AKADEMIA GÓRNICZO-HUTNICZA

IM. STANISŁAWA STASZICA W KRAKOWIE



Wydział Inżynierii Metali i Informatyki Przemysłowej



PRACA DYPLOMOWA MAGISTERSKA

pt.

„XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX”

Imię i nazwisko dyplomanta: **Paweł Popanda**

Kierunek studiów: **Informatyka Stosowana**

Specjalność:  **Modelowanie i Technologie Informacyjne**

Nr albumu: **226 141**

Promotor: dr inż. Łukasz Rauch

Recenzent: dr Danuta Szeliga

Podpis dyplomanta: Podpis promotora:

Kraków 2013

***(Oświadczenia zgodnie z Regulaminu studiów w AGH):***

***Oświadczam, świadomy(-a) odpowiedzialności karnej za poświadczenie nieprawdy, że niniejszą pracę dyplomową wykonałem(-am) osobiście i samodzielnie i że nie korzystałem(-am) ze źródeł innych niż wymienione w pracy.***

Kraków, dnia …… Podpis dyplomanta…………….

**Spis treści**

[1 Wstęp 4](#_Toc391042686)

[2 Analiza problemu 4](#_Toc391042687)

[2.1 Kontekst środowiska mobilnego 4](#_Toc391042688)

[3 Systemy 6](#_Toc391042689)

[3.1 Windows 6](#_Toc391042690)

[3.2 Linux 6](#_Toc391042691)

[3.3 iOS 6](#_Toc391042692)

[3.3.1 Wielozadaniowość w iOS 7](#_Toc391042693)

[3.4 Tworzenie aplikacji na system iOS 8](#_Toc391042694)

[4 Środowiska międzyplatformowe 9](#_Toc391042695)

[4.1 Unity3D 9](#_Toc391042696)

[4.2 Web/HTML5/Flash 9](#_Toc391042697)

[4.3 9](#_Toc391042698)

[5 Zaimplementowane programy testujące 9](#_Toc391042699)

[6 Wyniki 9](#_Toc391042700)

[7 Podsumowanie i wnioski 9](#_Toc391042701)

[8 Literatura 9](#_Toc391042702)

# Wstęp

# Analiza problemu

W obecnych czasach duża liczba architektur mobilnych oraz różnorodność systemów operacyjnych tychże zaczynają sprawiać problemy osobom tworzącym oprogramowanie. Smartfony posiadają coraz więcej możliwości programowych i sprzętowych, takie jak ekrany wielodotykowe, modemy 4G, WiFi i przede wszystkim możliwości instalowania oprogramowania dostarczonego przez osoby trzecie. Dzięki tym aplikacjom użytkownicy dostali do dyspozycji sporo nowych funkcjonalności, w szczególności wspierających mobilność konsumenta. Dzięki tym aplikacjom przykładowo użytkownik może śledzić swoją pozycję w nieznanym mu mieście, wyświetlać informację w czasie rzeczywistym (rzeczywistość rozszerzona) czy też zapłacić za zakupy w sklepie lub bilet w komunikacji miejskiej. [2] . XXX

Jednak w parze z udogodnieniami dla użytkownika poszły problemy dla twórców tych aplikacji. W świecie mobilnym można wyróżnić cztery podstawowe rodzaje problemów. Pierwszym z nich jest zapewnienie użytkownikowi tego samego doświadczenia z aplikacją niezależnie od platformy, z której korzysta. Interfejs aplikacji powinien być zbliżony pomiędzy platformami i intuicyjny. Drugim problemem jest zużycie zasobów i pobór energii. Pomimo, iż nowoczesne telefony posiadają znacznie więcej mocy niż kiedyś, dalej nie dorównują komputerom klasy PC pod względem dostępnej pamięci czy prędkości procesora. Trzecim z problemów jest utrzymywanie aplikacji. Mobilne systemy operacyjne są często uaktualniane, nierzadko bez wsparcia dla kompatybilności wstecznej. Aplikacje działające na starszych wersjach danego systemu nie muszą działać na nowszych. W jaki więc sposób utrzymywać i testować nowe wersje programu? Ostatnim z problemów jest duża różnorodność sprzętowa oraz systemów operacyjnych pomiędzy producentami współczesnych smartfonów. Każdy z nich korzysta z diametralnie innych technik tworzenia oprogramowania, języków i praktyk.

## Kontekst środowiska mobilnego

Rynek smartfonów dla naszych rozważań możemy uznać za podzielony na dwie grupy: tworzących urządzenia i dostarczających systemy operacyjne. Usługodawcy telefoniczni nie mają znaczenia dla naszych rozmyślań.

Grupa odpowiedzialna za tworzenie urządzeń buduje i komponuje każde urządzenie z konkretnych podzespołów takich jak WiFi, modem 4G, GPS czy akcelerometr, a także konfiguruje telefon i przystosowuje go do korzystania z jednego systemu operacyjnego. Oczywistym jest, że każda z tych firm ma swoje ustalone standardy, zasady i sposoby produkcji urządzeń. Grupa dostarczających systemy operacyjne odpowiedzialna jest za stworzenie połącznia pomiędzy użytkownikiem a leżącym pod spodem sprzętem. Na rynek ten składają się np. Apple z iOS, Open Handset Alliance z Androidem, Microsoft z Windows Phone, RIM z BlackBerry, Samsung z Bada czy też Nokia z Symbianem. Aktualnie dwoma z najpopularniejszych systemów mobilnych są iOS i Android. Firmy te tworzą systemy operacyjne niekompatybilne ze sobą. Niekompatybilność systemu dotyka przeciętnego tworzącego oprogramowanie bardziej niż niekompatybilność sprzętowa, jako że deweloper i tak nie posiada bezpośredniego dostępu do warstwy sprzętowej telefonu, a korzysta z API wystawionego przez tworzących system operacyjny. Poniżej przedstawiono kilka różnic pomiędzy najpopularniejszymi mobilnymi systemami operacyjnymi:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| System operacyjny | Maszyna wirtualna | Język programowania | Interfejs użytkownika | Zarzadzanie pamięcią | IDE | Platformy programistyczne | urządzenia |
| iOS | Brak | Objective-C | Cocoa touch | Liczenie referencji | XCode | Mac Os X | Homogeniczne |
| Android | Dalvik VM | Java | Pliki XML | Garbage Collector | Eclipse | Wiele platform | heterogeniczne |
| Windows Phone 7 | CLR | C# + .NET | Pliki XAML | Garbage Collector | Visual Studio | Windows Vista / 7 | Homogeniczne |
| BlackBerry OS | Java ME | Java | W kodzie | Garbage Collector | Eclipse | Wiele platform | Heterogeniczne |
| Symbian OS | dostępna | C++ | Qt | Ręczne | Qt Creator | Wiele platform | heterogeniczne |

# Systemy

## Windows

## Linux

## iOS



System iOS (wcześniej iPhone OS) to system operacyjny firmy Apple Inc. Dystrybuowany z produktami takimi jak iPhone, iPad, iPod Touch, i Apple TV. Po raz pierwszy zaprezentowany w 2007 roku na urządzeniach typu iPhone został rozwinięty i wzbogacony o wsparcie dla urządzeń iPod Touch (wrzesień 2007), iPad (styczeń 2010), iPad Mini (listopad 2012) i dla telewizji drugiej generacji Apple TV (wrzesień 2010). W październiku 2013 roku App Store firmy Apple posiadał ponad milion aplikacji dla iOS, z czego około połowa zoptymalizowana była dla iPada. Aplikacje te zostały ściągnięte ok. 60 miliardów razy. W czwartym kwartale 2012 21% wszystkich sprzedanych urządzeń mobilnych posiadały zainstalowany system iOs. W połowie roku 2012 na świecie znajdowało się około 410 milionów aktywnych urządzeń korzystających z systemu.

Interfejs użytkownika oparty jest o ideę manipulacji bezpośredniej i korzystanie z gestów wielopunktowych. Najczęstszymi sposobami na komunikację z systemem są suwaki, przyciski oraz toglery. System wspiera też rozmaite gesty takie jak ściśnięcie, ściśnięcie odwrotne, dotknięcie i przesunięcie. Wszystkie gesty mają swoje określone definicje w kontekście systemu iOS i jego interfejsu wielo dotykowego. Wbudowane akcelerometry wykorzystywane są w niektórych aplikacjach w odpowiedzi na np. potrząsanie urządzeniem (co często sygnalizuje komendę „cofnij”) lub w celu rozpoznania orientacji urządzenia w przestrzeni trójwymiarowej (np. dla przełączania się pomiędzy sposobami wyświetlania obrazu).

System ma pewne elementy wspólne z systemem OS X takie jak Core Foundation i Foundation, jednak jego interfejsem zajmuje się moduł Cocoa Touch, gdzie w OS X wykorzystywany jest Cocoa. Z powodu tych różnic aplikacje na system OS X nie są kompatybilne z systemem iOS. Dodatkowo, mimo że iOS dzieli z OS X podstawowe cechy systemu Darwin, dostęp do terminala systemowego jest niemożliwy dla użytkownika, przez co system nie jest też w pełni kompatybilny z systemami Unix.

Nowe wersje systemu dystrybuowane są co roku. Najnowsza, iOS 7, została przekazana użytkownikom 18 września 2013 roku. W systemie istnieją cztery abstrakcyjne płaszczyzny: Core OS, Core Services, Media oraz Cocoa Touch.

### Wielozadaniowość w iOS

Wielozadaniowość w iOS miała swój debiut w czerwcu 2010 roku wraz z premierą iOS 4.0. Tylko niektóre urządzenia Apple – iPhone 4, iPhone 3GS i iPod Touch trzeciej generacji – potrafiły korzystać z wielozadaniowości. Sposób implementacji wielozadaniowości w systemie iOS był wielokrotnie krytykowany, za to że aplikacje pracujące w tle zmuszone są korzystać z ograniczonego zasobu funkcji i za wymaganie od deweloperów tworzenia jawnego wsparcia dla tych funkcjonalności w aplikacjach.

Przed premierą systemu iOS 4 wielozadaniowość ograniczona była do wybranych aplikacji instalowanych przez Apple na urządzeniu. Użytkownicy mogli jednak „uwolnić” system („jailbreak”) i uzyskać częściowe, nieoficjalne wsparcie dla wielozadaniowości.

Wraz z iOS 4 wielozadaniowość wspierana jest przez siedem interfejsów (API) działających w tle:

* Audio w tle – aplikacja pozostaje w działaniu w tle dopóki nie zakończy odtwarzania plików multimedialnych
* Voice over IP – aplikacja jest zawieszona jeżeli nie jest aktywne połączenie telefoniczne
* Geolokalizacja w tle – aplikacja informowana jest o zmianie położenia urządzenia
* Powiadomienia push (Push notifications)
* Powiadomienia lokalne – aplikacja planuje powiadomienia, które dostarczane mają być o wybranej porze
* Task completion – aplikacja komunikuje się z systemem operacyjnym w celu ustalenia dodatkowego czasu procesora na zakończenie trwających zadań
* Szybkie zmiany aplikacji – aplikacja nie uruchamia żadnego kodu i może zostać usunięta z pamięci w dowolnym momencie

W systemie iOS 5 zadebiutowały trzy dodatkowe interfejsy:

* Newsstand – aplikacja może pobierać treści w tle tak by były gotowe dla użytkownika w późniejszym czasie
* External accessory – aplikacja komunikuje się i dzieli danymi z urządzeniami zewnętrznymi w ustalonych odstępach czasu
* Bluetooth accessory – aplikacja komunikuje się i dzieli danymi z urządzeniami klasy Bluetooth w ustalonych odstępach czasu

W systemie iOS7 Apple uruchomiło dodatkową funkcjonalność która pozwala wszystkim aplikacjom na dokonywanie aktualizacji w tle. Ta funkcjonalność preferuje aktualizowanie aplikacji z których użytkownik najczęściej korzysta i wszystkie aktualizacje stara się przeprowadzać gdy dostępna jest sieć WiFi, by nie korzystać z przesyłania danych przez moduł telefonii komórkowej.

### Tworzenie i dystrybucja aplikacji na system iOS

Aplikacje muszą być napisane i skompilowane z myślą o systemie iOS i 64 bitowej architekturze lub poprzedniej 32 bitowej. Przeglądarka Safari obsługuje aplikacje sieciowe. Istnieją autoryzowane aplikacje firm trzecich korzystające z kodu natywnego dla urządzeń korzystających z iOS 2.0 i późniejszych.

17 października 2007 roku w liście otwartym opublikowanym nam blogu firmy Apple „Hot news” Steve Jobs ogłosił wydanie pakietu deweloperskiego (SDK) w okolicach lutego 2008 roku[przypissss]. SDK zostało wydane 6 marca 2008 i pozawalało na tworzenie aplikacji dla iPhone, iPod Touch oraz na uruchamianie i testowanie aplikacji w emulatorze. Jednak załadowanie aplikacji na rzeczywiste urządzenie możliwe było tylko (jest nadal?) po opłaceniu składki członkowskiej programu deweloperskiego Apple.

Składki za przynależność do danego programu zostały zatwierdzone na 99 dolarów amerykańskich za każdy z dostępnych programów członkowskich: iOS i OSX. W lipcu 2011 roku Apple rozpowszechniło Xcode w swoim sklepie internetowym do pobrania bez opłat dla wszystkich użytkowników OS X Lion. Od swojego rozpowszechnienia Xcode 3.1 stał się podstawowym środowiskiem dla iOS SDK.

Deweloperzy mogą wybrać dowolną cenę za swoje aplikacje powyżej ustalonego w umowie minimum. Po sprzedaży aplikacji w App Store deweloper zachowuje 70% zysku a pozostałe 30% pozostaje dla Apple. Alternatywnie deweloperowi dana jest możliwość stworzenia aplikacji darmowej – w tym przypadku nie ponosi żadnych kosztów za rozpowszechnianie aplikacji poza wymienioną wcześniej opłatą za przynależność do programu deweloperskiego Apple.

### Dostępność środowisk w systemie iOS

Java, flash, html itp.

## Android



Android jest systemem operacyjnym na urządzenia mobilne opartym o kernel Linuksa rozwijanym przez firmę Google. Interfejs użytkownika oparty jest o manipulację bezpośrednią zaprojektowany jest z myślą o urządzeniach z wbudowanym interfejsem dotykowym takimi jak smartfony i tablety, ale obsługuje też wyspecjalizowane interfejsy telewizyjne (Android TV), zegarki (Android Wear) a nawet samochody (Android Auto). System korzysta z bazy gestów takich jak przewijanie, ściskanie czy też dotknięcie by manipulować obiektami na ekranie a także do obsługi wirtualnej klawiatury.

Od roku 2011 Android posiada największą bazę zainstalowanych aplikacji, a od roku 2013 urządzenia z tym systemem zostały zakupione więcej razy niż urządzenia z systemami Windows Phone, iOS i Mac OS razem wzięte. Od lipca 2013 Google Play (oficjalny sklep z aplikacjami firmy Google) posiadał w swojej bazie ponad milion opublikowanych aplikacji Android i ponad 50 miliardów pobrań tychże. Ankieta wśród deweloperów sporządzona na przełomie kwietnia i maja 2013 roku wykazała, iż 71% deweloperów aplikacji mobilnych tworzy aplikacje na Androida. Na konferencji Google I/O 2014 firma ujawniła, że liczba aktywnych użytkowników (aktywność z ostatnich 30 dni) wzrosła do miliarda z 538 milionów w czerwcu 2013 roku.

Kod źródłowy Androida rozpowszechniany przez Google jest oparty o licencje open source, choć większość urządzeń sprzedawana jest z mieszanką oprogramowania otwartego i komercyjnego. Początkowo rozwijany przez firmę Android, Inc. wspieraną finansowo, a następnie wykupioną przez Google Android zadebiutował na rynku w roku 2007 wraz z założeniem Open Handset Alliance – organizacji dedykowanej rozwijaniu standardów otwartych dla urządzeń mobilnych.

Android jest bardzo popularny wśród firm technologicznych wymagających gotowego, taniego i modularnego systemu operacyjnego dla urządzeń o wysokim skomplikowaniu konstrukcyjnym. Otwarta specyfikacja Androida zaskutkowała sporą społecznością programistów i entuzjastów gotowych na rozwijanie otwartych inicjatyw poprzez dodawanie nowych funkcjonalności do istniejących programów, a nawet do przystosowywania urządzeń, które początkowo korzystały z innych systemów operacyjnych, do Androida.

Android jest rozwijany przez firmę Google do momentu, w którym najnowsze zmiany i aktualizacje są gotowe do rozpowszechnienia, po czym kod źródłowy najnowszej wersji zostaje upubliczniony. Ta wersja kodu bez modyfikacji może zostać uruchomiona tylko na wybranych urządzeniach, zazwyczaj z serii Nexus. Następnie kod źródłowy przechodzi etap adaptacji przez firmy wytwarzające urządzenia. Kod źródłowy Androida nie zawiera komercyjnych sterowników potrzebnych do obsługi niektórych komponentów sprzętowych urządzenia.

### Częstotliwość aktualizacji

Google rozpowszechnia duże, inkrementalne aktualizacje dla Androida co 6-9 miesięcy. Większość aktualizacji możliwa jest do zainstalowania przez bezprzewodowe interfejsy sieciowe. Aktualnie najnowszą wersją systemu jest 4.4 „KitKat”.

W porównaniu do swojego największego rywala – iOS – aktualizacje systemu operacyjnego zazwyczaj docierają do urządzeń końcowych z dużym opóźnieniem. Dla urządzeń które nie należą do serii Nexus aktualizacje często docierają z kilkumiesięcznym opóźnieniem. Problem ten po części spowodowany jest dużym zróżnicowaniem sprzętowym urządzeń z systemem Android, gdzie każda aktualizacja musi być dostosowywana do konkretnego modelu urządzenia. Portowanie aktualizacji jest czasochłonne i wymaga dużych zasobów, dlatego producenci często nadają priorytet nowszym urządzeniom i często pomijają starsze. Z tego powodu starsze urządzenia bardzo często nie otrzymują niektórych aktualizacji jeżeli producent dojdzie do wniosku, że nie warto poświęcać czasu i zasobów na dostosowanie kodu, niezależnie od tego czy starsze urządzenie byłoby w stanie obsługiwać nowsze aktualizacje. Problem ten staje się jeszcze większą przeszkodą, gdy producenci indywidualizują swoje wydanie Androida poprzez dodawanie aplikacji i interfejsów, gdyż te też muszą być dostosowane do każdego nowego wydania. Dodatkowe opóźnienia często wprowadzane są też przez operatorów sieci komórkowych, którzy po otrzymaniu aktualizacji od producenta rozpoczynają swoje własne indywidualizacje a następnie długotrwale testują wewnątrz swojej sieci przed wypuszczeniem poprawki do użytkowników końcowych.

In 2012, Google began decoupling certain aspects of the operating system (particularly core applications) so they could be updated through Google Play Store, independently of Android itself. One of these components, Google Play Services, is a closed-source system-level process providing APIs for Google services, installed automatically on nearly all devices running Android version 2.2 and higher. With these changes, Google can add new operating system functionality through Play Services and application updates without having to distribute an upgrade to the operating system itself. As a result, Android 4.2 and 4.3 contained relatively fewer user-facing changes, focusing more on minor changes and platform improvements.[3][97]

Linux kernel

Android consists of a kernel based on the Linux kernel long-term support (LTS) branch. As of January 2014, current Android versions are built upon Linux kernel 3.4 or newer,[98][99] but the specific kernel version number depends on the actual Android device and chipset.[100][101][102] Android has used various kernels since its first 2.6.25.[40]

Android's Linux kernel has further architectural changes that are implemented by Google outside the typical Linux kernel development cycle, such as the inclusion of components like Binder, ashmem, pmem, logger, wakelocks, and different out-of-memory (OOM) handling.[103][104][105] Certain features that Google contributed back to the Linux kernel, notably a power management feature called "wakelocks", were rejected by mainline kernel developers partly because they felt that Google did not show any intent to maintain its own code.[106][107][108] Google announced in April 2010 that they would hire two employees to work with the Linux kernel community,[109] but Greg Kroah-Hartman, the current Linux kernel maintainer for the stable branch, said in December 2010 that he was concerned that Google was no longer trying to get their code changes included in mainstream Linux.[107] Some Google Android developers hinted that "the Android team was getting fed up with the process," because they were a small team and had more urgent work to do on Android.[110]

In August 2011, Linus Torvalds said that "eventually Android and Linux would come back to a common kernel, but it will probably not be for four to five years".[111] In December 2011, Greg Kroah-Hartman announced the start of Android Mainlining Project, which aims to put some Android drivers, patches and features back into the Linux kernel, starting in Linux 3.3.[112] Linux included the autosleep and wakelocks capabilities in the 3.5 kernel, after many previous attempts at merger. The interfaces are the same but the upstream Linux implementation allows for two different suspend modes: to memory (the traditional suspend that Android uses), and to disk (hibernate, as it is known on the desktop).[113] Google maintains a public code repository that contains their experimental work to re-base Android off the latest stable Linux versions.[114][115]

The flash storage on Android devices is split into several partitions, such as /system for the operating system itself, and /data for user data and application installations.[116] In contrast to desktop Linux distributions, Android device owners are not given root access to the operating system and sensitive partitions such as /system are read-only. However, root access can be obtained by exploiting security flaws in Android, which is used frequently by the open-source community to enhance the capabilities of their devices,[117] but also by malicious parties to install viruses and malware.[118]

Android is a Linux distribution according to the Linux Foundation,[119] Google's open-source chief Chris DiBona,[120] and several journalists.[121][122] Others, such as Google engineer Patrick Brady, say that Android is not Linux in the traditional Unix-like Linux distribution sense; Android does not include the GNU C Library and some of other components typically found in Linux distributions.[123]

### Tworzenie i dystrybucja aplikacji Android

# Środowiska międzyplatformowe

## Unity3D

## Web/HTML5/Flash

## 

# Zaimplementowane programy testujące

# Wyniki

# Podsumowanie i wnioski

# Literatura