czy 'Epoch'
EPOC32
- GUI EIKON
- dedykowany na ARM
- inspirowany VMS-em
- pojedynczy użytkownik
- zabezpieczona pamięć
- wielozadaniowy z wywłaszczeniem

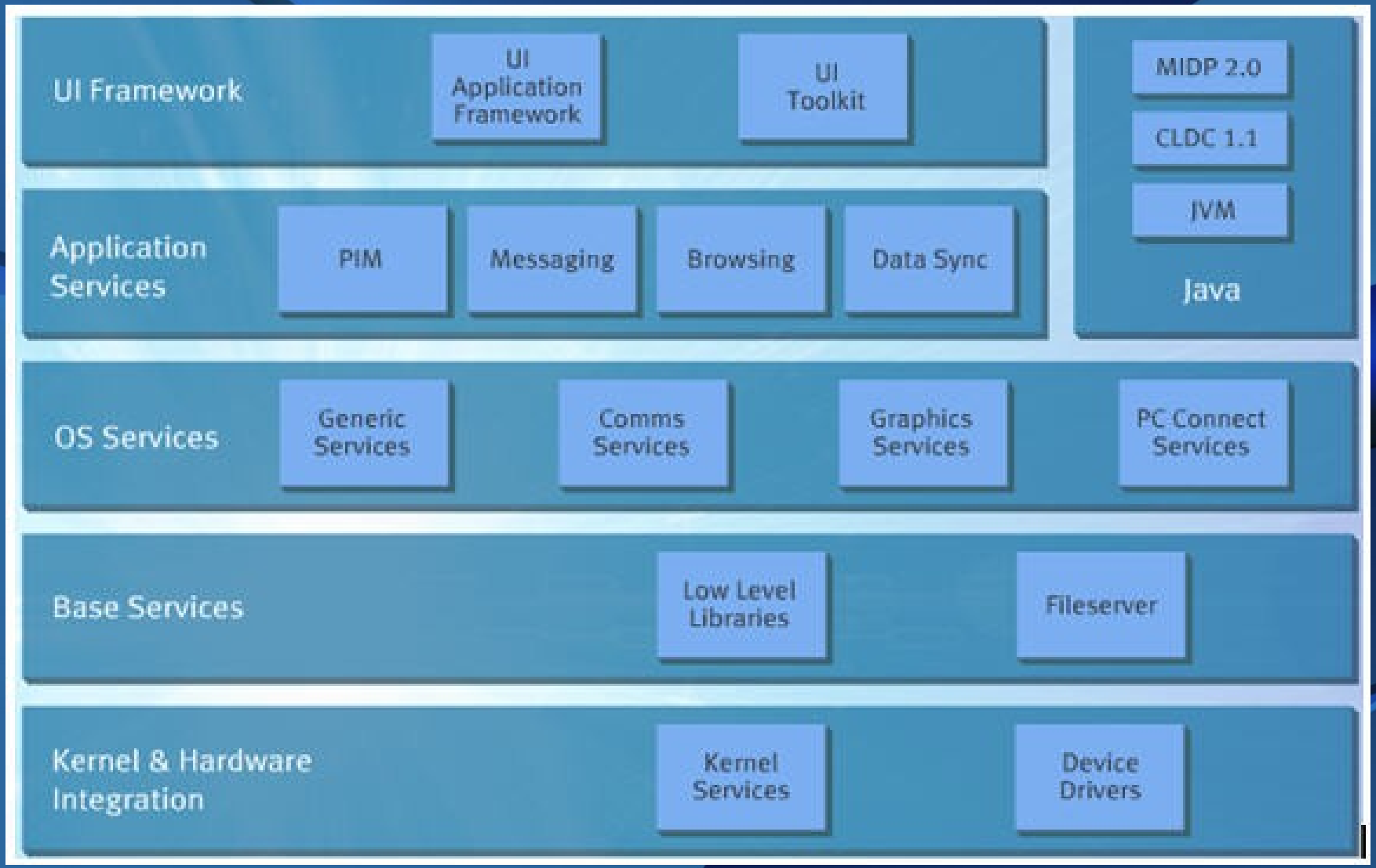
III Zasady Twórców Symbiana

1) Spójność i bezpieczeństwo danych użytkownika
2) Szybkość działania
3) Wszystkie zasoby są ważne

Następstwa:

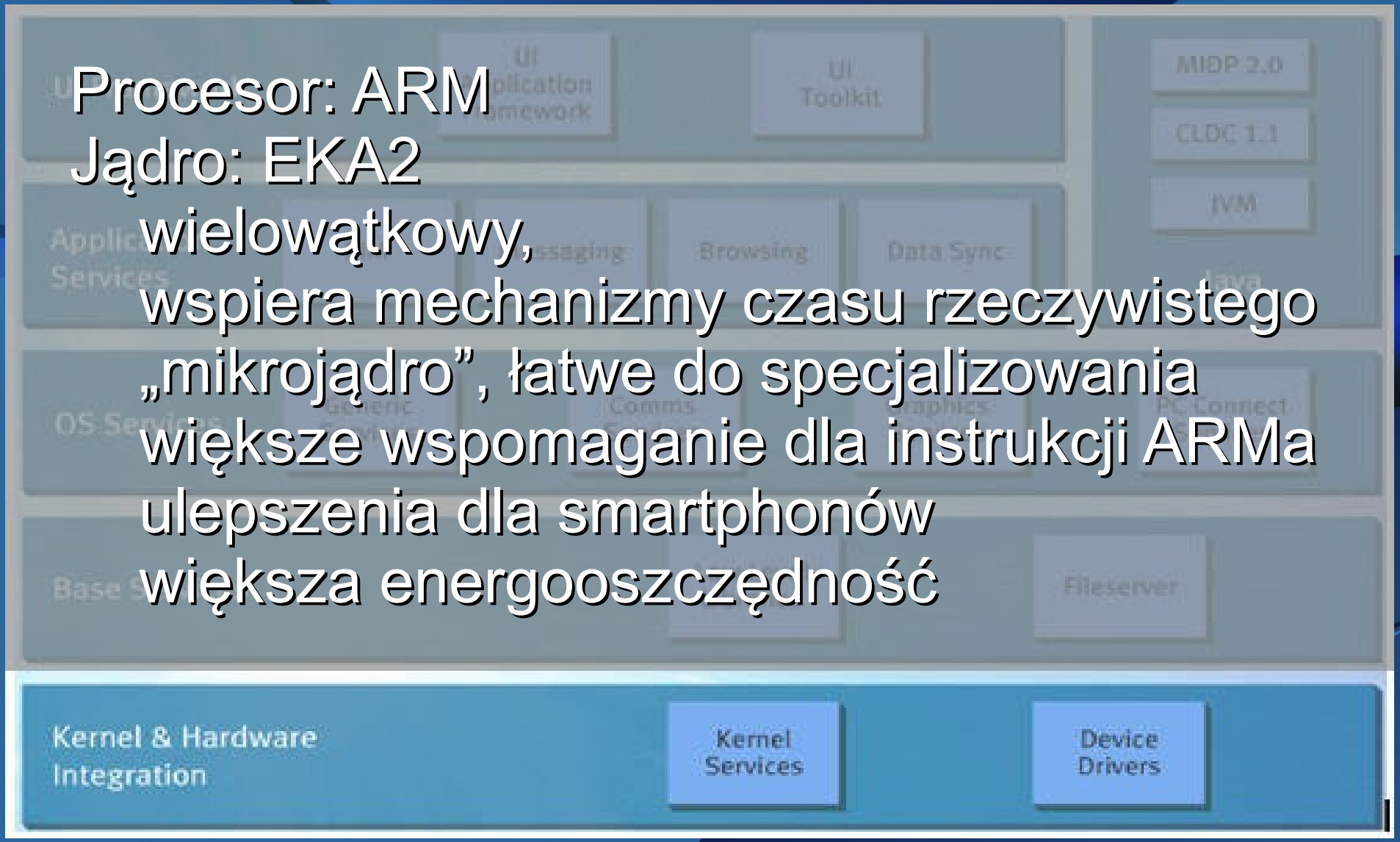
- Mikrojądro
- Rozszerzalność
- Aplikacje MVC
- Wyraźne oddzielenie interfejsu
- Podejście żądanie-odpowiedź.

Architektura

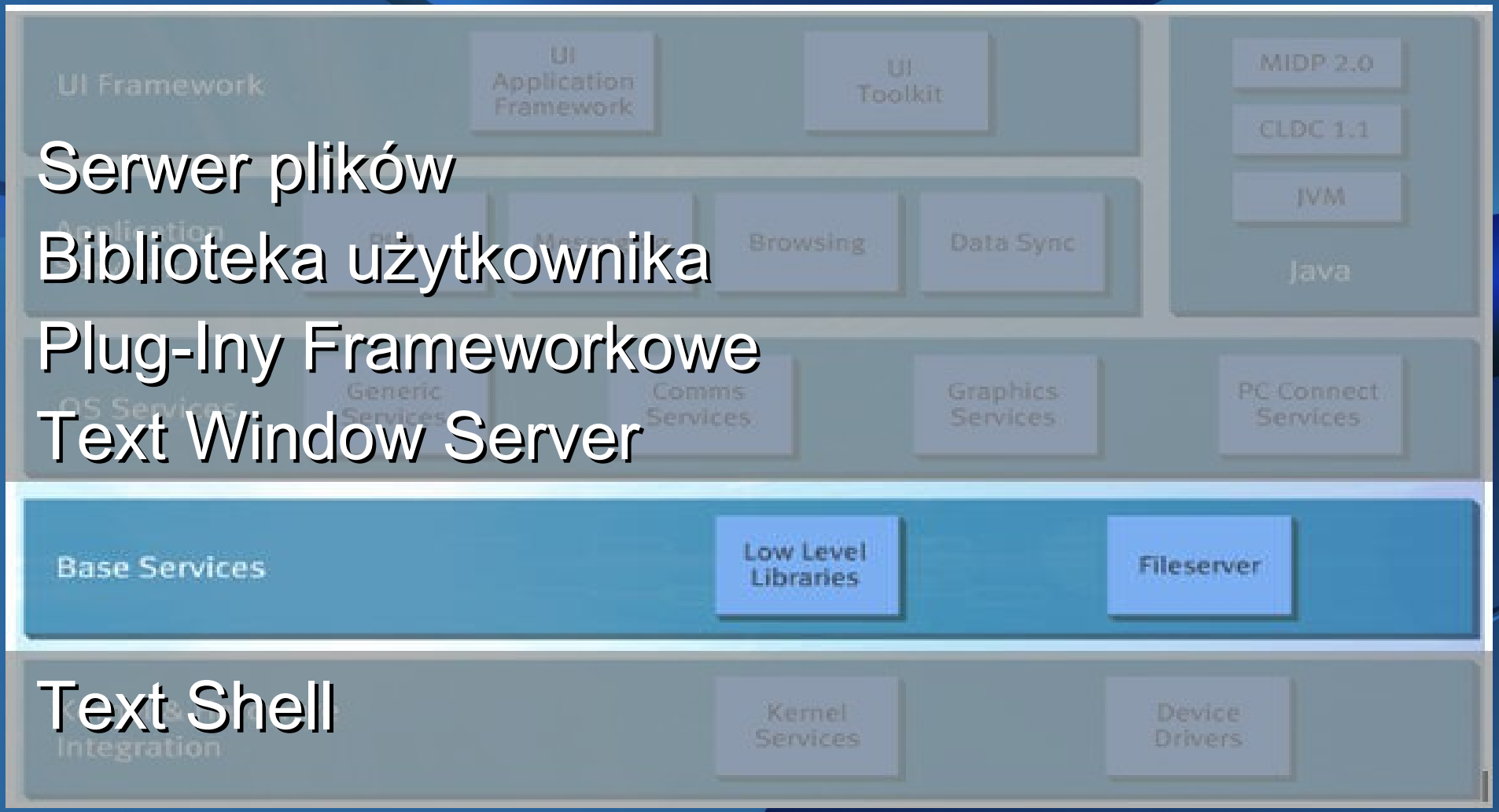


Jądro i sterowniki

Procesor: ARM
Jądro: EKA2
- wielowątkowy,
- wspiera mechanizmy czasu rzeczywistego
- „mikrojądro”, łatwe do specjalizowania
- większe wspomaganie dla instrukcji ARMa
- ulepszenia dla smartphonów
- większa energooszczędność



Symbian – Podstawowe biblioteki



Serwer plików

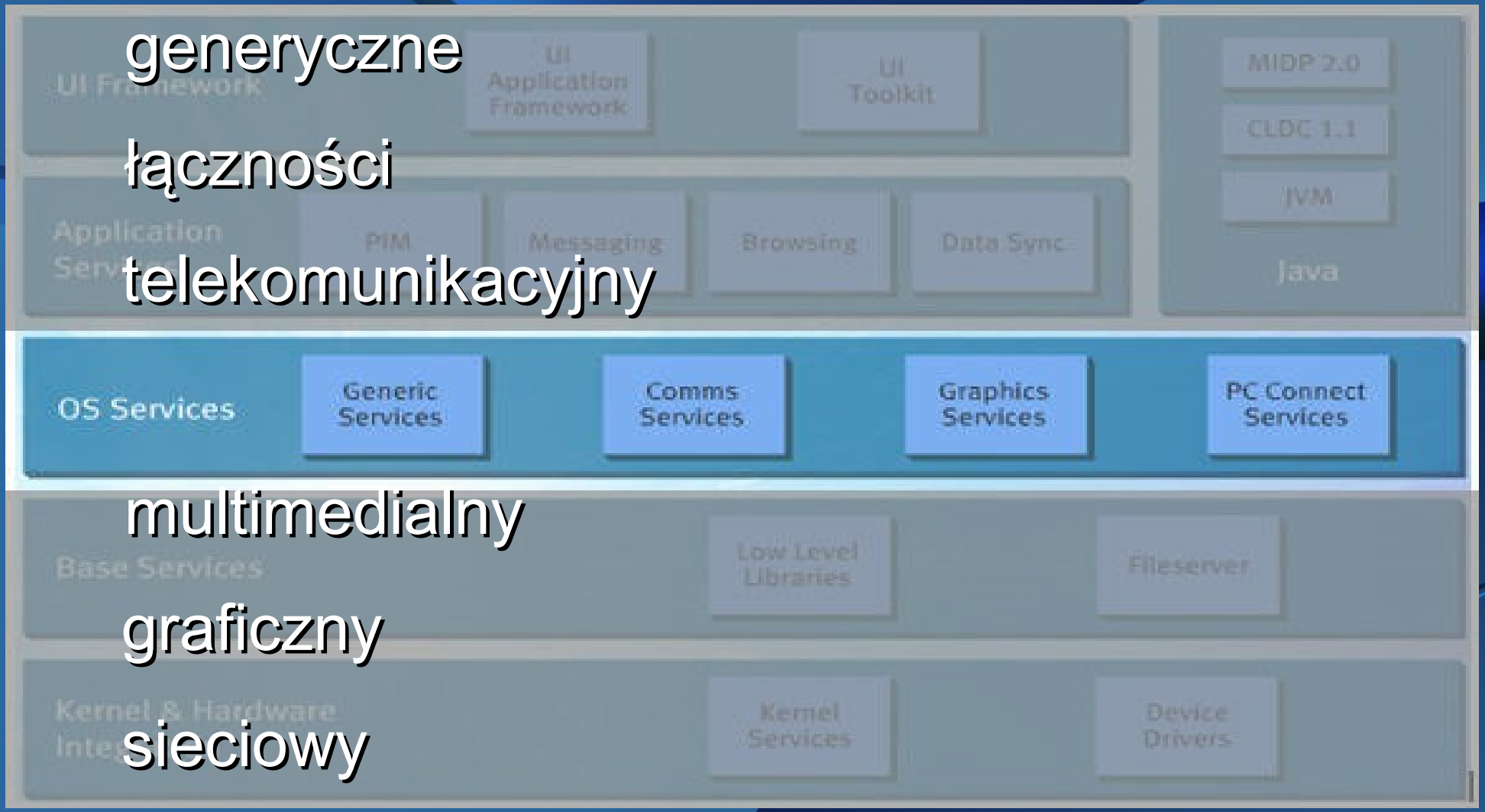
Biblioteka użytkownika

Plug-Iny Frameworkowe

Text Window Server

Text Shell

Symbian – OS Services



Symbian - AS

Biurowy

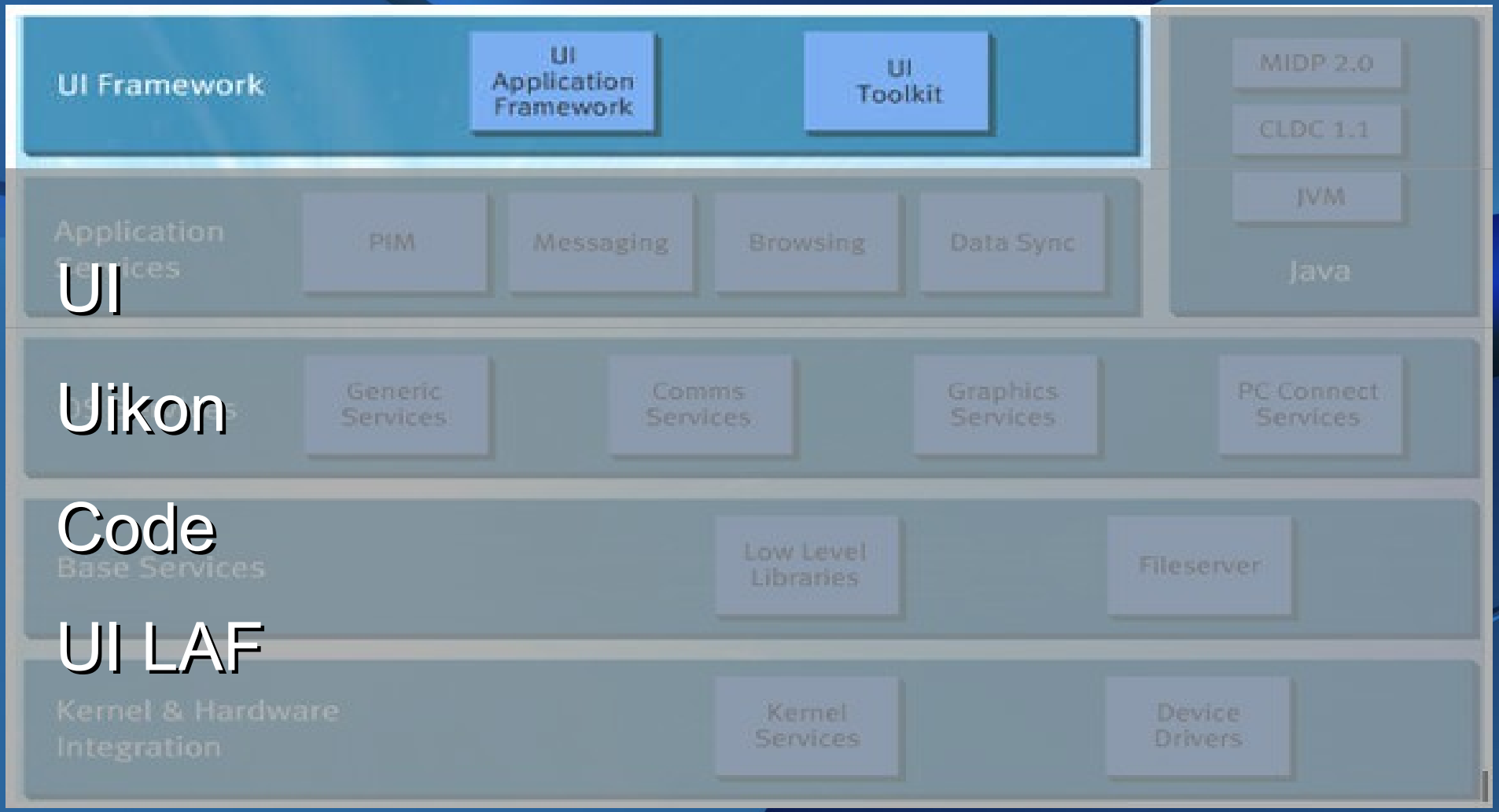
Synchronizacja daty i czasu

Internet

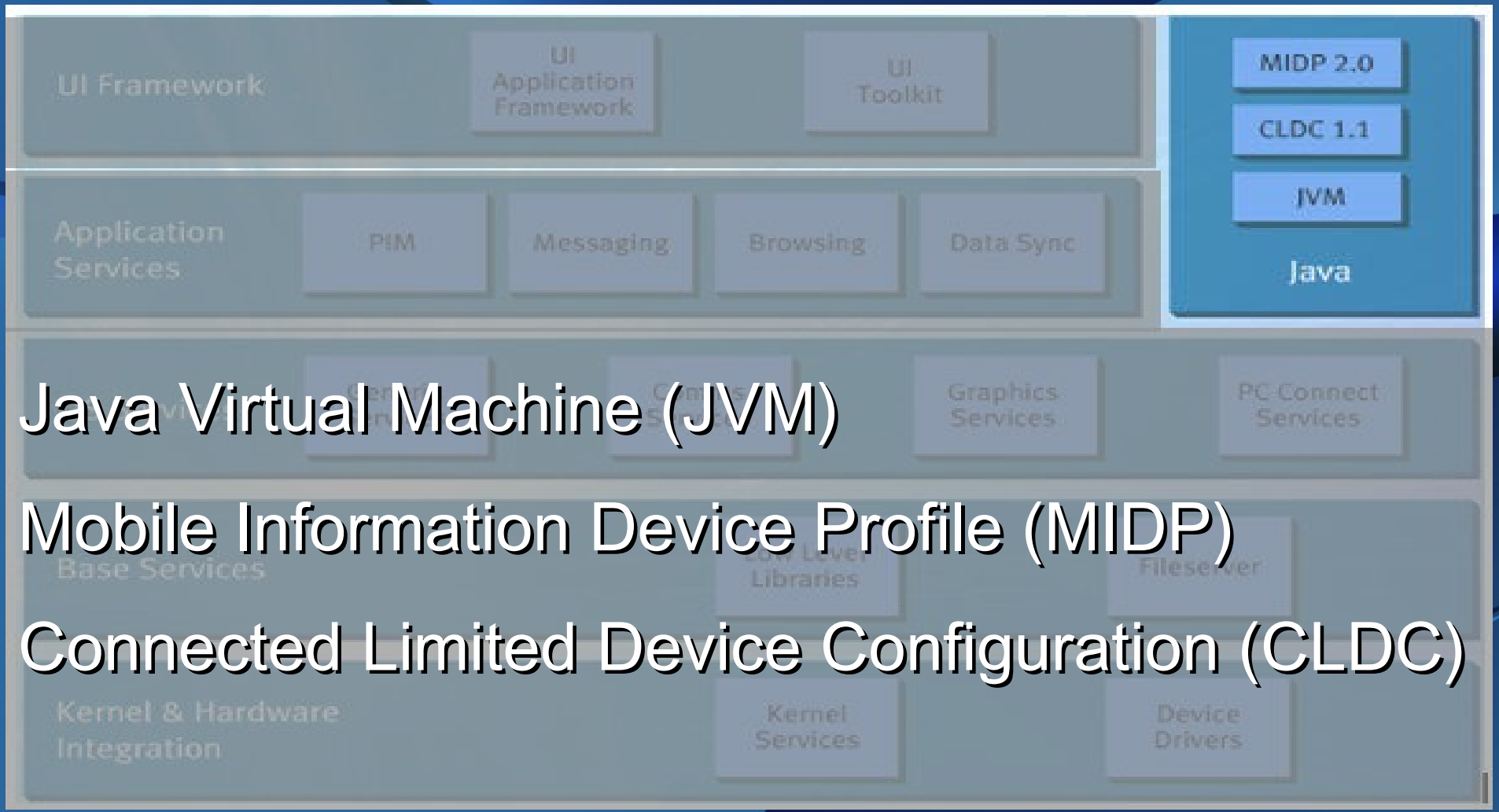
Wiadomości

Drukowanie

Symbian – UI Framework



Symbian - JAVA



Java Virtual Machine (JVM)

Mobile Information Device Profile (MIDP)

Connected Limited Device Configuration (CLDC)

Symbian – oprogramowanie

- Natywny język: C++
  - modyfikacje zmniejszające rozmiar binariów
- SDK dla poszczególnych rodzin urządzeń
  - kompilator GCC
  - emulator WINS systemu Symbian pod Windows
- IDE: Carbide.c++
  - plugin do Eclipse

Symbian – oprogramowanie

- Zmiany w C++
  - deskryptory
  - typy proste
  - stos dealokacji
  - mechanizm Leave/Trap obsługi wyjątków
  - brak STL
  - dwufazowe konstruktory
- Problemy
  - różne interfejsy graficzne
  - złożoność systemu

Symbian – programowanie

Inne języki:

- OPL
- Python
- Visual Basic
- Perl
- .NET (komercyjny)
- Java ME

Symbian Signed

- autorzy oprogramowania płacą za możliwość cyfrowego podpisania aplikacji
- zmniejszenie popularności w środowiskach FOSS

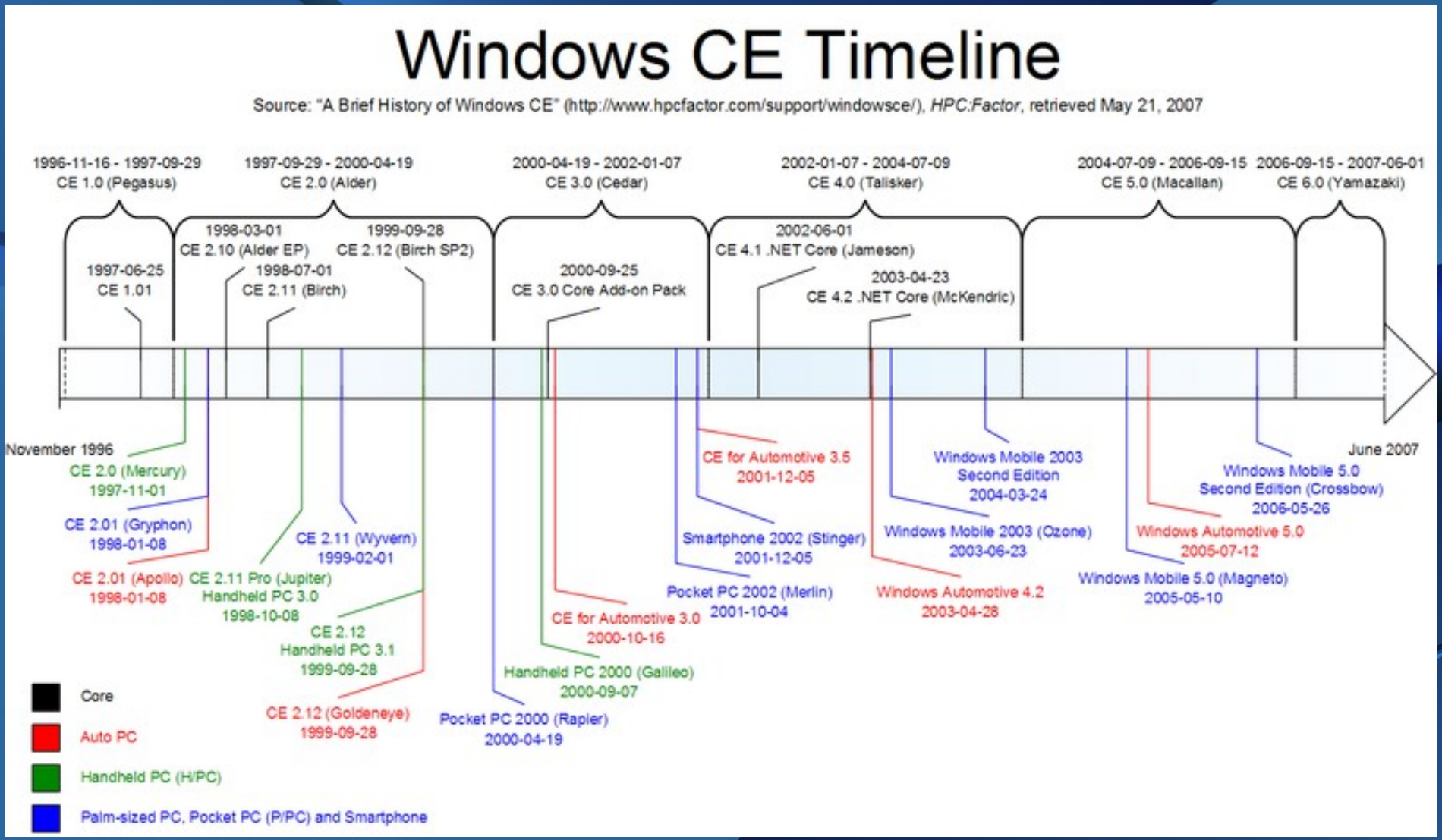
Symbian - bezpieczeństwo

- Zabezpieczenia oparte na „możliwościach” (capabilities)
  - niepodpisane aplikacje nie mogą wykonywać niebezpiecznego kodu
- Możliwe „zhackowanie” systemu
  - umożliwia uruchamianie niepodpisanego kodu.

Windows CE

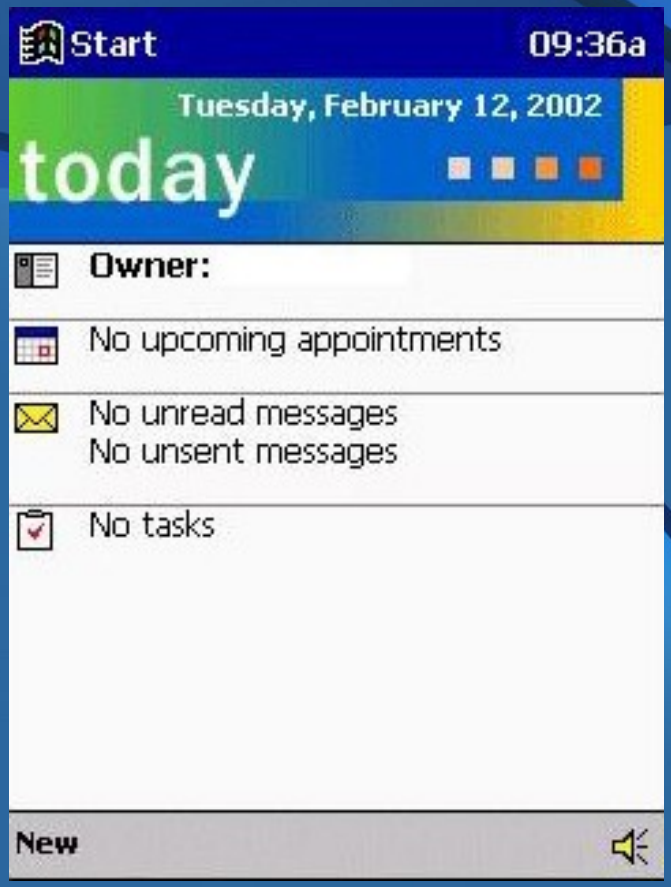
- 1995 – Windows Pegasus (CE 1.0)
  - urządzenia zasilane dwiema bateriami AA
  - 2 MB RAM
  - ekran dotykowy 480x240 w 4 odcieniach szarości
  - wielkość nieprzekraczająca 18 x 10 x 2,5 cm

Windows CE – historia



Windows Mobile – historia

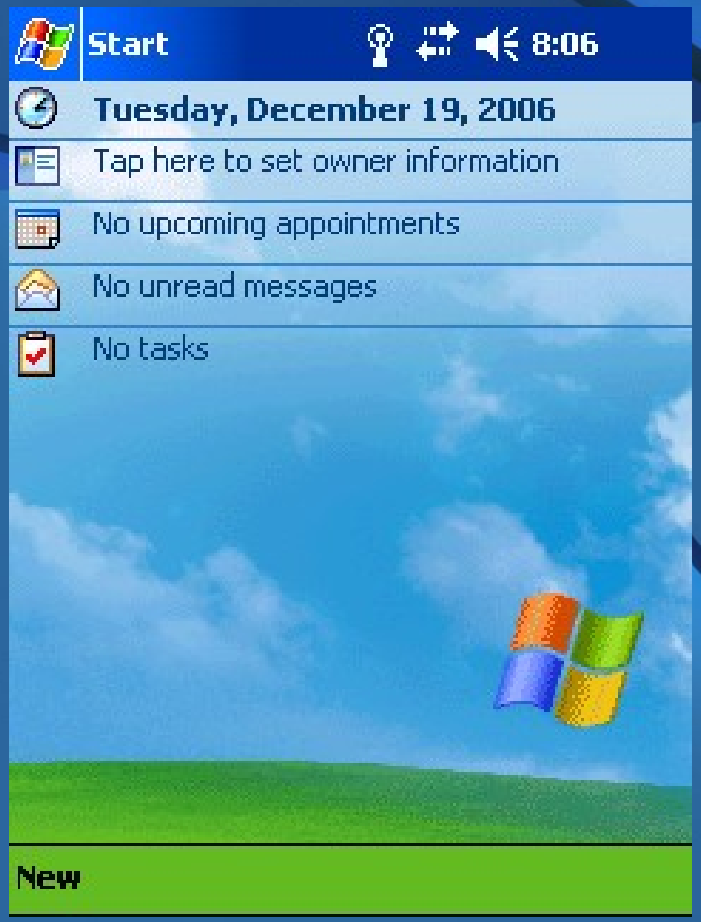
Pocket PC 2000



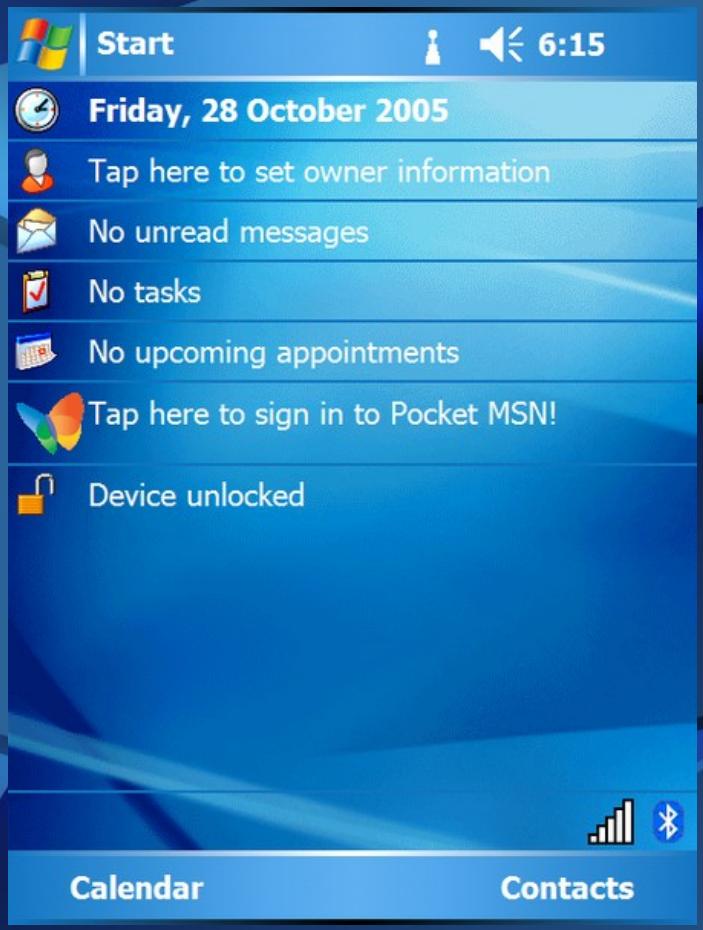
Pocket PC 2002

Windows Mobile – historia

Windows Mobile 2003

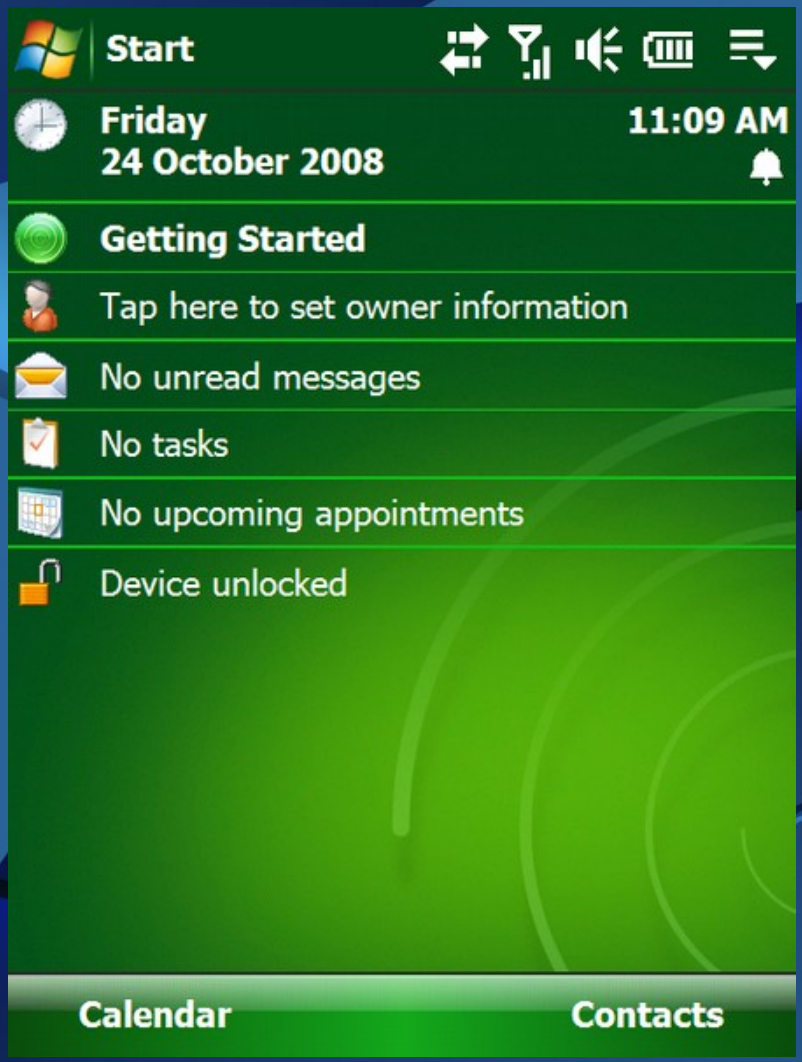


Windows Mobile 5.0



Windows Mobile – historia

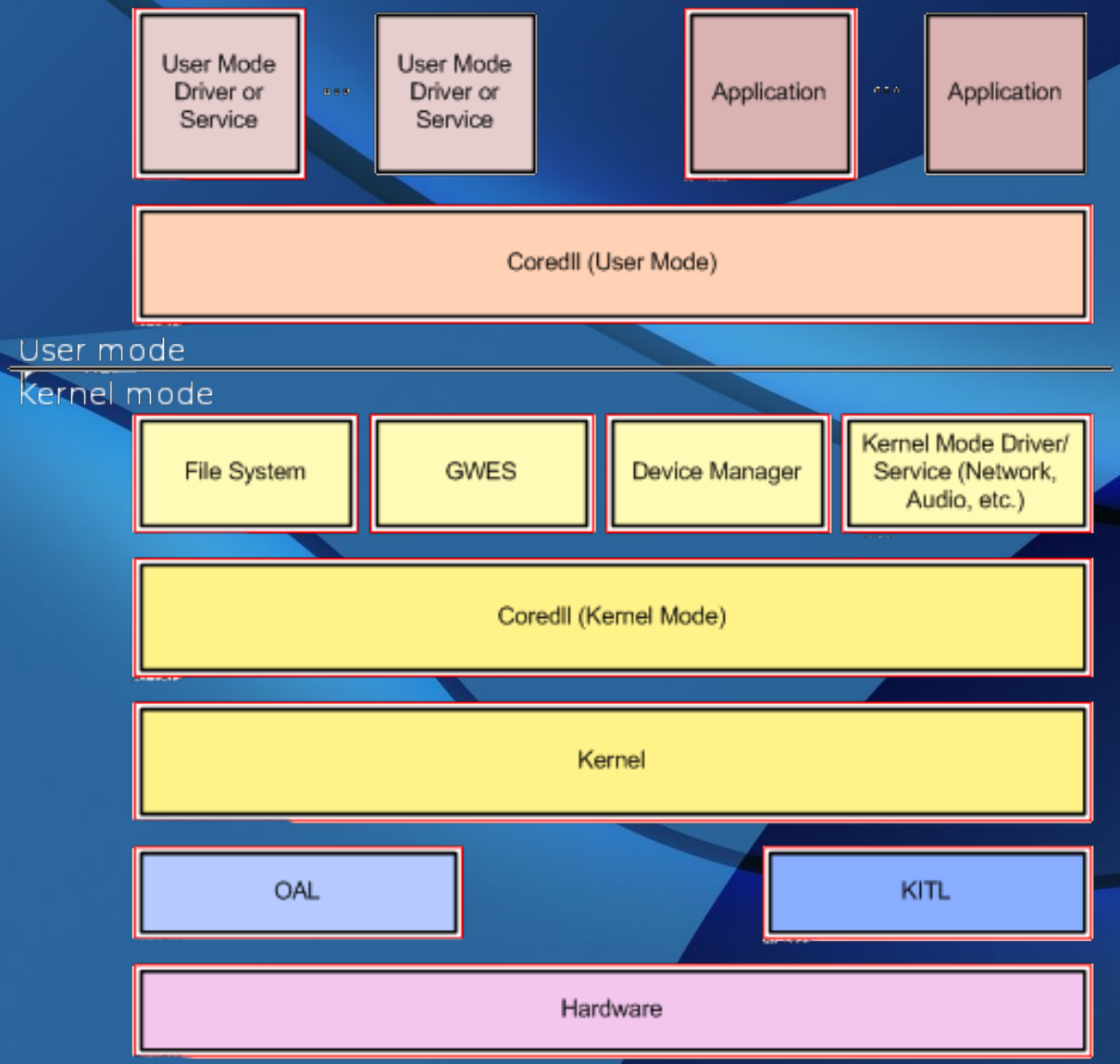
- Windows Mobile 6
  - Standard
    - smartfony (bez ekranu dotykowego)
  - Professional
    - PocketPC z telefonem
  - Classic
    - PocketPC



Windows CE

- Rdzeń systemu operacyjnego
  - system czasu rzeczywistego
    - deterministyczne opóźnienie przerwań
    - 256 poziomów priorytetu wykonywania
  - wielowątkowość
  - wieloplatformowość
    - obsługuje systemy z < 1MB pamięci operacyjnej
    - również systemy bez pamięci dyskowej i systemy na układach ROM

Windows CE 6.0 – architektura



Windows CE – architektura

- OAL – OEM Adaptation Layer
  - obsługa przerwań sprzętowych
  - zarządzanie energią
  - kanały komunikacji
- KITL – Kernel Independent Transport Layer
  - umożliwia debugowanie
  - działa w kanale komunikacji jądro-sprzęt

Windows CE 6.0 – architektura

- GWES – Graphics, Windowing and Events
  - łączy:
    - WinAPI – interfejs systemowy Windows
    - interfejs użytkownika
    - GDI – interfejs urządzeń graficznych
  - mechanizm komunikacji pomiędzy użytkownikiem, aplikacją i systemem operacyjnym

Windows CE – tworzenie systemów

- Platform Builder
  - zestaw narzędzi do MS Visual Studio
  - tworzenie BSP (Board Support Package):
    - sterowniki urządzeń
    - bootloader
    - OAL
  - katalogi komponentów, które można dodawać do systemu

Windows Mobile – tworzenie aplikacji

- MS Visual Studio
- kod natywny w C++
- kod zarządzany przez maszyny wirtualne
  - Visual Basic.NET
  - C#
    - .NET Compact Framework
  - Java

Windows Mobile – bezpieczeństwo

- Local Authentication Subsystem (LASS)
  - mechanizm uwierzytelniania wspierający wiele sposobów potwierdzania tożsamości
  - Local Authentication Plugin (LAP)
  - Authorization Event (EA)
- Szyfrowanie
  - CryptoAPI
  - CryptProtectData

Windows Mobile – zalety i wady

Zalety

- znajomy system
- framework do tworzenia aplikacji

Wady

- interfejs użytkownika
- wysokie wymagania sprzętowe

„The features are there, but actually using these features is another story.”

Derek Snyder, Product Manager, Microsoft

Linux

Openmoko

W zasadzie normalna dystrybucja Linuxa:

- Jądro 2.6
- GNU libc
- X.Org Server 7.1
- Matchbox WM
- GTK+

Openmoko – sprzęt

Neo FreeRunner GTA02

Sprzęt powstały na zasadach Open Hardware

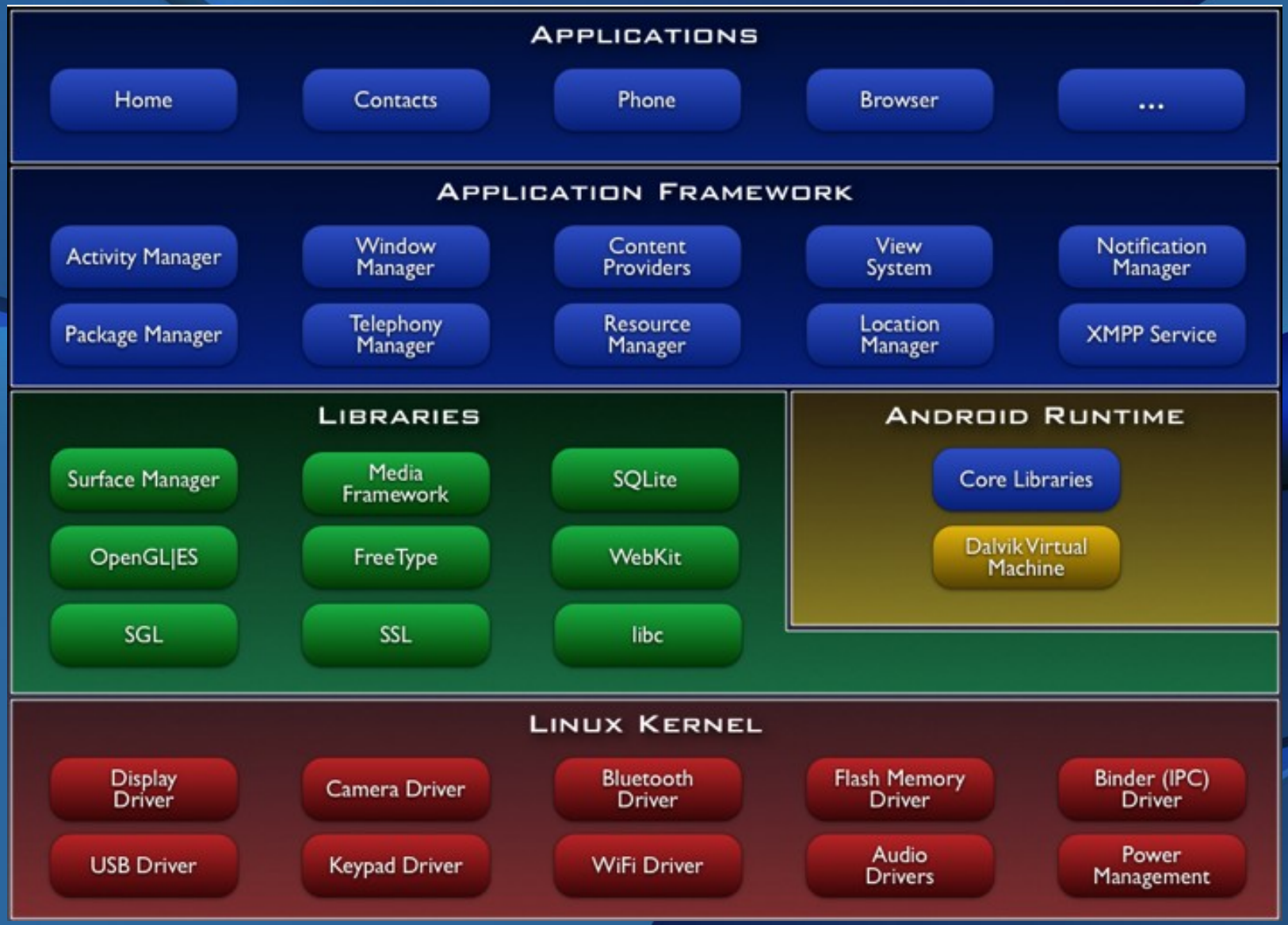
Dostępny od kwietnia 2008 dla użytkowników, u polskiego dystrybutora za 1360 zł

CPU Samsung 2442B (ARM) 400 MHz
128 MB RAM
256 MB pamięci Flash + czytnik SD
dotykowy ekran 480 x 600 pikseli

Android

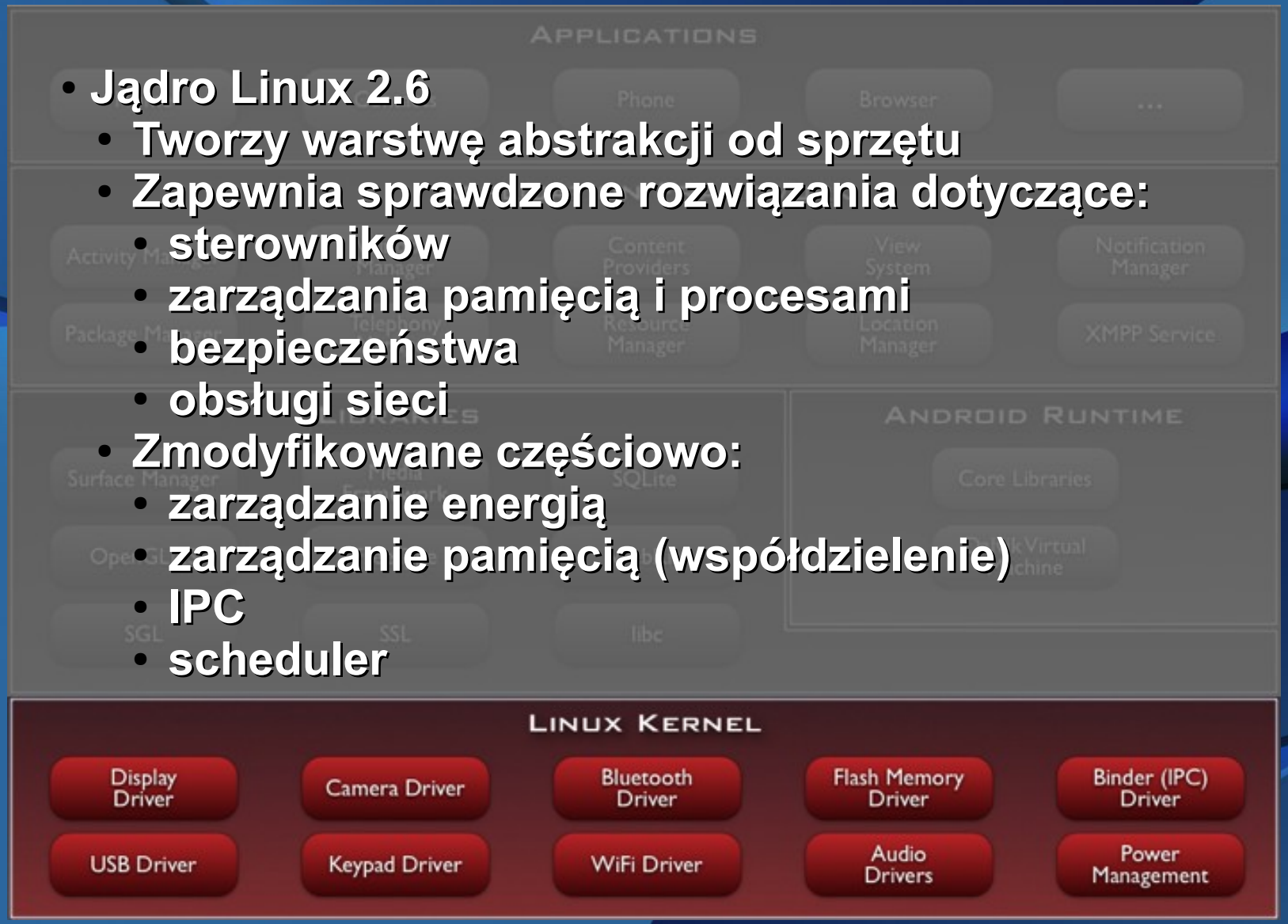
- Platforma mobilna oparta na jądrze Linux 2.6
- twórca: OpenHandset Alliance (Google + dziesiątki innych firm) i społeczność open source
- Szybki rozwój ($$$$...)
- Licencja Open Source?

Android – architektura

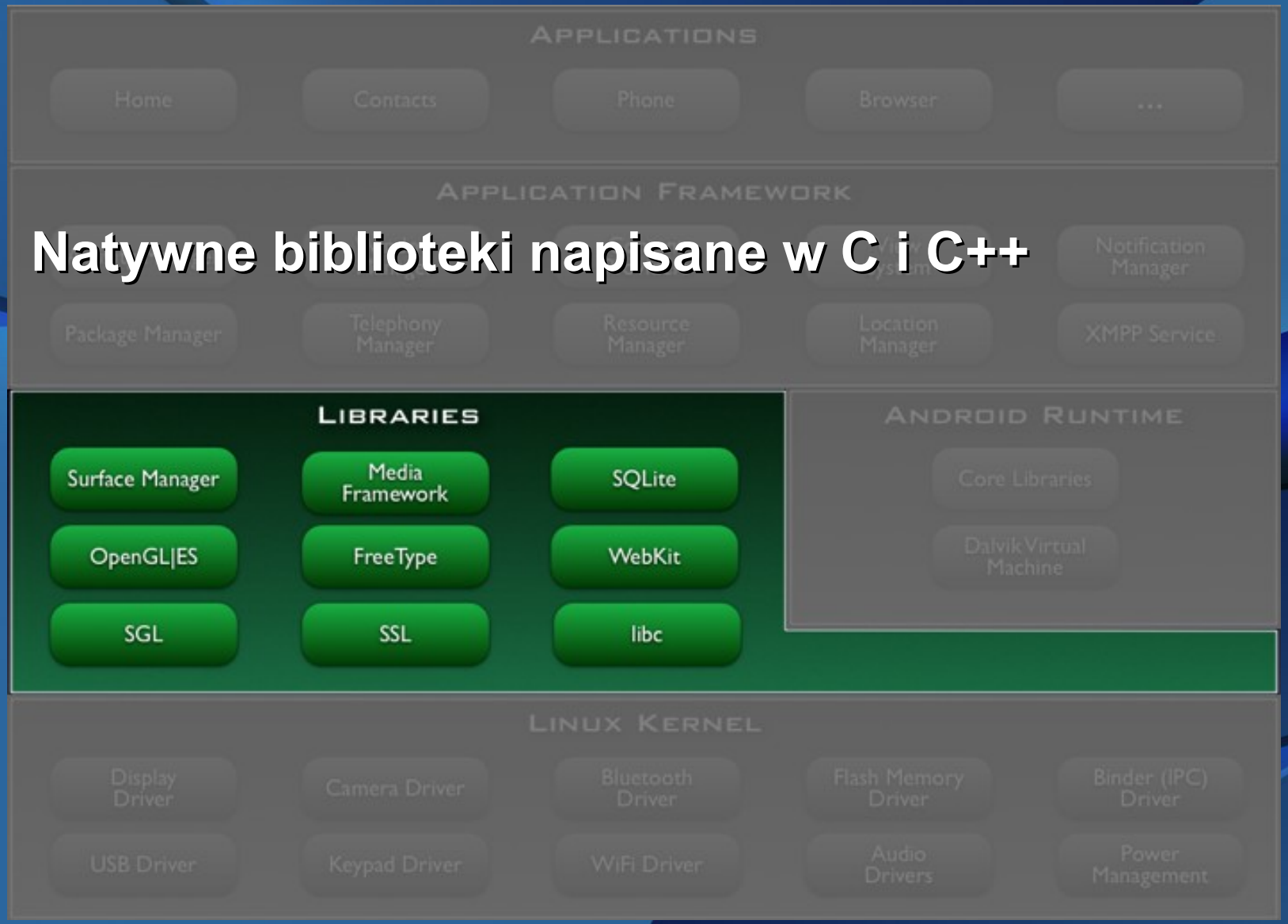


Android – architektura

- Jądro Linux 2.6
  - Tworzy warstwę abstrakcji od sprzętu
  - Zapewnia sprawdzone rozwiązania dotyczące:
    - sterowników
    - zarządzania pamięcią i procesami
    - bezpieczeństwa
    - obsługi sieci
  - Zmodyfikowane częściowo:
    - zarządzanie energią
    - zarządzanie pamięcią (współdzielenie)
    - IPC
    - scheduler



Android – architektura



Natywne biblioteki napisane w C i C++

Android – architektura

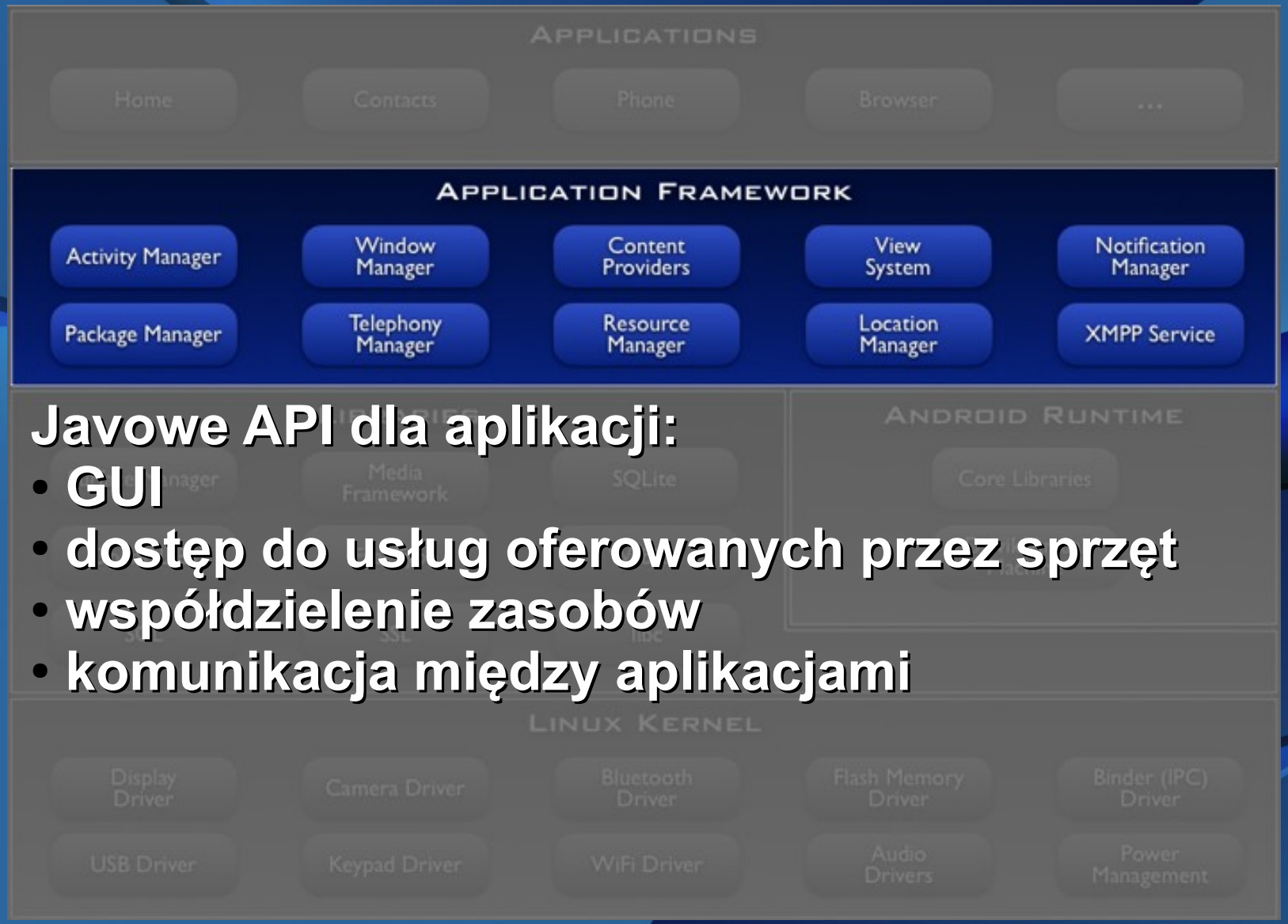
Część systemu stworzona specjalnie dla potrzeb platformy Android:

- Maszyna Wirtualna Dalvik
  - Wykonuje kod zarządzany (pliki .dex), do którego kompilowane są aplikacje.

  Taka konwersja pozwala wykonywać kod dużo efektywniej, między innymi zapewnia możliwość współdzielenia struktur danych, gdzie to możliwe

- Biblioteka standardowa napisana w Javie (operacje na typach podstawowych, kolekcje...)

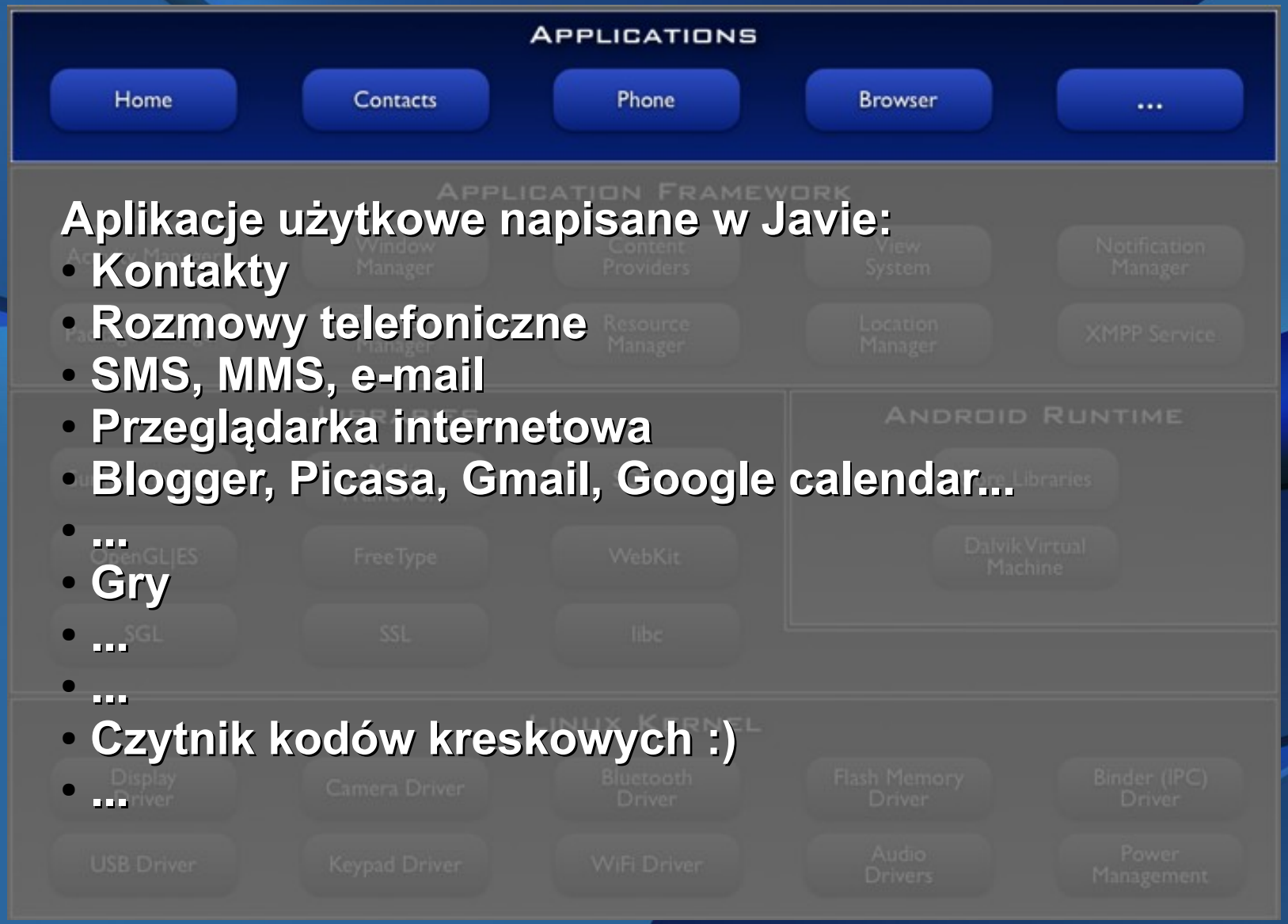
Android – architektura



Javowe API dla aplikacji:

- **GUI**
- **dostęp do usług oferowanych przez sprzęt**
- **współdzielenie zasobów**
- **komunikacja między aplikacjami**

Android – architektura

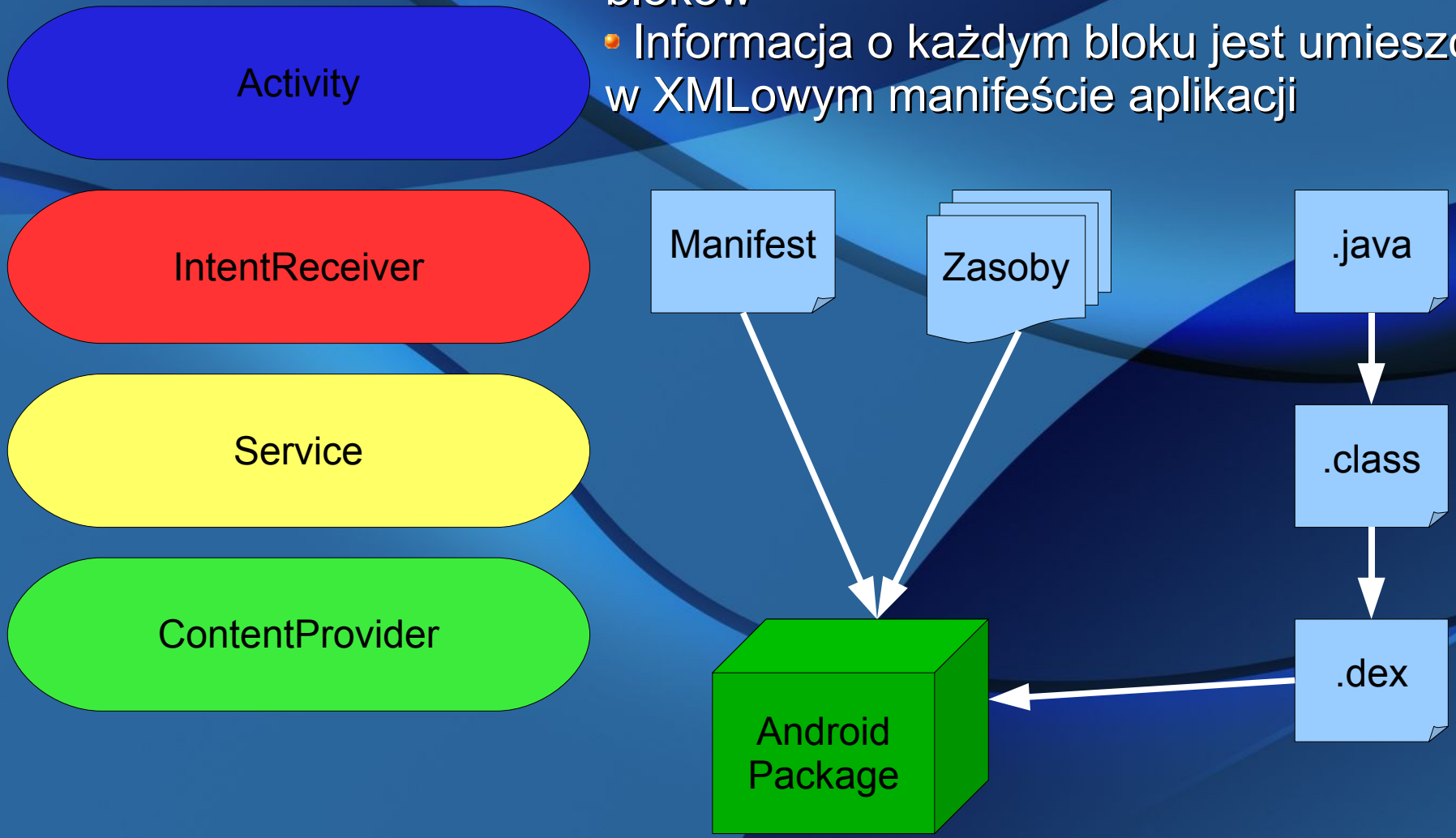


Aplikacje użytkowe napisane w Javie:

- **Kontakty**
- **Rozmowy telefoniczne**
- **SMS, MMS, e-mail**
- **Przeglądarka internetowa**
- **Blogger, Picasa, Gmail, Google calendar...**
- **...**
- **Gry**
- **...**
- **...**
- **Czytnik kodów kreskowych :)**
- **...**

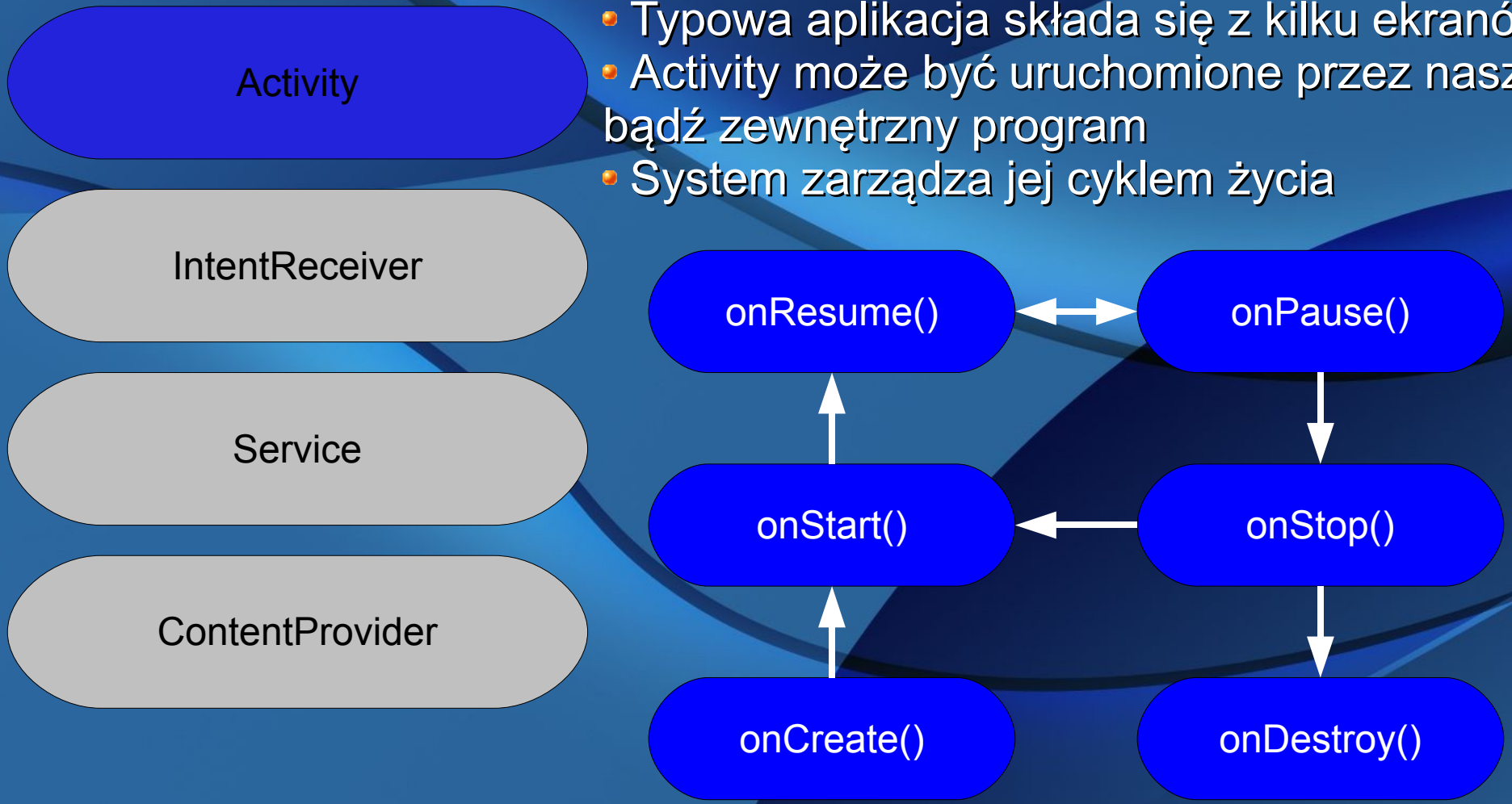
Android – aplikacja

- Aplikacja składa się z czterech rodzajów bloków
- Informacja o każdym bloku jest umieszczona w XMLowym manifeście aplikacji



Android – Activity

- Ekran aplikacji
- Typowa aplikacja składa się z kilku ekranów
- Activity może być uruchomione przez nasz bądź zewnętrzny program
- System zarządza jej cyklem życia



Android – Intents

Część aplikacji odpowiedzialna za reagowanie na żądania generowane przez inne aplikacje i na zdarzenia generowane przez system

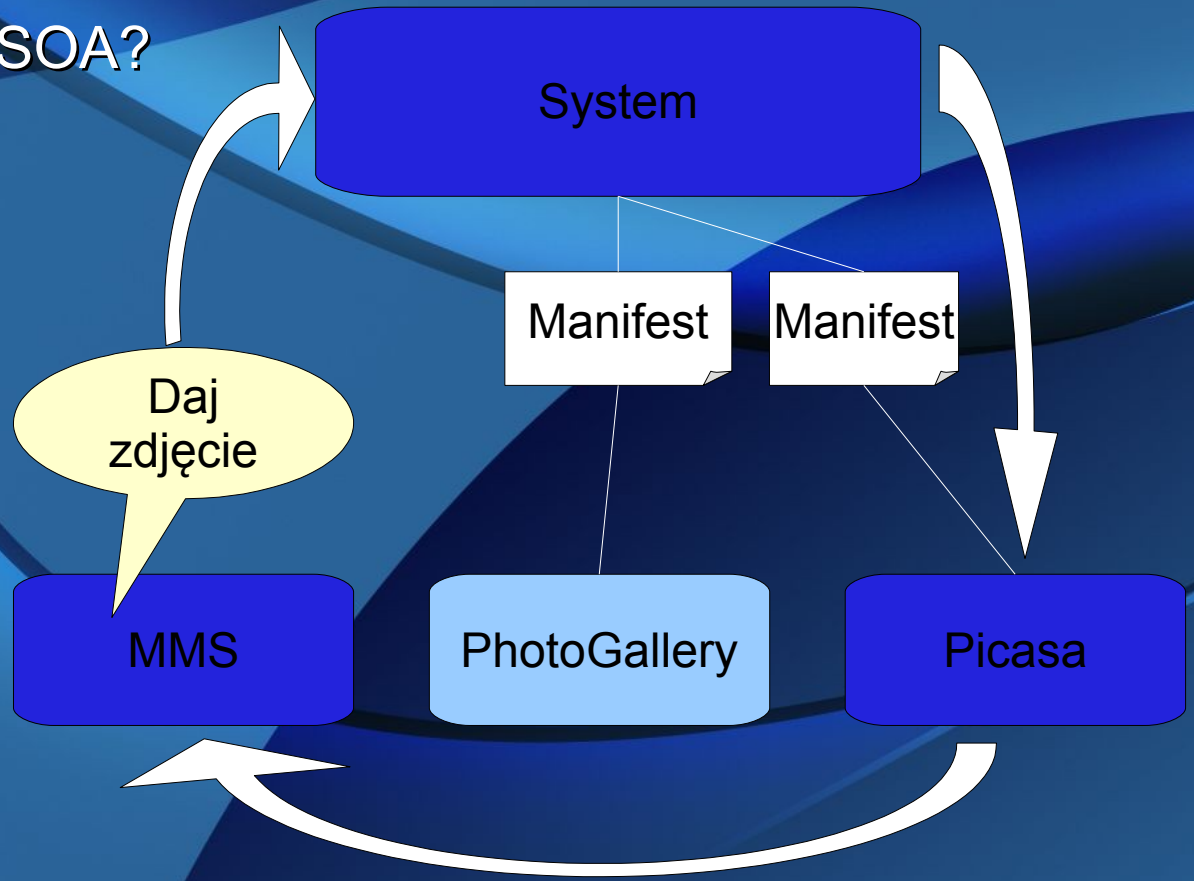
Activity

IntentReceiver

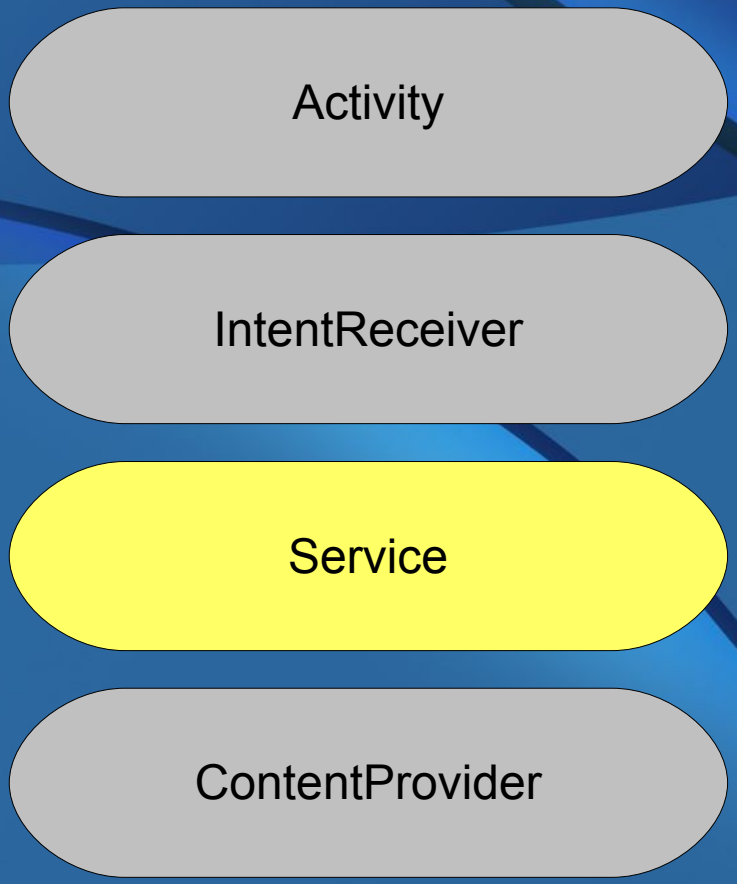
Service

ContentProvider

SOA?



Android – Service



Wątek działający w tle

Android – ContentProvider

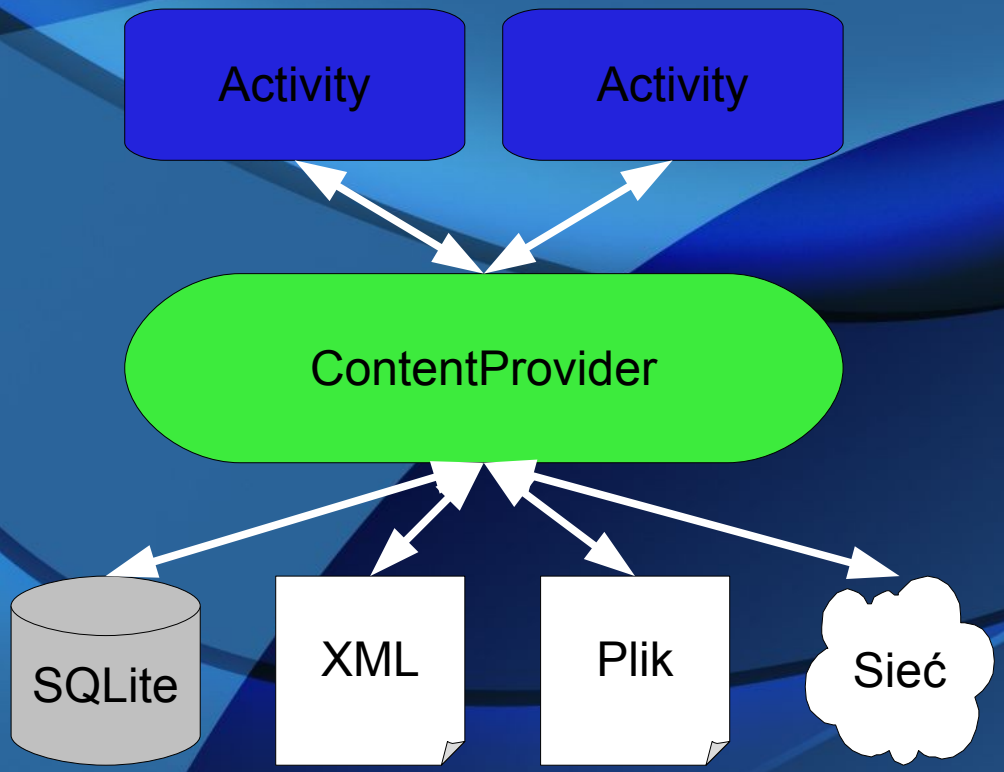
Activity

IntentReceiver

Service

ContentProvider

Mechanizm współdzielenia danych między aplikacjami



Android – bezpieczeństwo

```
<?xml version="1.0" encoding="utf-8"?>
    <manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.my_domain.app.helloactivity">
        <uses-permission android:name="android.permission.RECEIVE_SMS" />
        <application android:label="@string/app_name">
            (...)
        </application>
    </manifest>
```

Każda aplikacja musi określić w Manifeście, jakie uprawnienia są jej potrzebne do działania. Przykładowe uprawnienia:
BATTERY_STATS, BLUETOOTH, CALL_PHONE, CAMERA, CHANGE_CONFIGURATION, CHANGE_WIFI_STATE, DELETE_PACKAGES, GET_TASKS, INTERNET, READ_CALENDAR, READ_SMS, REBOOT, RECEIVE_MMS, SEND_SMS, SET_WALLPAPER, STATUS_BAR, VIBRATE, WRITE_SMS

Android – T-Mobile G1

Dostępny od października 2008 w USA ($399), w Europie środkowej już niedługo
dwurdzeniowe CPU/GPU Qualcomm (ARM), 528 MHz
192 MB DDR SDRAM

Urządzenia mobilne

Poza urządzeniami które intuicyjnie nazywamy „mobilnymi” mamy jeszcze całą masę innych urządzeń:

- odtwarzacze MP3/MP4
- Aparaty cyfrowe
- odbiorniki GPS

A także: notebooki, kamery video, pagery, czytniki e-book’ów, przenośne konsole gier (GBA, PSP), pendrive, ...

Urządzenia mobilne

... na szczęście nie wystarczy czasu żeby się wszystkim przyjrzeć ;).

Pokażę 2 systemy operacyjne:

- RockBox (opensource SO na odtwarzacze MP3)
- VxWorks (na przykładzie aparatów Canona; wykorzystany też np. w Beoingu 747-8 i marsjańskiej sondzie kosmicznej O_o), wspomnę o CHDK i DryOS

Będzie też trochę o metodach podmiany takich systemów w urządzeniach... bo to ciekawe ^_^.

RockBox

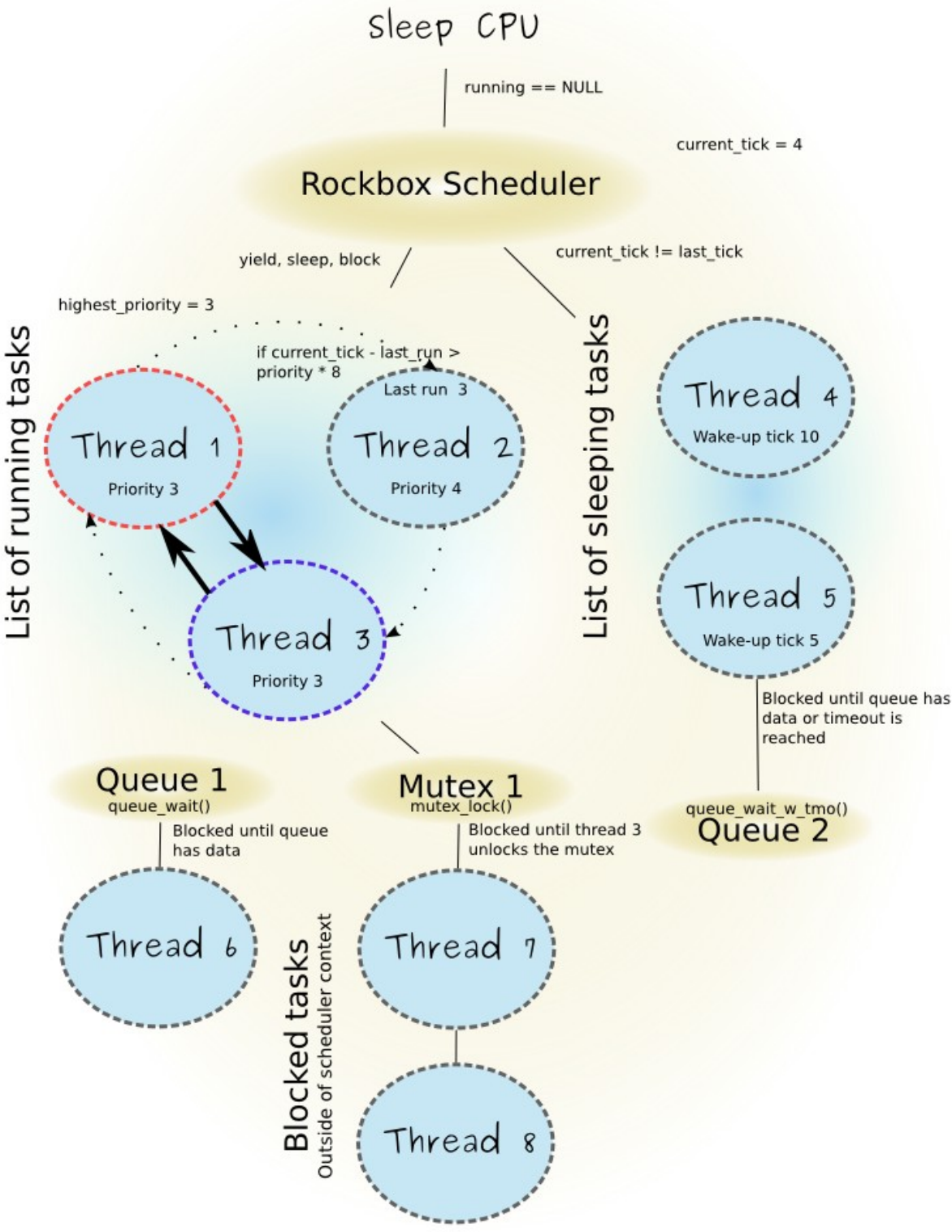
RockBox składa się z 2 części – **systemu operacyjnego** (zależny od architektury) i **aplikacji** (zależna tylko od możliwości hardware'u, np. czy jest graficzny wyświetlacz).

Taki podział pozwala działać RockBox'owi na urządzeniach różniących się drastycznie możliwościami – od wczesnych Archos'ów z 1-bitowymi wyświetlaczami tekstowymi, poprzez iPody z kolorowym ekranem wysokiej rozdzielczości, do najnowszych iRiver'ów z cyfrowym optycznym wyposażeniem audio i zaawansowanymi funkcjami nagrywania

RockBox - jądro

Jądro RockBox'a jest zaawansowanym kernelem ze wsparciem priorytetów, obsługą wątków z możliwością przesyłania danych między nimi i blokadami; wspiera także przerwania i proste timery.

Z ciekawszych rzeczy: wątek jest uruchamiany do czasu, gdy sam nie zwolni procesora (poza przerwaniami i timerami), wątki uruchamiane są w kolejności ich tworzenia :).



RockBox – architektura

Wątki używane są **tylko** w przestrzeni jądra, aplikacja uruchamiana jest w jednym wątku (zwanym Main thread). Poza tym mamy wątki:

- ATA thread (MPczynki z dyskiem twardym) - spinup/spindown
- Backlight thread – zarządzanie podświetleniem ekranu
- MMC thread (MPczynki z pamięcią flash) – hotswap karty
- Playback thread – niskopoziomowe nagrywanie/odtwarzanie
- Power thread - power management i ładowanie baterii
- Scroll thread - handles scrolling
- USB thread – informuje o podłączeniu/odłączeniu USB

RockBox – możliwości

- Mnóstwo kodeków audio i video (dla urządzeń odtwarzających programowo)
- gapless playback, crossfader, korektor graficzny, **`prawdziwy`** tryb losowego odtwarzania, TagCache (tagowanie muzyki), LastFm Scrobbler, dynamiczne playlisty, skórkowanie, sterowanie komendami głosowymi (dla osób niewidomych / zepsuty wyświetlacz), itp.
- Ponadto wtyczki: JPEG viewer, emulator GB i GBA, a także ZX Spectrum, Doom, MPEG video player, Doom (i inne gry), a także użyteczne (np. stroik do gitary).
- RockBox'a tworzy grupa szaleńców próbująca wycisnąć ze sprzętu tyle, ile się tylko da, np: uzyskanie 129 odcieni szarości na 1- lub 2-bitowych wyświetlaczach przez wykorzystanie niskiej częstotliwości odświerzania pasywnego; zwiększanie czasu działania urządzenia przez maksymalne wykorzystanie RAMu na bufor dźwięku, itp.

RockBox

Instalacja RockBox'a jest prawie bezbolesna. Flash odtwarzacza:

* Reset vector (0x000000 - 0x000007)
* Original firmware (0x000008 - 0x1EFFFF)
* Bootloader (0x1F0000 - 0x1FFFFF)

Reset vector = wskaźnik stosu i entry-point firmware'u.

Dzięki temu, że na końcu jest trochę wolnego miejsca doklejamy tam bootloader i zmieniamy Reset Vector. Bootloader umożliwia dual boot (oryginalny firmware, lub RockBox z dysku/karty).

Można też wgrać firmware do pamięci – wtedy trochę inny układ...

VxWorks

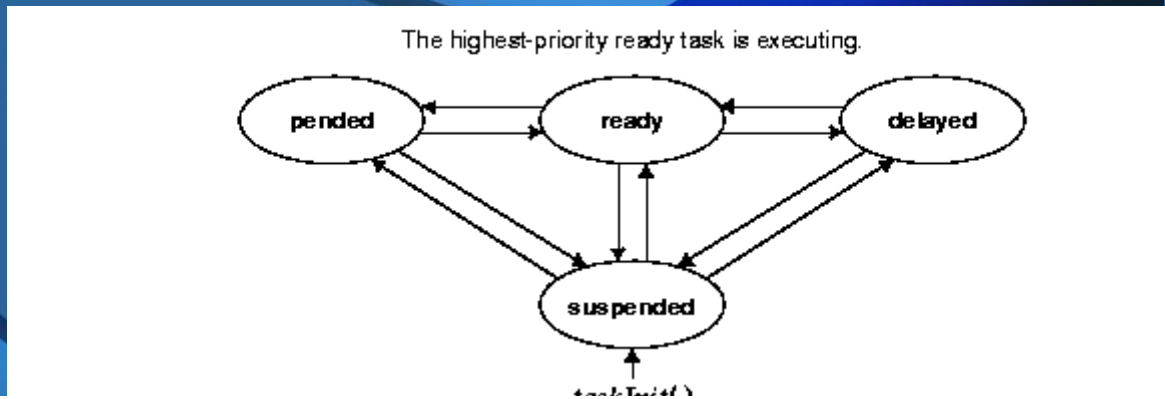
VxWorks jest dużo bardziej zaawansowany od RockBox'a, jednak będziemy skupiać się na jego podstawowych właściwościach (tych mających zastosowanie w urządzeniach mobilnych).

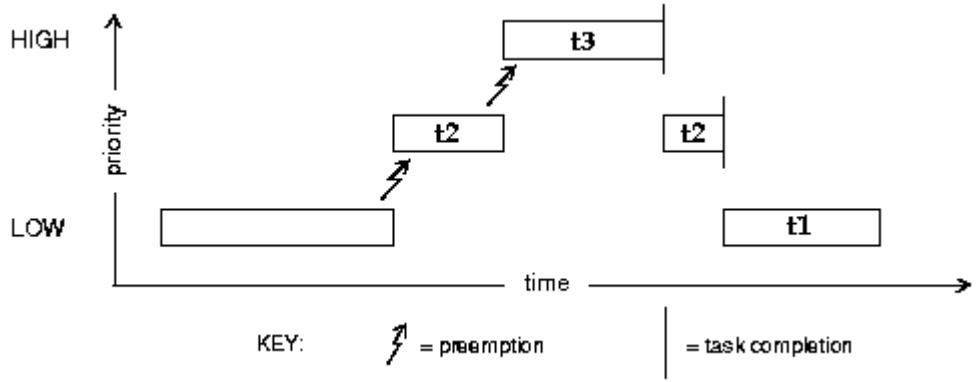
VxWorks jest komercyjnym produktem firmy Wind River (źródła nie są dostępne). Dlatego też nie ma **dokładnej** wiedzy o architekturze tego systemu. Mimo to wiele można dowiedzieć się z pomocy do Tornado(IDE do VxWorks 5.x) oraz dzięki pracy deweloperów CHDK (**C**anon **H**ack **D**evelopment **K**it).

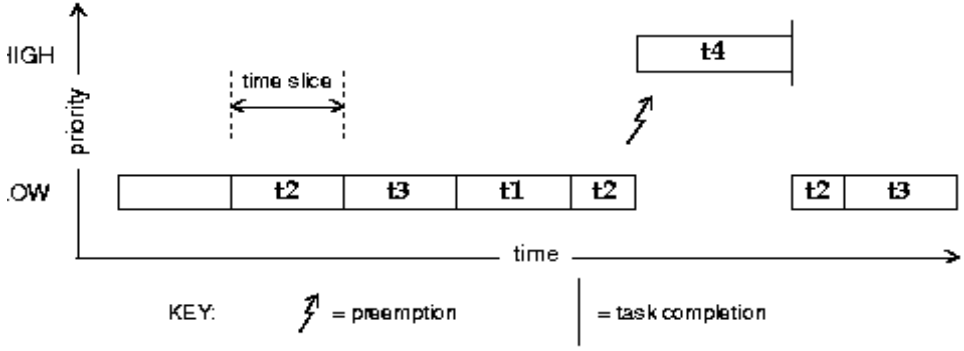
Będziemy mówić tu o aparatach Canon PowerShot opartych o DIGIC II i DIGIC III (wczesne wersje), później (prawdopodobnie ze względu na umowy koncesyjne Canon przeszedł na DryOS'a – ich własny system operacyjny będący w zastosowaniu już wcześniej np. na kamerach wideo Sony).

VxWorks - jądro

Jądro VxWorks operuje na **taskach** (uproszczona wersja POSIX threads).Taski są **wywłaszczalne** (dużo większy kontekst procesu, brakuje tylko przestrzeni adresowej, jako że przestrzeń jest wspólna). Ponadto mamy obsługę priorytetów (256). Scheduling na dwa sposoby – **Preemptive Priority** i **Round-Robin**. Możliwość blokady wywłaszczania i przerwań, współdzielenie kodu, zmienne globalne, ... oraz POSIXowe interfejsy.







VxWorks - jądro

Metody komunikacji międzyprocesowej:

- Shared memory, for simple sharing of data.
- Semaphores, for basic mutual exclusion and synchronization.
- Mutexes and condition variables for mutual exclusion and synchronization using POSIX interfaces.
- Message queues and pipes, for intertask message passing within a CPU.
- Sockets and remote procedure calls, for network-transparent intertask communication.
- Signals, for exception handling.
- Events (synchroniczne)

VxWorks - I/O

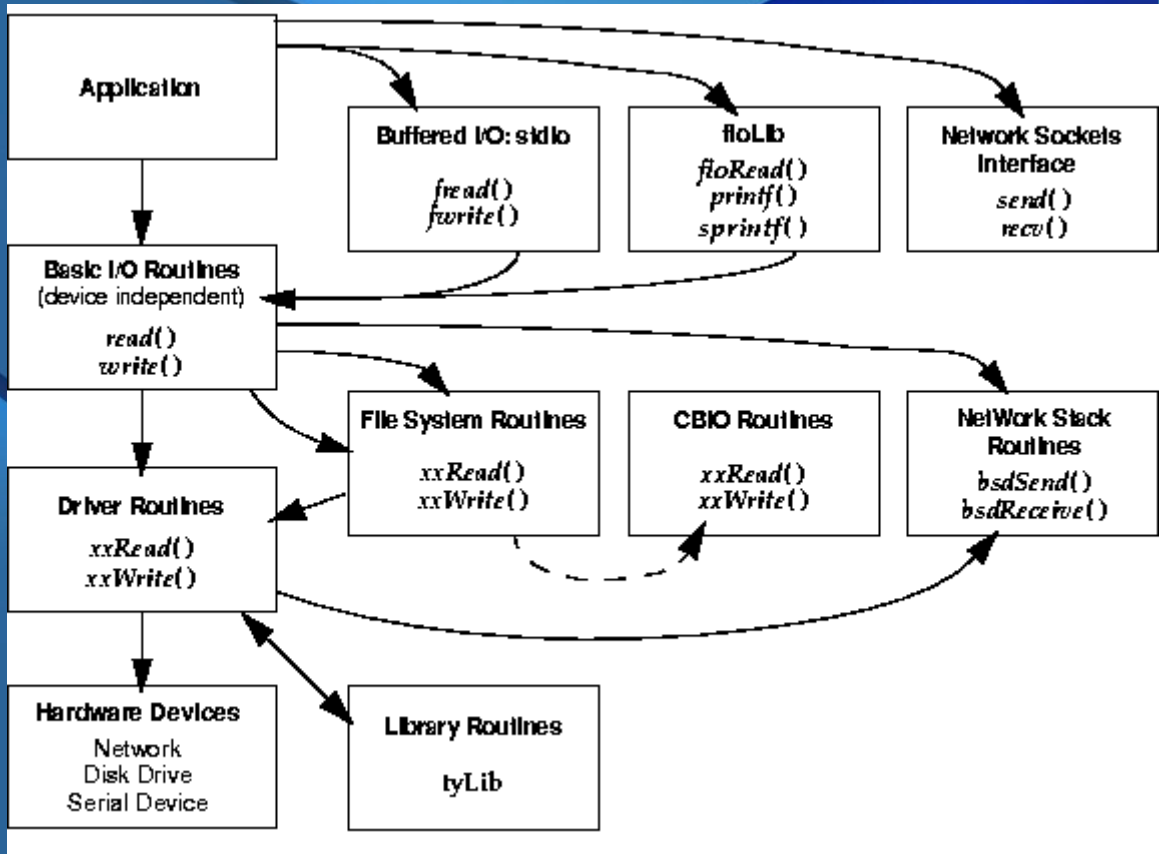
Wszystko jest plikiem. Deskryptory plików są **globalne** (wszystkie taski mogą do nich pisać). Buffered I/O, formatted I/O, Asynchronous I/O. **Urządzenia**: serial(pseudo-tty), pipe, pseudo-memory, NFS (i inne sieciowe), block device), ...

Uwaga: w VxWorks funkcje sterowników urządzeń uruchamiane są w kontekście wywołującego taska (więc **są wywłaszczalne(!)**)

/usr/myfile

/pipe/mypipe

/tyCo/0



VxWorks + Canon

VxWorks jest **zaawansowanym systemem czasu rzeczywistego**. Skąd wiemy, że Canon go używa(ł) i na których aparatach?

Jak zwykle stało się to dzięki grupce hakerów próbujących dowiedzieć się jak działa to co trzymają w łapkach (i jak to działanie usprawnić :)). Oczywiście należało zdobyć oryginalny firmware w celu analizy. Niestety Canon wydawał tylko „firmware update” nie zawierające pełnego obrazu systemu. Mimo to udało się odszyfrować tego updatera i na podstawie jego napisać własny, który umożliwiłby zrzut całej pamięci (odpowiednio go podpisując, tak by aparat myślał, że to oryginalny).

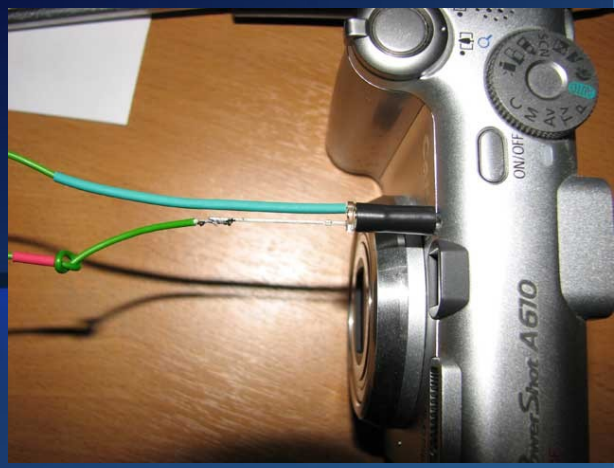
Gdyby to było takie proste...

VxWorks + Canon

... bo skąd mamy wiedzieć jaki jest adres funkcji do zapisu danych na kartę pamięci?

Dlatego też w ponad połowie przypadków firmware uzyskano przesyłając ją **przez diodę(!)**. Do odbioru takich danych wystarczy fotodioda podłączona pod wejście mikrofonowe więc można to zrobić metodą domową. Wysłanie całej zawartości pamięci (prostym protokołem) zajmuje od 1 do 7 godzin.

Gdy mamy już firmware możemy przystąpić do żmudnego procesu analizy kodu assemblerowego i oznaczaniu odpowiednich funkcji (na szczęście duża część kodu jest podobna, więc można to zrobić automagicznie).



... i tak powstało CHDK

C H D K czyli **C** anon **H** acking **D** evelopment **K** it nie zastępuje oryginalnego firmware ani VxWorks. Działa podobnie do firmware-updater'a – podmienia odpowiednie wywołania procedur w kodzie aplikacji uruchomionej pod VxWorks na własne i robi software_reset aparatu, by korzystał on z podmienionich funkcji. Firmware w pamięci flash **nie jest zmieniany** (w szczególności oznacza to, że trzeba ładować CHDK przy każdym starcie aparatu).

Krótki spis możliwości CHDK: RAW, nadpisanie parametrów aparatu i video (czasy, ISO, itp.), bracketing, **skryptowanie (!)**, wykrywanie ruchu, wykrywanie krawędzi, histogram na żywo, zebra-mode, konfigurowalne OSD, przeglądarka plików, czytnik e-booków, gry i wiele innych

VxWorks + CHDK

CHDK korzysta z możliwości jakie daje VxWorks. Działanie CHDK oparte jest (w tej chwili) na trzech taskach:

- PhySw – sterownik klawiatury
  - występuje w oryginalnym firmware, poza zdefiniowanymi klawiszami wywoływany jest oryginalny kod
  - korzysta z **timerów POSIXowych**, budzi się co 10ms i sprawdza, czy nie nastąpiło naciśnięcie klawisza, potem sprawdza czy jesteśmy w ALT-mode – jeśli nie, przekazujemy naciśnięcie do firmware aparatu
  - Jest silnikiem skryptowym. Jeśli uruchomiony jest jakiś skrypt, to wywoływany jest interpreter, żeby przetworzył następną komendę (dlatego każda komenda w *ubasic* zajmuje 10ms)
  - PhySw obsługuje wszystkie akcje zapoczątkowane przez użytkownika (np. skrótem klawiszowym)
- CaptSeqTask – robi zdjęcia.
  - tak jak PhySw jest to zmodyfikowana wersja istniejącego taska. Dodajemy obsługę formatu RAW i dark-frame substraction
- spytask – kontrola CHDK.
  - inicjuje część hardware-independent, rysuje chdk-GUI i zapisuje pliki RAW
  - Wywołuje funkcje niezależne od użytkownika (np. zależne od czasu)

Źródła

- Galeria Karola Kreńskiego, http://www.inf.sgsp.edu.pl/pub
- Wikipedia (http://en.wikipedia.org/wiki/Windows_Mobile/)
- MSDN (http://msdn.microsoft.com/en-us/library/aa924061.aspx)
- http://en.wikipedia.org/wiki/Openmoko_Linux
- http://wiki.openmoko.org/wiki/Main_Page
- Dystrybutor NeoFreerunner: http://www.mobilesolutions.pl/107,openmoko-neo-freerunner-gta02.html
- http://www.android.com
- http://code.google.com/intl/pl/android/
- Architektura Androida: http://www.youtube.com/watch?v=QBGfUs9mQYY
- Architektura Androida:
  www.openexpo.ch/fileadmin/documents/2008Zuerich/Slides/33_Printemps.pdf
- http://www.microsoft.com/poland/technet/bazawiedzy/centrumrozwiazan/cr227_01.mspx
- http://msdn.microsoft.com/en-us/library/
- http://www.codeproject.com/KB/mobile/Symbian_OS_design_faults.aspx
- http://gizmodo.com/gadgets/what.s-wrong-with-windows-mobile/whats-wrong-with-windows-mobile-and-how-wm7-and-wm8-are-going-to-fix-it-333536.php