Masarykova univerzita
Fakulta informatiky

# Vývoj komunitní mobilní aplikace z prostředí gastronomie pro platformu Google Android

Bakalářská práce

**Marek Osvald**

Brno, jaro 2015

# Prohlášení

Prohlašuji, že tato bakalářská práce je mým původním autorským dílem, které jsem vypracoval samostatně. Všechny zdroje, prameny a literaturu, které jsem při vypracování používal nebo z nich čerpal, v práci řádně cituji s uvedením úplného odkazu na příslušný zdroj.

Marek Osvald

**Vedoucí práce:** Mgr. Luděk Bártek, Ph.D.

# Poděkování

Rád bych na tomto místě poděkoval svým rodičům, kteří mi umožnili vystudovat, panu doktoru Bártkovi za vedení této práce a vyučujícím Fakulty informatiky za předané znalosti a vědomosti.

# Shrnutí

Cílem této práce je zanalyzovat vývoj komunitních aplikací pro mobilní zařízení, zejména pak pro platformu Android. V práci si dále kladu za cíl vytvořit komunitní aplikaci pro evidenci provozoven rychlého občerstvení a zodpovídání dotazů nad těmito daty. V práci dále popisuji kompletní cyklus vývoje mobilní komunitní aplikace od fáze návrhu až po úspěšné nasazení do Google Play, tržiště aplikací pro platformu Android.

# Klíčová slova

Android, Java, mobilní aplikace, Eclipse

# Obsah


1 **Úvod** . . . 3
2 **Mobilní zařízení** . . . 5
2.1 *Specifika vývoje mobilních aplikací* . . . 5
2.1.1 Displej . . . 6
2.1.2 Ovládání . . . 6
2.1.3 Napájení . . . 7
2.2 *Moderní mobilní platformy* . . . 8
2.2.1 Android . . . 8
2.2.2 Bada . . . 9
2.2.3 BlackBerry OS . . . 10
2.2.4 iOS . . . 10
2.2.5 Symbian . . . 11
2.2.6 Windows Mobile a Windows Phone . . . 11
3 **Platforma Android** . . . 13
3.1 *Historie platformy Android* . . . 13
3.2 *Hardwarová a softwarová standardizace* . . . 14
3.3 *Architektura platformy Android* . . . 16
3.3.1 Aplikace . . . 16
3.3.2 Aplikační rámec . . . 17
3.3.3 Knihovny . . . 18
3.3.4 Android Runtime . . . 19
3.3.5 Linuxové jádro . . . 19
3.4 *Nástroje pro vývoj aplikací* . . . 20
3.4.1 Nástroje pro vývojáře . . . 20
3.4.2 Android Studio . . . 21
3.4.3 Android Development Tools pro Eclipse . . . 21
3.4.4 Emulátor . . . 21
3.5 *Distribuce aplikací pro platformu Android* . . . 22
3.5.1 Google Play . . . 22
3.5.2 Amazon App Store . . . 23
3.5.3 Mimotržištní distribuce . . . 23
4 **Vývoj aplikací pro platformu Android** . . . 26
4.1 *Možnosti vývoje aplikací pro platformu Android* . . . 26
4.1.1 Android SDK a Android NDK . . . 26
4.1.2 Webové aplikace . . . 26

4.1.3 Alternativní vývoj . . . . . 26
4.2 *Vývoj aplikací pomocí Android SDK* . . . . . 27
4.2.1 Komponenty aplikace pro Android . . . . . 27
4.2.2 Fragmenty (Fragments) . . . . . 29
4.2.3 Manifest . . . . . 29
4.2.4 Záměry (Intents) . . . . . 30
4.2.5 Serializace . . . . . 31
4.2.6 Práce se zdroji . . . . . 31
5 **Aplikace Hladová okna** . . . . . 33
5.0.1 Průzkum mezi potenciálními uživateli . . . . . 33
5.0.2 Doména . . . . . 33
5.0.3 Komunitní aplikace . . . . . 34
5.1 *Diagramy* . . . . . 35
5.1.1 Diagram případů užití . . . . . 35
5.1.2 Konceptuální ERD diagram . . . . . 40
5.1.3 Fyzický ERD diagram (server) . . . . . 41
5.1.4 Fyzický ERD diagram (klient) . . . . . 41
5.2 *Klient* . . . . . 45
5.2.1 Komunikační rozhraní . . . . . 45
5.2.2 Databáze a ORMLite . . . . . 45
5.2.3 Určení lokace a práce s mapou . . . . . 46
5.2.4 Zpracování velkých bitmap . . . . . 48
5.2.5 Použité knihovny . . . . . 50
5.3 *Server* . . . . . 52
5.3.1 Použité knihovny . . . . . 52
5.3.2 Nasazení na platformě OpenShift . . . . . 53
6 **Nasazení** . . . . . 56
7 **Závěr** . . . . . 58
A **Přílohy** . . . . . 67

# 1 Úvod

Telefonie je z hlediska počtu uživatelů nejúspěšnější komunikační technologií na světě. Obzvláště úspěšná je pak mobilní telefonie, která si za několik málo desetiletí získala několik miliard uživatelů jak v rozvojových, tak rozvinutých zemích. Počet aktivních mobilních telefonů převyšuje jak počet počet osob s přístupem k Internetu, tak počet domácností s osobním počítačem.

Lze také pozorovat jasný trend v klesání počtu pevných telefonních přípojek a naopak trvalého růstu počtu aktivních mobilních telefonů. V letech 2001-2007 stále počet aktivních pevných telefonních přípojek stoupal, nicméně od roku 2007 můžeme pozorovat znatelný pokles počtu přípojek a i výhled Mezinárodní telekomunikační unie (ITU) je klesající. Pro rok 2013 odhadla ITU pokles na 16,5/100 obyvatel, což je dokonce pod úrovní roku 2001 [42].

Naopak počet mobilních telefonů v letech 2001-2007 vzrostl. V absolutních číslech ze zvýšil počet aktivních mobilních telefonů na světě z 962 milionů v roce 2001 na celkových 5,9 miliardy v roce 2011. Pro rok 2013 ITU odhadla 6,8 miliardy aktivních mobilních telefonů, což odpovídá 98,5/100 obyvatel [41]. Dle ITU se tak za posledních 12 let efektivně smazal rozdíl mezi rozvinutými a rozvojovými zeměmi a dá se očekávat, že výhledově brzy počet aktivních mobilních telefonů překoná světovou populaci [43].

Pro informatiku je rozvoj mobilní telefonie důležitý hlavně s nástupem tzv. chytrých mobilních telefonů (z angl. *smart phone* nebo též *smartphone*). Ačkoli termín smartphone není pevně definovaný, můžeme zařízení tohoto typu chápat jako mobilní telefon, který poskytuje uživateli přístup ke konektivitě a výpočetnímu výkonu tohoto zařízení. Pro dnešní chytré telefony je typické rozšiřování jejich funkcionality prostřednictvím nahrávání dalších aplikací.

Tento jev můžeme nazvat jako tzv. Post-PC nebo PC+ éru [8], kdy nelze již považovat osobní počítače typu PC ani za jediné, ale ani za výrazně dominující zařízení, ze kterého budou uživatelé k aplikacím přistupovat. Obecně tak můžeme pozorovat trend, kdy se distribuce a vnitřní architektura aplikací vrací k centralizovanému modelu, kdy data a většina aplikační logiky probíhá vzdáleně na serveru, zatímco uživatelovo zařízení funguje jako tzv. tenký klient [50].

Nová platforma s sebou logicky přináší i nové nároky vyplývající z ergonomických a výpočetních omezení. Zároveň přináší i nové možnosti, které vycházejí z pohodlné přenositelnosti těchto zařízení a hardwarových komponent, které jsou unikátní pro mobilní zařízení, jako jsou například čip NFC, vestavěný fotoaparát nebo modul GPS.

Jedním z významných aspektů, které mobilní zařízení značně ovlivnila, je i distribuce aplikací. Ačkoli je to v některých případech možné, není zvykem instalovat nové aplikace z instalačního média. Namísto toho jsou na mobilních platformách aplikace obvykle distribuovány pomocí systému aplikačních tržišť, díky kterým lze aplikace nejen jednoduše stahovat a instalovat z Internetu, ale i automaticky aktualizovat.

# 2 Mobilní zařízení

Historicky poskytovaly mobilní zařízení nižší výpočetní výkon než počítače typu PC. To v současné době již neplatí. Moderní mobilní zařízení disponují několikajádrovými procesory a operační pamětí v řádech jednotek GB. Jako omezující podmínku nelze chápat ani sekundární paměť, protože moderní telefony a tablety disponují úložišti o velkosti několika desítek GB.

Mobilní zařízení jsou ale zpravidla připojena k méně stabilním a pomalejším připojením k Internetu než osobní počítače.

Jiný je i způsob využití aplikací na mobilních zařízení. Podle americko-německé studie Matthiase Böhmera a kolektivu provedené v roce 2011 byla průměrná délka používání mobilních zařízení 53,23 minut denně. Průměrná délka využití jednotlivé aplikace (od okamžiku spuštění až po ukončení) pak činila pouhých 71,56 vteřin [7].

Této velice rychlé a krátké interakci se také přizpůsobuje způsob interakce, kterou mobilní zařízení nabízejí – například minimem potvrzovacích dialogů a okamžitým projevováním změn na základě uživatelových pokynů.

## 2.1 Specifika vývoje mobilních aplikací

Mobilní aplikace se v mnohém podobají klasickým desktopovým aplikacím. Jedná se o aplikace s grafickým uživatelským rozhraním, využívající standardní ovládací prvky definované v rámci konkrétní mobilní platformy. V mnohém tak lze využít postupy softwarového inženýrství a vývojové techniky známé z vývoje grafických aplikací pro osobní počítače.

Oproti desktopovým aplikacím však mají několik specifik. Mobilní zařízení bývají navíc oproti PC mnohem více heterogenní, co se technických parametrů týče, což může znesnadňovat vývoj a ladění aplikací.

### 2.1.1 Displej

Tablety a mobilní telefony mají oproti počítačům typu PC menší displej (obvykle v rozmezí od 3 do 5 palců u telefonů a od 6 do 10 palců u tabletů). Velikost mobilních telefonů a tabletů ovlivňuje způsob interakce s těmito zařízeními (viz obrázky 2.1 a 2.3). Historicky měly chytré mobilní telefony i nižší rozlišení obrazovky, nicméně to v současné době již zcela neplatí. Zařízení s Androidem mohou mít rozlišení od QVGA (320 $\times$ 240 pixelů) až po tzv. QHD rozlišení (2564 $\times$ 1440 pixelů).

Při návrhu uživatelského rozhraní toto musí být v bráno v potaz. Ovládací prvky, které jsou na obrazovce umístěny, bývají obvykle větší než ovládací prvky na osobních počítačích z důvodu lepší čitelnosti a přehlednosti uživatelského rozhraní. U dotykových obrazovek platí, že vizuální prvky musí být dostatečně velké, aby s nimi bylo možno snadno interagovat dotykem prstu.

### 2.1.2 Ovládání

Oproti počítačům typu PC, u kterých je standardní ovládání pomocí klávesnice a myši a u kterých se dá přítomnost těchto vstupních zařízení předpokládat, ovládání mobilních zařízení není nijak standardizováno.

Mobilní telefony obsahují určitou formu klávesnice a to buď hardwarovou (QWERTY[1] či dvanáctitlačítkovou[2]) nebo softwarovou, která je typická pro dotykové displeje a která fyzickou klávesnici emuluje. Kromě klávesnice mohou chytré telefony obsahovat i dodatečná tlačítka např. pro ovládání hlasitosti, navigaci nebo vyhledávání.

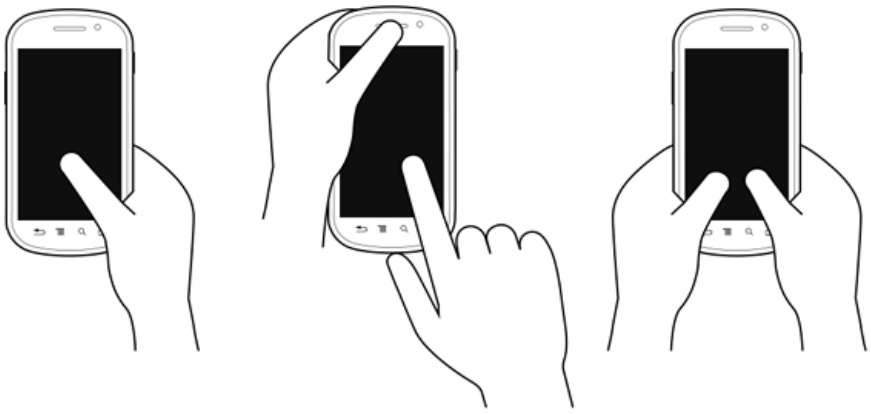
Pro výběr ovládacích prvků slouží buďto dotykový displej ovládaný prstem či stylem, případně další zařízení jako je například trackball nebo joystick. Existují i taková mobilní zařízení, která výše uvedené přístupy různě kombinují. V současné době jsou nejhojněji rozšířena zařízení, která jsou ovládána dotykovým displejem a softwarovou klávesnicí.

---

1. obdobu klávesnice psacího stroje firmy Remington & Sons [40]
2. klávesnice obsahující dvanáct tlačítek: 10 pro číslice a 2 pro symboly * a #, která je známá z pevných a starších mobilních telefonů

Obecně platí, že přesnost těchto ovládacích zařízení je podstatně nižší. Při ovládání mobilních zařízení tak dochází velice často k uživatelským chybám a při vývoji je na to nutno pamatovat. Některé chyby interakce, jako jsou např. překlepy bývají opravovány již na úrovni operačního systému. U ostatních platí podobné zásady jako u vývoje desktopových aplikací – např. při provádění destruktivní operace by uživatel měl svou volbu stvrdit prostřednictvím potvrzovacího dialogu, apod.

Naopak rozmístění vizuálních ovládacích prvků by se mělo přizpůsobit těmto zařízením vzhledem k ergonomickým omezením (viz obrázky 2.2 a 2.4).



Obrázek 2.1: Typické příklady interakce s chytrým telefonem
Zdroj: `http://www.lukew.com/ff/entry.asp?1649`

### 2.1.3 Napájení

V současné době je největším technologickým omezením mobilních zařízení kapacita akumulátoru [5]. Díky tomu mobilní zařízení vyžadují mnohem vyšší úroveň optimalizace, než desktopové nebo serverové aplikace [48], protože každá instrukce spotřebovává elektrickou energii.

Zároveň by aplikace, bez ohledu na platformu, měla dodržovat jakýsi *koncept slušnosti* [46], což znamená využívat pouze ty zdroje,

Obrázek 2.2: Dosažitelnost ovládacích prvků u chytrého telefonu
Zdroj: `http://www.lukew.com/ff/entry.asp?1649`

které aplikace v daný moment vyžaduje a to pouze po nezbytně nutnou dobu.

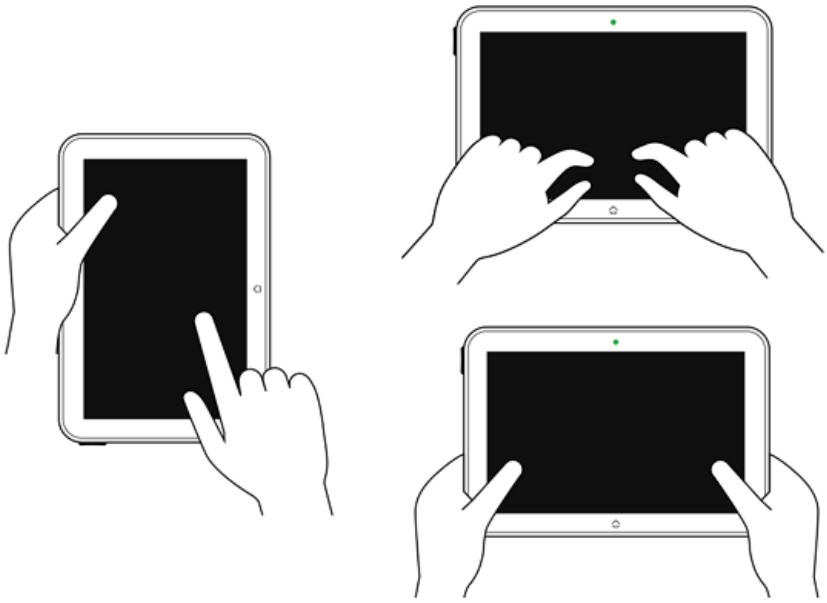
## 2.2 Moderní mobilní platformy

Preference uživatelů mobilních platforem a rozšíření těchto platforem se v čase významně mění (jak dokládá graf 2.5).

Zatímco na začátku roku 2007 byl naprosto dominantním operační systém Symbian s 61,2% podílem na trhu, dnes se jedná o velice okrajovou platformu. Naopak operační systémy Android a iOS, které byly představeny teprve před několika lety si dokázaly získat na trhu za dobu své existence celkově více než tříčtvrtinový podíl.

### 2.2.1 Android

Android je platforma vyvinutá společností Google, distribuovaná jako open-source pod licencí Apache Licence 2.0. Celkem bylo aktivováno více než 224 milionů zařízení s Androidem [9]. Androidu je podrobněji věnována kapitola 3.



Obrázek 2.3: Typické příklady interakce s počítačovým tabletem
Zdroj: http://www.lukew.com/ff/entry.asp?1649

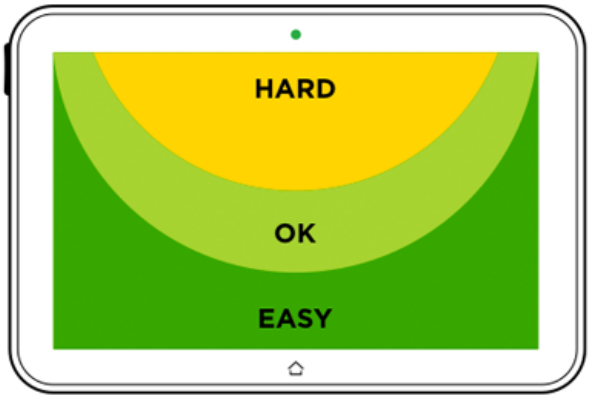
### 2.2.2 Bada

Bada je mobilní platforma vyvinutá společností Samsung Electronics pro mobilní telefony a počítačové tablety značky Samsung. První verze Bady byla představena v roce 2010. Zdrojový kód Bady je v současné době částečně otevřený [55]. Bada je napsána pomocí programovacích jazyků C, C++ a Java. Součástí Bady je i technologie Adobe Flash [4]. Vývoj Bady byl únoru roku 2013 ukončen.

Aplikace pro Badu jsou distribuovány pomocí tržiště aplikací Samsung apps[3].

Na trhu se vyskytuje necelých 10 milionů mobilních telefonů s operačním systémem Bada [6]. Současný podíl Bady na trhu je odhadován na 0,3 % [31].

3. http://www.samsungapps.com/




Obrázek 2.4: Dosažitelnost ovládacích prvků u počítačového tabletu
Zdroj: `http://www.lukew.com/ff/entry.asp?1649`

### 2.2.3 BlackBerry OS

BlackBerry OS je proprietární operační systém společnosti Research In Motion pro mobilní telefony značky BlackBerry. První verze BlackBerry OS byla vydána v roce 1999. Zdrojový kód BlackBerry OS je naprogramován v jazyce C++ a není veřejně přístupný.

Aplikace pro BlackBerry OS jsou psány v jazyce Java (Micro Edition). Od verze 4.5 je součástí systému i tržiště aplikací BlackBerry AppWorld[4], na kterém je k dispozici více než 25 000 aplikací [49].

Od verze 10.3 umožňuje BlackBerry OS spouštět také aplikace určené pro Android.

### 2.2.4 iOS

iOS (původně nazvaný iPhone OS) je proprietární operační systém společnosti Apple pro mobilní telefony iPhone, tablety iPad a hudební přehrávače iPod Touch. První veřejná verze byla vydána v roce 2007. iOS je naprogramován v jazycích C, C++ a Objective-C. Zdrojové kódy nejsou veřejně přístupné.

Aplikace pro iOS jsou distribuovány pomocí tržiště aplikací AppStore[5].

4. `http://appworld.blackberry.com`
5. `http://www.apple.com/iphone/from-the-app-store/`

### 2.2.5 Symbian

Symbian je operační systém pro mobilní telefony původně vyvíjený společností Symbian, Ltd. pro mobilní telefony značky Nokia. Nokia skoupila společnost Symbian v roce 2008 a přesunula vývoj tohoto operačního systému přímo pod sebe. Symbian je napsán v jazyce C++.

V roce 2011 Nokia kontraktovala další vývoj Symbianu firmě Accenture, která se zavázala udržovat Symbian nejméně do roku 2016.

Nokia provozovala vlastní tržiště aplikací jménem Ovi Store, které bylo později přejmenováno na Nokia Store. Nicméně od roku 2014 již nemohli vývojáři přidávat nové aplikace ani aktualizace stávajících a jeho provoz byl ukončen 18. února 2015.

### 2.2.6 Windows Mobile a Windows Phone

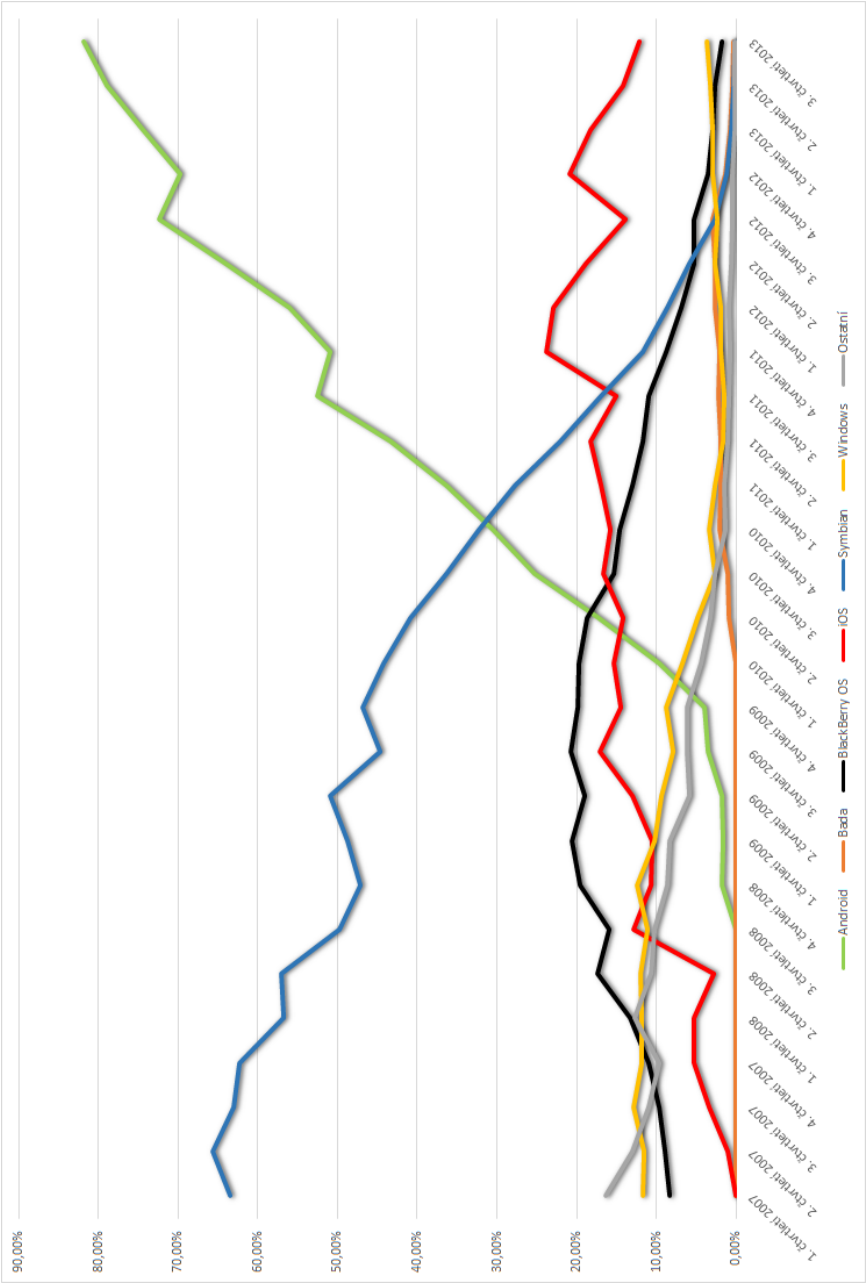
Windows Mobile je mobilní operační systém vyvíjený firmou Microsoft Corporation na jádře systému Windows CE pro mobilní zařízení typu PDA a mobilní telefony. První verze Windows Mobile byla vydána v roce 2000. Windows Mobile distribuován pod licencí Microsoft EULA a jeho zdrojový kód je neveřejný. Operační systém napsán v jazyce C++.

Vývoj Windows Mobile byl ukončen, poslední stabilní verze nese označení 6.5.3 a byla vydána v únoru 2010. Microsoft nicméně poskytuje kompatibilitu aplikacím pro Windows Mobile v operačním systému Windows Embedded Handheld, který je určen převážně pro průmyslová mobilní zařízení.

Windows Phone je proprietární operační systém společnosti Microsoft pro chytré telefony. Windows Phone je vydán s uzavřeným kódem pod licencí Microsoft EULA. V současné době je poslední stabilní verze 8.1. Aplikace pro Windows Mobile a Windows Phone nejsou vzájemně kompatibilní.

Aplikace pro operační systém Windows Phone jsou distribuovány prostřednictvím tržiště aplikací Windows Phone Store[6] (dříve Windows Phone Marketplace), které obsahuje více než 11 000 aplikací [49].

6. `http://www.windowsphone.com/store`




Obrázek 2.5: Podíly jednotlivých operačních systémů na trhu
Zdroj: autor na základě zpráv Gartner Group [11, 12, 13, 17, 16, 15, 14, 21, 19, 18, 20, 25, 22, 24, 23, 29, 26, 27, 28, 33, 30, 32, 31]

# 3 Platforma Android

Android je platforma zahrnující operační systém, middleware a knihovny, určená pro mobilní telefony, tabletové počítače, tzv. inteligentní televize (iTV) a nositelné počítače (z angl. *wearables*).

Android je napsán v jazycích C a C++. Knihovny uživatelského rozhraní jsou napsány v jazyce Java. Android je distribuován jako open-source pod licencí Apache License 2.0 [35].

## 3.1 Historie platformy Android

Android začala v roce 2003 vyvíjet společnost Android, Inc. V roce 2005 tuto společnost koupila firma Google a začlenila ji pod sebe jako jednu ze svých divizí.

V roce 2007 Google, společně s dalšími 34 firmami z oblasti mobilních operátorů, softwarových společností, výrobců hardwaru a dalších, vytvořil konsorcium Open Headset Alliance (OHA). Cílem OHA je vytvářet otevřené standardy pro mobilní zařízení. Google pak vyvíjí Android podle těchto standardů. V současné době má konsorcium OHA 84 členů [53].

Jednotlivé verze platformy Android mají hned několik názvů a kódových označení. Každá verze je označená číslem dle zásad sémantického verzování[1].

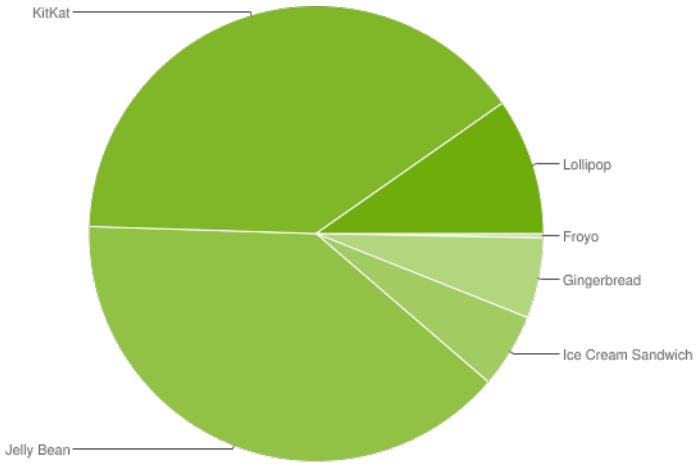
První čtyři verze (bráno včetně vývojářské alfa a betaverze) byly označeny několika neoficiálními kódovými označeními. Od verze 1.5 dostávají nové verze systému abecední kódová označení podle názvů cukrovinek.

Pro vývojáře je nejdůležitější číslování takzvané úrovně API. Toto číslo uvádí verzi systémových knihoven. Systémové knihovny jsou zpětně kompatibilní, ovšem některé funkcionality jsou dostupné až od vyšších úrovní API. Určitou výjimku tvoří verze označované kódovým slovem Honeycomb. Zdrojový kód této verze nebyl volně zveřejněn a byl uvolněn pouze některým výrobcům tabletových počítačů, protože verze Honeycomb byla určena výhradně pro tablety.

1. značení pomocí tří čísel označovaných jako hlavní (*major*), vedlejší (*minor*) a opravné (*patch*) oddělených tečkami [52]

Druhou výjimku tvoří verze s úrovní API 20, které jsou upravenou variantou verzí KitKat a které jsou určené pro nositelná zařízení.

Detailní přehled jednotlivých verzí Androidu ukazuje tabulka 3.1. V době psaní této práce je nejnovější verzi Androidu verze 5.1.1 s kódovým označením Lollipop a úrovní API 22.




Obrázek 3.1: Distribuce verzí platformy Android ke 4. 5. 2015
Zdroj: `http://developer.android.com/about/dashboards`

## 3.2 Hardwarová a softwarová standardizace

Hardware a software zařízení s Androidem je velice měkce definován v dokumentu Compatibility Definition Document[2]. Nízká úroveň povinných standardů vede k tomu, že jednotlivé modely se značně liší co do hardwarového, tak do předinstalovaného softwarového vybavení.

Pro Android jsou také typické tzv. proprietární grafické nástavby (např. TouchWiz společnosti Samsung Electronics nebo HTC Sense společnosti HTC Corporation), které přispívají k další fragmentaci platformy.

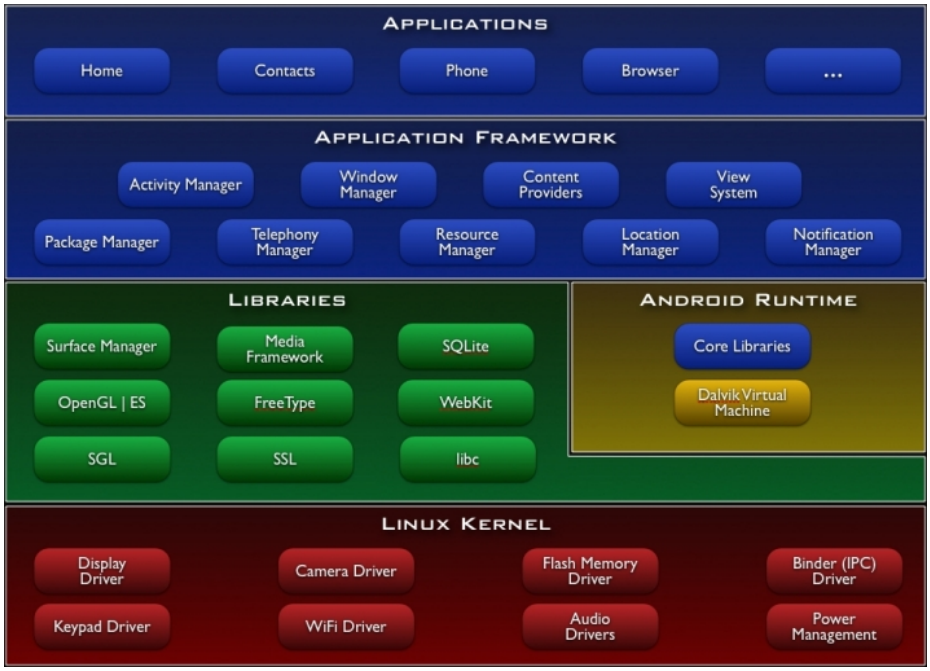
2. `http://source.android.com/compatibility/android-cdd.pdf`

| Verze | Úroveň API | Kódové označení | Datum vydání | Podíl na trhu[37] |
|---|---|---|---|---|
| Android 1.0 | 1 | – | 23. 9. 2008 | < 0,1 % |
| Android 1.1 | 2 | – | 9. 2. 2009 | < 0,1 % |
| Android 1.5 | 3 | Cupcake | 30. 4. 2009 | < 0,1 % |
| Android 1.6 | 4 | Donut | 15. 9. 2009 | < 0,1 % |
| Android 2.0 | 5 | Eclair | 26. 10. 2009 | < 0,1 % |
| Android 2.0.1 | 6 | | 3. 12. 2009 | |
| Android 2.1 | 7 | | 12. 1. 2010 | < 0,1 % |
| Android 2.2 | 8 | Froyo | 20. 5. 2010 | 0,3 % |
| Android 2.2.1 | | | 18. 1. 2011 | |
| Android 2.2.2 | | | 22. 1. 2011 | |
| Android 2.2.3 | | | 21. 11. 2011 | |
| Android 2.3 | 9 | Gingerbread | 6. 12. 2010 | < 0,1 % |
| Android 2.3.1 | | | 1. 1. 2011 | |
| Android 2.3.2 | | | 20. 1. 2011 | |
| Android 2.3.3 | 10 | | 9. 2. 2011 | 5,7 % |
| Android 2.3.4 | | | 28. 4. 2011 | |
| Android 2.3.5 | | | 25. 7. 2011 | |
| Android 2.3.6 | | | 2. 9. 2011 | |
| Android 2.3.7 | | | 15. 9. 2011 | |
| Android 3.0 | 11 | Honeycomb | 22. 2. 2011 | < 0,1 % |
| Android 3.1 | 12 | | 10. 5. 2011 | < 0,1 % |
| Android 3.2 | 13 | | 15. 7. 2011 | < 0,1 % |
| Android 3.2.1 | | | 20. 9. 2011 | |
| Android 3.2.2 | | | 30. 8. 2011 | |
| Android 3.2.4 | | | 15. 12. 2011 | |
| Android 3.2.6 | | | 15. 2. 2012 | |
| Android 4.0 | 14 | Ice Cream Sandwich | 18. 10. 2011 | < 0,1 % |
| Android 4.0.1 | | | 19. 10. 2011 | |
| Android 4.0.2 | | | 28. 11. 2011 | |
| Android 4.0.3 | 15 | | 16. 12. 2011 | 5,3 % |
| Android 4.0.4 | | | 29. 3. 2012 | |
| Android 4.1 | 16 | Jelly Bean | 9. 7. 2012 | 15,6 % |
| Android 4.1.1 | | | 23. 7. 2012 | |
| Android 4.1.2 | | | 9. 10. 2012 | |
| Android 4.2 | 17 | | 13. 11. 2012 | 18,1 % |
| Android 4.2.1 | | | 27. 11. 2012 | |
| Android 4.2.2 | | | 11. 2. 2013 | |
| Android 4.3 | 18 | | 24. 7. 2013 | 5,5 % |
| Android 4.3.1 | | | 3. 10. 2013 | |
| Android 4.4 | 19 | Kit Kat | 31. 10. 2013 | 39,8 % |
| Android 4.4.1 | | | 5. 12. 2013 | |
| Android 4.4.2 | | | 9. 12. 2013 | |
| Android 4.4.3 | | | 2. 6. 2014 | |
| Android 4.4.4 | | | 19. 6. 2014 | |
| Android 4.4W | 20 | Kit Kat (Wearable) | 25. 6. 2014 | |
| Android 4.4W.1 | | | 6. 9. 2014 | |
| Android 4.4W.2 | | | 21. 10. 2014 | |
| Android 5.0 | 21 | Lollipop | 12. 11. 2014 | 9,0 % |
| Android 5.0.1 | | | 2. 12. 2014 | |
| Android 5.0.2 | | | 19. 12. 2014 | |
| Android 5.1 | 22 | | 9. 3. 2015 | 0,7 % |
| Android 5.1.1 | | | 21. 4. 2015 | |

Tabulka 3.1: Přehled verzí Androidu

## 3.3 Architektura platformy Android

Android je postaven na pětivrstvé modulární architektuře (viz obrázek 3.2). Jednotlivé vrstvy od sebe nejsou vždy striktně odděleny, protože některé platformou poskytované funkcionality přesahují hranice jedné vrstvy.




Obrázek 3.2: Architektura platformy Android
Zdroj: `http://developer.android.com/images/system-architecture.jpg`

### 3.3.1 Aplikace

Součástí platformy Android je sada předinstalovaných aplikací jako jsou např. e-mailový klient, klient pro zprávy SMS, kalendář, internetový prohlížeč, správce kontaktů apod.

Důležitou vlastností platformy Android je to, že nerozlišuje mezi předinstalovanými aplikacemi a aplikacemi třetích stran. Vývojář tak může napsat vlastní aplikaci, která může případně nahradit pře-

dinstalovanou aplikaci a která se organicky integruje s ostatními aplikacemi v systému [47]. Vývoji aplikací pro platformu Android se blíže věnuje kapitola 4.

### 3.3.2 Aplikační rámec

Android pro vývojáře poskytuje otevřenou platformu. Díky ní mohou vývojáři volně přistupovat k hardwaru, informacím o fyzické lokaci mobilního zařízení, službám na pozadí, notifikacím na stavové liště a četným dalším klíčovým prvkům pomocí stejného aplikačního programového rozhraní jako předinstalované aplikace.

Mezi prvky aplikačního rámce patří:

- **správce aktivit** (Activity Manager), jenž spravuje životní cyklus aplikací a poskytuje společný navigační zásobník,
- **správce oken** (Window Manager), jenž zobrazuje vizuální komponenty na obrazovce zařízení,
- **poskytovatelé obsahu** (Content Providers), jenž umožňují sdílení dat mezi jednotlivými aplikacemi,
- **sada rozšířitelných vizuálních komponent** (View System), která obsahuje standardizovanou sadu vizuálních ovládacích prvků (jako např. seznamy, mřížky, textová pole pro zadávání uživatelského vstupu, tlačítka, apod.),
- **správce balíčků** (Package Manager), jenž poskytuje informace o aplikacích, které jsou v zařízení nainstalovány,
- **správce telefonie** (Telephony Manager), který umožňuje přístup k telefonní funkcionalitě zařízení,
- **správce zdrojů** (Resource Manager) umožňující přístup ke statickým zdrojům mimo vlastní spustitelný kód aplikace (např. rastrové a vektorové grafice, lokalizovaným řetězcům, rozvržení ovládacích prvků, apod.),

- **správce geolokace** (Location Manager), jenž umožňuje přístup ke geolokační funkcionalitě zařízení pomocí modulu GPS a případně dalšího hardwaru,

- **správce notifikací** (Notification Manager), jenž umožňuje zobrazování upozornění ve stavové liště.

### 3.3.3 Knihovny

Android obsahuje implementace následující sady knihoven napsaných v jazycích C a C++:

- **systémovou knihovnu jazyka C** (libc) – standardní systémovou knihovnu jazyka C odvozenou z BSD verze operačního systému Unix[3] a upravenou pro mobilní zařízení,

- **knihovny pro přehrávání médií** – obsahující knihovnu pro zobrazování grafického a multimediálního obsahu založenou na knihovně OpenCORE firmy PacketVideo, díky níž Android podporuje zobrazování formátů MPEG4, H.264, MP3, AAC, AMR, JPEG, PNG a dalších,

- **Surface Manager** – umožňující kompozitní skládání 2D a 3D obsahu z různých aplikací,

- **WebKit**[4] – moderní renderovací jádro pro prohlížení webových stránek,

- **SGL**[5] – knihovnu pro zobrazování 2D grafiky,

- **3D knihovny** – implementaci rozhraní OpenGL ES[6], díky níž aplikace mohou používat hardwarovou 3D akceleraci (v zařízeních, která ji podporují) nebo vysoce výkonný softwarový 3D rasterizér,

3. http://www.cplusplus.com/reference/cstdlib/
4. http://www.webkit.org/
5. http://sgl.sourceforge.net/
6. http://www.khronos.org/opengles/

- **FreeType**[7] – knihovnu pro bitmapové a vektorové renderování písma,

- **SQLite**[8] – výkonný a nenáročný systém řízení báze dat dostupný pro všechny aplikace.

### 3.3.4 Android Runtime

Android používá vlastní implementaci virtuálního stroje jazyka Java jménem Dalvik Virtual Machine. Referenční virtuální stroj Java Virtual Machine totiž není volně šiřitelný a Dalvik je také lépe optimalizován vzhledem k omezujícím podmínkám mobilních zařízení.

Každá aplikace v systému Android běží ve vlastním procesu s vlastní instancí virtuálního stroje Dalvik. Architektura Dalviku umožňuje efektivní spouštění více virtuálních strojů zároveň.

Důležité je zmínit, že Dalvik nespouští přímo binární soubory v Java bytekódu, nýbrž tyto binární soubory napřed převádí do vlastního formátu (označovaného příponou `.dex` – Dalvik Executable), který je optimalizovaný z hlediska využití operační paměti.

Ve verzi 4.4 (KitKat) přibyla do systému také druhá implementace virtuálního stroje jménem Android Runtime (ART), která využívá tzv. ahead-of-time kompilaci, při které je aplikace převedena do nativního kódu při instalaci. Výsledkem jsou rychlejší aplikace za cenu dodatečné sekundární paměti. Ve verzi 5.0 (Lollipop) byl Dalvik plně nahrazen ARTem.

### 3.3.5 Linuxové jádro

Android je vystavěn nad linuxovým jádrem. Toto jádro se stará o nízkoúrovňové funkce jako je například správa paměti, souborového systému, sítí apod. Linuxové jádro poskytuje abstraktní vrstvu architektonicky vyšším vrstvám [36].

---

7. `http://www.freetype.org/`
8. `http://sqlite.org/`

## 3.4 Nástroje pro vývoj aplikací

Vývoj aplikací pro Android probíhá v tzv. vývojovém režimu hostitel-cíl[9]. Aplikace pro Android je možné vyvíjet na počítačích s operačním systémem Windows, Linux a OS X.

Přestože je sestavení aplikace pro Android poměrně komplikované, co se počtu kroků týče, součástí nástrojů pro vývojáře je i řada programů a skriptů, které tyto kroky umožňují zjednodušit a automatizovat.

### 3.4.1 Nástroje pro vývojáře

Sada nástrojů pro vývojáře (SDK) obsahuje mimo jiné následující programy usnadňující vývoj aplikací pro Android:

- **SDK Manager a AVD Manager** – vizuální nástroje, které umožňují automaticky stahovat zdrojové kódy a knihovny pro jednotlivé verze platformy Android a nástroj umožňující jednoduchou správu emulátorů,
- **Dx** – nástroj pro konverzi zkompilovaných soubory ve formátu Java bytekódu do formátu Dalvik Executable,
- **Android Asset Packaging Tool (AAPT)** – nástroj, který vytváří ze zkompilovaných souborů typu `.dex` a statických zdrojů balíčky typu `.apk`, které slouží k distribuci aplikací pro Android,
- **Android Debug Bridge (ADB)** – nástroj typu klient-server, který zajišťuje komunikaci s emulátory a fyzickými zařízeními,
- **Dalvik Debug Monitoring Services (DDMS)** – nástroj umožňující monitorování a ovládání emulátorů a fyzických zařízení se systémem Android.

9. z angl. *Host-Target*, tedy v režimu kdy je aplikace vyvíjená v jiném prostředí, než do kterého je nasazena [54]

### 3.4.2 Android Studio

Android Studio[10] je v současné době oficiálním vývojovým prostředím určeným pro vývoj aplikací pro Android, které tuto pozici převzalo po níže zmíněném Eclipse. Je postaveno nad komunitní verzi vývojového prostředí IntelliJ IDEA[11] české firmy JetBrains. Android Studio je k dispozici pro operační systémy Windows, Linux a OS X.

### 3.4.3 Android Development Tools pro Eclipse

Eclipse[12] je vývojové prostředí napsané v jazyce Java podporující mnoho programovacích jazyků jako jsou např. Java, C, C++, Ruby, PHP a mnohé další [10]. Eclipse je distribuován jako open-source a je dostupný pro operační systémy Windows, Linux, Solaris a OS X.

Nástroje pro vývojáře jsou integrovány do prostředí Eclipse prostřednictvím zásuvného modulu jménem ADT (Android Development Tools) [34].

### 3.4.4 Emulátor

Pro testování a vývoj není zapotřebí fyzického zařízení, ale je možné využít emulátoru, který je součástí SDK. Historicky byly s používáním emulátoru problémy z důvodu jeho značné pomalosti a dlouhé doba odezvy při zpracovávání grafických instrukcí. Tyto nedostatky byly způsobeny dvěma důvody.

Za prvé, Android byl původně vyvíjen pouze pro procesory typu ARM a na běžném počítači typu PC se tak kromě operačního systému musela emulovat i procesorová architektura.

Za druhé, grafika emulátoru byla vykreslována softwarově a nikoli s podporou grafického akcelerátoru. Výsledkem proto byla trhaná grafika a pomalý emulátor. Toto omezení již bylo odstraněno a pro vykreslování emulátoru je možné použít implementaci rozhraní OpenGL hostitelského počítače.

---

10. `https://developer.android.com/sdk/index.html`
11. `https://www.jetbrains.com/idea/`
12. `http://www.eclipse.org/`

Firma Intel vytvořila obrazy systému Android emulátor architektuře Intel x86 a které jsou spustitelné pomocí nástroje Intel HAXM[13] (Hardware Accelerated Execution Manager). HAXM je dostupný pro operační systém Windows, pro Linux jej lze využít spolu s nástrojem jménem KVM[14] (Kernel-Base Virtual Machine). Ukázku emulace s využitím technologie Intel HAXM ukazuje obrázek 3.3.

## 3.5 Distribuce aplikací pro platformu Android

Aplikace pro Android se distribuují ve formátu APK (Android Package) označovaném příponou `.apk`. Android Package je v podstatě zkomprimovaný archiv typu `zip`, který obsahuje zkompilované soubory tříd ve formátu Dalvik Executable, statické zdroje a další metasoubory, které aplikace vyžaduje pro svou instalaci na mobilní zařízení a běh.

Podobně jako na ostatních platformách i Android má v základní verzi systému předinstalovanou aplikaci, která slouží jako distribuční systém neboli tržiště aplikací s názvem Google Play.

### 3.5.1 Google Play

Tržiště společnosti Google zahájilo svou činnost pod názvem Android Market 23. října 2008, ale 6. března 2012 bylo přejmenováno na Google Play.

Aplikace na Google Play oproti některým jiným tržištím nepodléhají žádným schvalovacím řízením, publikovat na Google Play může každý vývojář, který dodrží smluvní podmínky a zaplatí pomocí platební karty jednorázový poplatek 25 USD. Publikaci aplikací na Google Play se blíže věnuje kapitola 6.

V České republice lze placené aplikace z Google Play stahovat od 1. 10. 2010 [44]. Od 11. dubna 2012 se navíc Česká republika stala jednou z 74 zemí [39], ze kterých mohou být placené aplikace distribuovány.

---

13. `http://software.intel.com/en-us/articles/intel-hardware-accelerated-execution-manager`
14. `http://www.linux-kvm.org/page/Main_Page`

### 3.5.2 Amazon App Store

Konkurencí ke Google Play je tržiště aplikací Amazon App Store[15]. Amazon App Store je momentálně dostupný v téměř dvou stech zemích světa. Není nutné platit žádný vývojářský poplatek, ale aplikace na rozdíl od Google Play procházejí schvalovacím řízením.

Amazon App Store je předinstalován na tabletech Kindle Fire společnosti Amazon a také na telefonech BlackBerry s verzí BlackBerry OS 10.3.

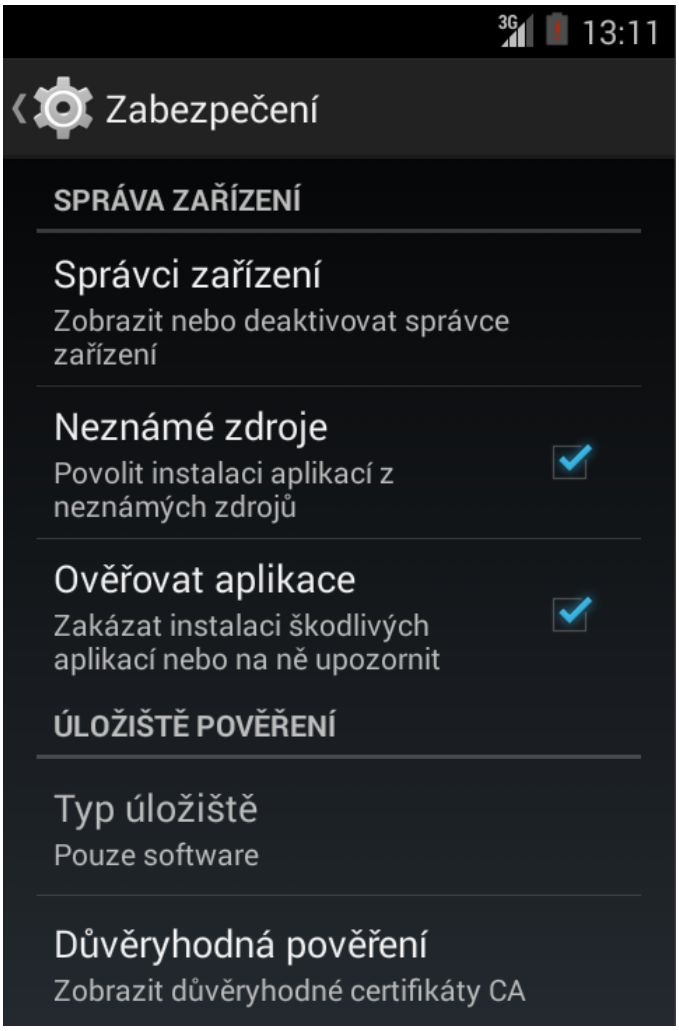
### 3.5.3 Mimotržištní distribuce

Oproti ostatním platformám umožňuje Android i instalaci aplikací, které nepocházejí z tržiště aplikací a to zcela bez omezení. Po nahrání balíčku typu (`.apk`) do cílového zařízení je možné tuto aplikaci nainstalovat otevřením tohoto souboru.

V nastavení zařízení je avšak nutné explicitně povolit instalaci aplikací z neznámých zdrojů (viz obrázek 3.4).

15. `www.amazon.com/mobile-apps/`

Obrázek 3.3: Emulace Androidu 4.4.2 (KitKat) ve Windows 8
Zdroj: autor




Obrázek 3.4: Povolení aplikací z neznámých zdrojů
Zdroj: autor

# 4 Vývoj aplikací pro platformu Android

Pro vývoj aplikací pro platformu Android lze použít několik vývojových nástrojů. V této kapitole je krátce představím, přičemž blíže se věnuji vývoji pomocí Android SDK.

## 4.1 Možnosti vývoje aplikací pro platformu Android

### 4.1.1 Android SDK a Android NDK

Součástí vývojových nástrojů je tzv. SDK (Software Development Kit, viz podkapitola 3.4.1) – sada knihoven v jazyce Java pro snadnou práci se systémem, jeho aplikačními rozhraními a službami, které v rámci operačního systému běží.

Pro výpočetně náročné operace jako je například zpracování videa nebo fyzikální simulace nebo je možné využít tzv. NDK (Native Development Kit)[1], jenž umožňuje nízkoúrovňový přístup k operačnímu systému pomocí jazyka C nebo C++.

### 4.1.2 Webové aplikace

Součástí platformy Android je také webový prohlížeč s jádrem Webkit a grafická komponenta `WebView`[2]. S využitím této komponenty je možné aplikaci vyvíjet jako klasickou webovou aplikaci.

Nevýhodou tohoto řešení je, že prohlížení webových stránek je strojově poměrně náročná operace, která oproti aplikacím vytvořených pomocí SDK/NDK zatěžuje mnohem více procesor a díky tomu i akumulátor. Z prohlížeče také není možné využívat většinu pokročilých funkcí.

### 4.1.3 Alternativní vývoj

Pro usnadnění vývoje mobilních aplikací pro více platforem existují tzv. multiplatformní aplikační rámce pro tvorbu mobilních aplikací.

---

1. `https://developer.android.com/tools/sdk/ndk/index.html`
2. `http://developer.android.com/reference/android/webkit/WebView.html`

Obvykle řeší vývoj jako webovou aplikaci psanou v JavaScriptu a některé zpřístupňují i rozhraní pro pokročilé funkce dané platformy pomocí rozšiřujících knihoven. Příklady takových aplikačních rámců jsou např. RhoMobile[3], Appcelerator[4] nebo PhoneGap[5].

Existují také možnosti, jak vyvíjet aplikace pro Android ve zcela jiných jazycích. Příklady takových projektů jsou například Ruboto[6] pro skriptovací jazyk Ruby nebo Xamarin[7] pro jazyk C#.

## 4.2 Vývoj aplikací pomocí Android SDK

Vývoj aplikací pomocí Android SDK je nejčastějším způsobem vyvíjení aplikací pro platformu Android. V této podkapitole představím základy vývoje pro platformu Android a jeho specifika oproti programování v jazyce Java.

### 4.2.1 Komponenty aplikace pro Android

Aplikace pro Android jsou sestaveny ze čtyř základních komponent: aktivit, služeb, poskytovatelů obsahu a přijímačů vysílání.

V literatuře je také možné nalézt zmínku o páté základní komponentě – `Application`[8], která odkazuje na reprezentaci vlastní aplikace a jejího životního cyklu [45].

#### Aktivity (Activities)

Aktivita je základní vizuální komponenta aplikace. Aktivity se v mnohém podobají například formulářům z aplikačního rozhraní Windows Forms nebo rámečkům z knihovny Swing jazyka Java.

Aktivity v prvé řadě slouží jako kontejner pro grafické ovládací prvky jako jsou například tlačítka, zaškrtávací políčka, textová pole

3. http://rhomobile.com
4. http://www.appcelerator.com/
5. http://phonegap.com/
6. http://ruboto.org/
7. http://xamarin.com/platform
8. http://developer.android.com/reference/android/app/Application.html

a další. Aktivity jsou však v mnohém oproti výše jmenovaným příkladům i značně odlišné. V prvé řadě jsou mnohem volněji provázány. Ačkoli každá aplikace vytvořená pomocí Android SDK obsahuje nejméně jednu aktivitu, aktivity implicitně nemají povědomí o existenci ostatních aktivit v aplikaci a data pro jejich vzájemnou komunikaci je třeba mezi nimi předávat pomocí primitivních nebo serializovatelných typů (této problematice se blíže věnuje podkapitola 4.2.5).

Zároveň, aktivity mohou spouštět i aktivity jiných aplikací. To je obzvláště užitečné při využívání funkcionalit předinstalovaných aplikací (viz podkapitola 4.2.4).

Aktivity zpravidla vyplňují celou obrazovku zařízení, přestože to není podmínkou.

#### Služby (Services)

Služby jsou komponenty, které slouží ke zpracování dlouhotrvajících operací na pozadí. Na rozdíl od aktivit nemají žádnou pro uživatele viditelnou reprezentaci. Služby je možné spouštět ve dvou režimech:

**Vázaný režim (Bound)**

Ve vázaném režimu spouští službu jiná komponenta. Ty s touto službou mohou interagovat, dokud existuje alespoň jedna komponenta, která má tuto službu navázánu. V opačném případě se služba automaticky ukončí.

**Spouštěný režim (Started)**

Ve spouštěném režimu služba typicky vykoná určitou činnost a následně se sama ukončí.

Toto rozdělení není vzájemně exkluzivní, mohou existovat služby, které umožňují jak vázaný, tak spouštěný režim.

#### Poskytovatelé obsahu (Content providers)

Poskytovatelé obsahu jsou komponenty, které umožňují sdílení dat mezi aplikacemi. S poskytovateli obsahu je možné komunikovat pomocí aplikačního programového rozhraní, které připomíná rozhraní

webových služeb typu REST. Pro přístup k poskytovatelům obsahu se používá speciální URI s prefixem `content://`.

#### Přijímače vysílání (Broadcast receivers)

Přijímače vysílání jsou komponenty, které reagují na určité události probíhající v rámci systému nebo aplikace. Příkladem takové události je například spuštění systému (kterému odpovídá systémová událost `android.intent.action.BOOT_COMPLETED`[9]) nebo připojení zařízení k adaptéru (systémová událost `android.intent.action.ACTION_POWER_CONNECTED`[10]).

Od verze 3.1 je Android implicitně blokuje přijímače aplikací, které na daném zařízení byly nainstalovány, ale nebyly nikdy spuštěny [56].

### 4.2.2 Fragmenty (Fragments)

V souvislosti se základními komponentami je vhodné zmínit také novější, hojně používané komponenty, kterými jsou tzv. fragmenty. Fragmenty byly představeny poprvé ve verzi 3.0. Fragment je vizuálně-funkční komponenta, který se v mnoha ohledech chová podobně jako aktivita, sama však aktivitou není. Jedna aktivita může obsahovat několik fragmentů.

Od úrovně API 4.2 (Jelly Bean) je možné navíc vkládat fragmenty také do fragmentů jako tzv. zanořené fragmenty.

### 4.2.3 Manifest

Součástí každé aplikace pro platformu Android je soubor `AndroidManifest.xml`, který se nachází v kořenovém adresáři projektu. Jedná se o jakýsi základní rozcestník, který popisuje pro které verze systému je aplikace určena, seznam komponent, verzi aplikace a další metadata[11].

---

9. `http://developer.android.com/reference/android/content/Intent.html#BOOT_COMPLETED`
10. `http://developer.android.com/reference/android/content/Intent.html#ACTION_POWER_CONNECTED`
11. `http://developer.android.com/guide/topics/manifest/manifest-intro.html`

### 4.2.4 Záměry (Intents)

Záměry jsou základním způsobem komunikace uvnitř aplikace a mezi aplikacemi navzájem. Pomocí záměrů lze spouštět aktivity a služby a vysílat signály přijímané přijímači vysílání.

Záměry se dělí na dvě kategorie.

#### Explicitní záměry

Explicitní záměry slouží ke spuštění konkrétní aktivity nebo služby definované svou třídou. Příklad spuštění konkrétní aktivity explicitním záměrem uvádí zdrojový kód 4.1.

Zdrojový kód 4.1: Spuštění aktivity pomocí explicitního záměru

```
1 Intent intent =
2     new Intent(context, JointActivity.class);
3 startActivity(intent);
```

#### Implicitní záměry

Implicitní záměry fungují obecněji než záměry explicitní. Nespecifikují konkrétní třídu, ale obecnou akci, která je záměrem spuštěna. Při vyvolání obecného záměru se spustí aktivita, která je uživatelem preferována nebo seznam aplikací, které dokáží daný záměr zpracovat.

Příklad využití implicitního záměru uvádí zdrojový kód 4.2.

Zdrojový kód 4.2: Spuštění aktivity pomocí implicitního záměru

```
1 Intent sendIntent = new Intent(Intent.ACTION_SEND);
2 sendIntent.putExtra(
3     Intent.EXTRA_TEXT,
4     "Text sdileny zamerem."
5 );
6 sendIntent.setType("text/plain");
7 startActivity(sendIntent);
```

### 4.2.5 Serializace

Předávání parametrů pro komunikaci a inicializace komponent se typicky provádí pomocí třídy `Bundle`[12], která funguje jako asociativní pole s klíči typu řetězec (`String`). Hodnoty jsou pak primitivní typy, případně serializovatelné objekty.

V Androidu je možné použít standardní rozhraní `Serializable` definovaným v balíčku `java.io`. Nicméně, vzhledem výpočetním nárokům je toto rozhraní nevhodné pro mobilní zařízení.

Proto je součástí Android SDK rozhraní `Parcelable`[13] v balíčku `android.os`. To předepisuje pro serializovatelné třídy metodu `writeToParcel` ve které probíhá vlastní serializace a statický člen `CREATOR`, který předepisuje proces deserializace.

### 4.2.6 Práce se zdroji

Android poskytuje jednoduché rozhraní pro přístup je statickým zdrojům jako jsou například grafické soubory, textové konstanty, animace, zvukové soubory nebo konfigurační soubory. Pomocí statických zdrojů lze také vytvářet rozložení grafických uživatelských prvků a aplikačního menu snadněji, než pomocí instrukcí jazyka Java. Statické zdroje se umisťují do adresáře `res` v kořenovém adresáři projektu.

Ke statickým zdrojům je možné přistupovat ze třídy `Context`[14] či jejích potomků (zejména pak tříd `Activity`, `Service` a `Application`). Příklad přístupu k textové konstantě ukazuje zdrojový kód 4.3.

Pomocí tohoto rozhraní je možné také poskytovat různé zdroje pro různé typy a konfigurace. Například implicitně se textové konstanty umisťují do souboru `strings.xml` v adresáři `res/values`. Například při lokalizaci do jiných jazyků je možné vytvořit adresáře s postfixem kódu daného jazyka (tedy např. pro češtinu adresář

12. `http://developer.android.com/reference/android/os/Bundle.html`
13. `http://developer.android.com/reference/android/os/Parcelable.html`
14. `http://developer.android.com/reference/android/content/Context.html`

values-cs). Pokud daný zdroj v daném jazyce neexistuje, použije se implicitní hodnota.

Při sestavení projektu se v adresáři gen vytvoří třída R, která obsahuje identifikátory definovaných statických zdrojů.

Zdrojový kód 4.3: Přístup k textovým konstantám

```
1 Resources resources = getResources();
2 String string = resources.getString(R.string.app_name);
```

Obdobný způsob lze použít také u grafických souborů v návaznosti na rozlišení a rozměry obrazovky, orientaci displeje, typ klávesnice apod[15].

Ke statickým zdrojům definovaným samotnou platformou je možné přistupovat pomocí třídy R umístěné v balíčku android.

15. http://developer.android.com/guide/topics/resources/providing-resources.html

# 5 Aplikace Hladová okna

Cílem vyvíjené aplikace je vytvořit řešení typu klient-server, pomocí kterého budou moci uživatelé participovat na vytváření obsahu.

### 5.0.1 Průzkum mezi potenciálními uživateli

V rámci vyvíjené práce jsem provedl internetový průzkum mezi potenciálními uživateli[1]. Kompletní výsledky včetně zdrojových dat jsou na přiloženém CD.

Z průzkumu vyplývá, že většina uživatelů nevyužívá mobilní datové připojení, proto je vhodné, aby aplikace co možná nejvíce funkcionality poskytovala v režimu offline.

### 5.0.2 Doména

Organizace v rámci domény provozovatelů rychlého občerstvení lze kategorizovat do dvou hlavních skupin.

**Globální řetězce**

První skupinu tvoří řetězce rychlého občerstvení. Informace o těchto řetězcích jsou zpravidla dostupné v nějaké formě na Internetu (prezentace ve formě webových stránek, katalogy firem, apod.). Také obvykle fungují na základě franšízového obchodního modelu, což ale znamená, že ceny sortimentu uvedené v oficiálních materiálech provozovatelů, jsou často označeny pouze jako *doporučené* a na konkrétních provozovnách se mohou lišit. Stejně se může lišit i otevírací doba jednotlivých provozoven a nabízený sortiment.

Mezi tyto řetězce patří např. v Česku známé řetězce McDonald's, KFC nebo Subway. Z českých můžeme jmenovat např. síť stánků Parky's.

1. `https://www.vyplnto.cz/realizovane-pruzkumy/pruzkum-trhu-mobilnich-telef/`

**Jednotlivé lokální provozovny**

Druhou skupinu tvoří jednotlivé provozovny, které typicky nemají žádné webové stránky nebo většinou nemají webové stránky a informace o jejich provozovnách, sortimentu, otevírací době, apod. nelze snadno převzít ze žádného veřejně dostupného informačního zdroje.

Co obě skupiny spojuje je fakt, že nelze snadno přebírat data pro aplikaci z žádného veřejně dostupného zdroje. Řešením je proto vytvoření vlastní, komunitně spravované databáze.

### 5.0.3 Komunitní aplikace

Komunitní aplikace je typ aplikace, která počítá s aktivním zapojením svých uživatelů do procesu zadávání, úpravy a případně i publikace dat.

Je možné je rozdělit do následujících dvou skupin.

**Moderované aplikace**

Moderované aplikace jsou takové, u nichž je změna (např. v datech) schvalována moderátorem před vlastní publikací.

**Nemoderované aplikace**

U nemoderovaných aplikací se uživatelem provedená změna projeví okamžitě a neprochází schvalovacím řízením.

Toto vymezení není striktní, protože existují aplikace, které oba přístupy kombinují, např. internetová encyklopedie Wikipedia[2], která je implicitně nemoderovaná, nicméně u tzv. zamčených stránek využívá moderovaný režim.

Pro aplikaci Hladová okna jsem zvolil moderovaný typ. Hlavním důvodem je snaha o přesnost dat a také možnost snadné obrany proti spamu a případným *destruktivním* snahám některých uživatelů. V případě mobilní aplikace navíc schvalovací řízení neodrazuje uživatele, neboť uživatelské zařízení obsahuje vlastní databázi, změny se uživateli zobrazí ihned a zbytku uživatelské komunity až po schválení moderátorem.

2. `http://www.wikipedia.org`

## 5.1 Diagramy

### 5.1.1 Diagram případů užití

Diagram případů užití (obrázek 5.1) ukazuje 2 aktéry a 7 případů užití, které jsou blíže popsány níže.

#### Aktéři

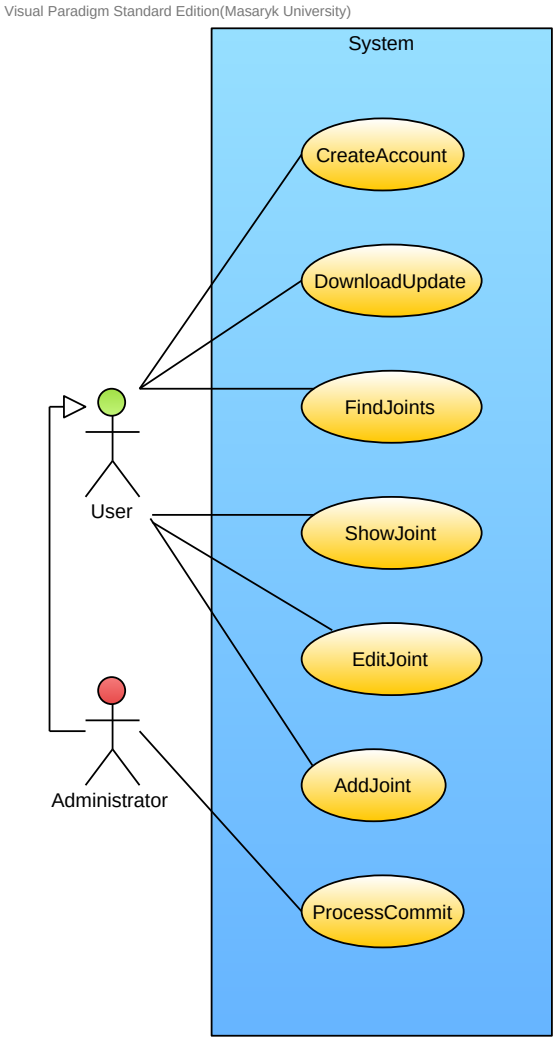
**User**

Aktér `User` představuje běžného uživatele aplikace.

**Administrator**

Aktér `Administrator` představuje uživatele s administrátorským oprávněním, který moderuje databázi aplikace.

| | |
|---|---|
| **Název:** | **`CreateAccount`** |
| **Identifikátor:** | UC01 |
| **Popis:** | UC01 trvale přihlásí uživatele k uživatelskému účtu. |
| **Primární aktéři:** | `User` |
| **Sekundární aktéři:** | *žádní* |
| **Vstupní podmínky:** | Uživatel není přihlášen. |
| **Tok událostí:** | 1. Uživatel zadá e-mail, heslo a zda si přeje vytvořit nový účet.<br>2. Uživatel potvrdí zadané hodnoty.<br>3. Systém ověří zadané hodnoty.<br>4. POKUD jsou hodnoty validní:<br>(a) POKUD uživatel zadal volnu *vytvořit nový účet*:<br>i. Systém vytvoří nový účet.<br>(b) Systém trvale přihlásí uživatele do systému.<br>5. JINAK:<br>(a) Systém zobrazí chybové hlášení.<br>(b) GOTO 1. |
| **Alternativní toky:** | *žádné* |
| **Výstupní podmínky:** | Uživatel je přihlášen. |




Obrázek 5.1: Diagram případů užití
Zdroj: autor

| | |
|---|---|
| **Název:** | **`DownloadUpdate`** |
| **Identifikátor:** | UC02 |
| **Popis:** | UC02 stáhne uživatelem zvolenou aktualizaci. |
| **Primární aktéři:** | User |
| **Sekundární aktéři:** | *žádní* |
| **Vstupní podmínky:** | Uživatel je přihlášen. |
| **Tok událostí:** | 1. Uživatel zvolí balíčky provozoven, které chce mít v zařízení dostupné.<br>2. Uživatel potvrdí zvolené balíčky.<br>3. POKUD je dostupné připojení k Internetu:<br>(a) Systém stáhne požadovaná data.<br>(b) Systém zpraví uživatele o dokončení stahování.<br>4. JINAK:<br>(a) Systém zobrazí zprávu o nedostupnosti připojení. |
| **Alternativní toky:** | *žádné* |
| **Výstupní podmínky:** | Databáze je v konzistentním stavu. |

| | |
|---|---|
| **Název:** | **`FindJoints`** |
| **Identifikátor:** | UC03 |
| **Popis:** | UC03 vyhledá a zobrazí provozovny dle uživatelova vstupu. |
| **Primární aktéři:** | User |
| **Sekundární aktéři:** | *žádní* |
| **Vstupní podmínky:** | Uživatel je přihlášen. |
| **Tok událostí:** | 1. Uživatel zadá část názvu hledané provozovny a zda prohledávat pouze právě otevřené provozovny.<br>2. Systém provede vyhledávání v lokální databázi na základě zadané část názvu.<br>3. POKUD existuje jeden či více výsledků vyhledávání:<br>(a) Systém uživateli zobrazí výsledky vyhledávání.<br>4. JINAK:<br>(a) Systém zobrazí uživateli informaci o nulovém počtu vyhledávacích výsledků. |
| **Alternativní toky:** | *žádné* |
| **Výstupní podmínky:** | Databáze je v konzistentním stavu. |

| | |
|---|---|
| **Název:** | `ShowJoint` |
| **Identifikátor:** | UC04 |
| **Popis:** | UC04 zobrazí informace o požadované provozovně rychlého občerstvení. |
| **Primární aktéři:** | User |
| **Sekundární aktéři:** | *žádní* |
| **Vstupní podmínky:** | Uživatel je přihlášen. |
| **Tok událostí:** | 1. Uživatel zvolí požadovanou provozovnu rychlého občerstvení.<br>2. Systém zobrazí informace o požadované provozovně. |
| **Alternativní toky:** | *žádné* |
| **Výstupní podmínky:** | *žádné* |

| | |
|---|---|
| **Název:** | `EditJoint` |
| **Identifikátor:** | UC05 |
| **Popis:** | UC05 upraví atributy požadované provozovny rychlého občerstvení. |
| **Primární aktéři:** | User |
| **Sekundární aktéři:** | *žádní* |
| **Vstupní podmínky:** | Uživatel je přihlášen. |
| **Tok událostí:** | 1. Uživatel zvolí požadovanou provozovnu rychlého občerstvení.<br>2. Uživatel zadá požadované změny atributů provozovny.<br>3. Systém ověří zadaná data.<br>4. POKUD jsou data validní:<br>(a) Systém uloží zadané atributy.<br>(b) Systém zobrazí zprávu potvrzující dokončení operace.<br>5. JINAK:<br>(a) Systém zobrazí chybové hlášení.<br>(b) GOTO: 2 |
| **Alternativní toky:** | *žádné* |
| **Výstupní podmínky:** | Databáze je v konzistentním stavu. |

| | |
|---|---|
| **Název:** | **`AddJoint`** |
| **Identifikátor:** | UC06 |
| **Popis:** | UC06 Vytvoří novou provozovnu rychlého občerstvení. |
| **Primární aktéři:** | `User` |
| **Sekundární aktéři:** | *žádní* |
| **Vstupní podmínky:** | Uživatel je přihlášen. |
| **Tok událostí:** | 1. Uživatel zvolí volbu *Přidat hladové okno*.<br>2. Uživatel zadá atributy provozovny do formuláře.<br>3. Uživatel potvrdí zadané hodnoty.<br>4. Systém ověří zadané hodnoty.<br>5. POKUD jsou data korektní:<br>(a) Systém zobrazí zprávu o úspěšném přidání nové provozovny.<br>6. JINAK:<br>(a) Systém zobrazí chybové hlášení.<br>(b) GOTO 2.<br>7. Systém informuje uživatele o provedení změny. |
| **Alternativní toky:** | *žádné* |
| **Výstupní podmínky:** | Databáze je v konzistentním stavu. |

| | |
|---|---|
| **Název:** | **`ProcessCommit`** |
| **Identifikátor:** | UC07 |
| **Popis:** | UC07 zpracuje navrhovanou uživatelskou změnu. |
| **Primární aktéři:** | `Administrator` |
| **Sekundární aktéři:** | *žádní* |
| Vstupní podmínky | Administrátor je přihlášen. |
| Tok událostí | 1. Administrátor zvolí požadovanou uživatelskou změnu.<br>2. Administrátor zvolí akci, kterou chce provést na zvolené uživatelské změně.<br>3. Administrátor stvrdí požadovanou akci.<br>4. POKUD je požadovanou akcí schválení:<br>(a) Systém aktualizuje serverovou databázi dle uživatelské změny.<br>5. POKUD je požadovanou akcí zamítnutí:<br>(a) Systém odstraní na serveru požadovanou uživatelskou změnu. |
| **Alternativní toky:** | *žádné* |
| **Výstupní podmínky:** | Databáze je v konzistentním stavu. |

### 5.1.2 Konceptuální ERD diagram

Konceptuální diagram (viz obrázek 5.2) popisuje celkem 7 entitních množin, které jsou stručně popsány níže.

#### Entitní množina `Joint`

Entitní množina `Joint` obsahuje objekty typu provozovna rychlého občerstvení.

#### Entitní množina `Meal`

Entitní množina `Meal` obsahuje objekty typu pokrm či nápoj.

#### Entitní množina `MenuItem`

Entitní množina `MenuItem` je vazební množina mezi entitními množinami `Joint` a `Meal`. Vyjadřuje vztah typu *M:N* mezi těmito entitními množinami.

**Entitní množina `Pack`**

Entitní množina `Pack` obsahuje objekty typu balíček aktualizací aplikace Hladová okna.

**Entitní množina `Photo`**

Entitní množina `Photo` obsahuje objekty typu fotografie provozovny rychlého občerstvení.

**Entitní množina `User`**

Entitní množina `User` obsahuje objekty typu uživatel aplikace Hladová okna.

### 5.1.3 Fyzický ERD diagram (server)

Oproti konceptuálnímu návrhu je na serveru obsažena ještě pomocná tabulka `commits`, která obsahuje neschválené změny zadané uživateli aplikace. Datové typy odpovídají implementaci v databázi MySQL.

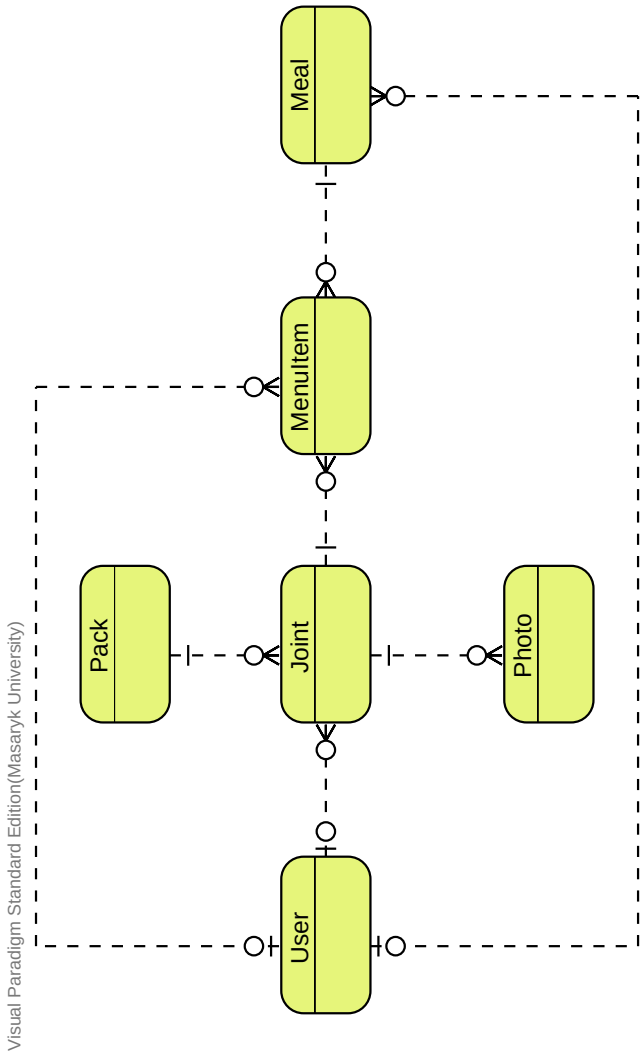
Fyzický entitně-relační diagram serverové databáze ukazuje obrázek 5.3.

### 5.1.4 Fyzický ERD diagram (klient)

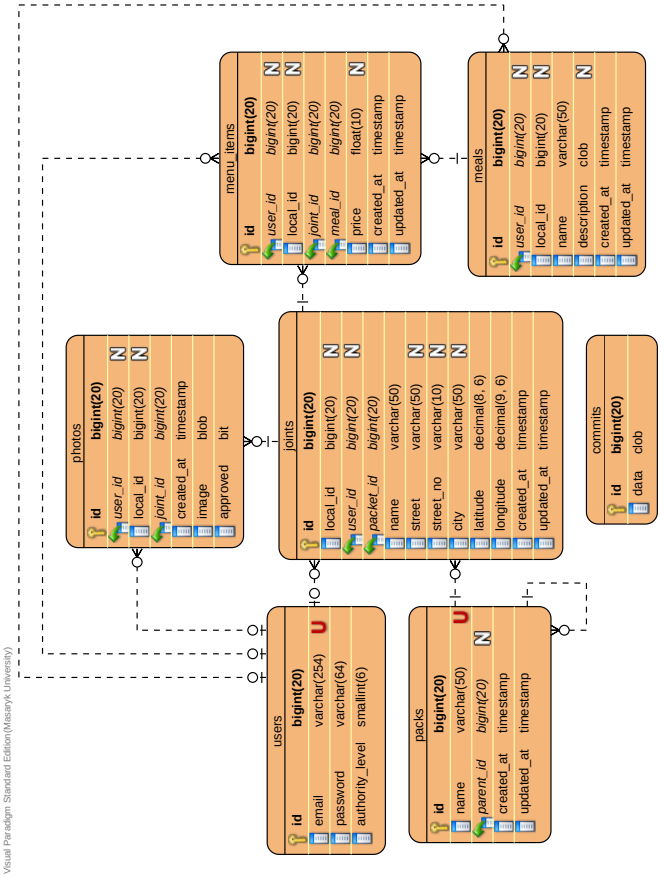
Oproti serverové implementaci není v klientské databázi vůbec obsažena tabulka `users`, protože v aplikaci je uložen vždy nejvýše jeden přihlašovací účet. Tento účet ukládám jako tzv. synchronizační účet[3]. Také chybí tabulka `commits`, jelikož změny vytvořené jednotlivými uživateli není nutné synchronizovat a uchovávat lokálně. Oproti serveru není nutné ukládat celou hierarchickou strukturu balíčků, proto chybí reflexivní vazba tabulky `packs`.

Datové typy odpovídají implementaci v databázi SQLite. Výsledný diagram popisuje obrázek 5.4.

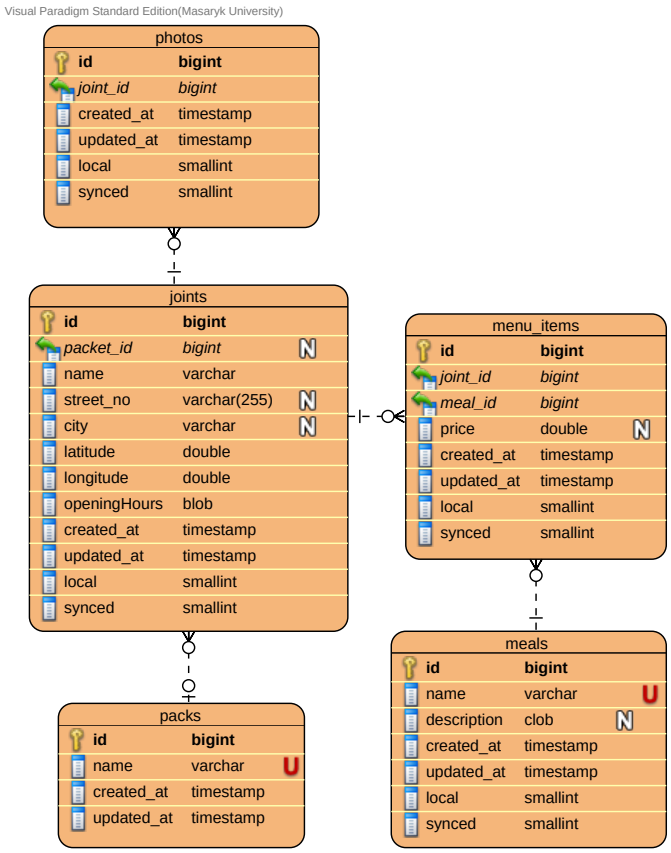
3. `http://developer.android.com/reference/android/accounts/Account.html`




Obrázek 5.2: Konceptuální ERD diagram
Zdroj: autor




Obrázek 5.3: Fyzický ERD diagram (server)
Zdroj: autor




Obrázek 5.4: Fyzický ERD diagram (klient)
Zdroj: autor

## 5.2 Klient

### 5.2.1 Komunikační rozhraní

Klient komunikuje se serverem pomocí protokolu HTTPS. Komunikaci s tímto protokolem v Androidu zajišťuje třída `HttpUrlConnection`[4]. Server poskytuje aplikační programové rozhraní ve formě webové služby typu REST. Pro výměnu dat je použit formátu JSON[5].

JSON je generován objektovým mapovačem[6] knihovny Jackson, který automaticky vytvoří ze všech vlastností dané třídy vytvoří objekt ve formátu JSON. Stejný postup je pak použit v druhé části aplikace pro následnou deserializaci.

Během vývoje jsem úspěšně použil nástroj pro dokumentaci a testování serverových aplikačních programových rozhraní Apiary[7].

Kompletní dokumentace aplikačního rozhraní včetně vzorových požadavků a odpovědí je dostupná na adrese: `http://docs.ffmap1.apiary.io/`.

### 5.2.2 Databáze a ORMLite

Součástí platformy Android je také databáze SQLite, která je upravena pro mobilní zařízení (viz podkapitola 3.3.3). K této databázi se nepřipojuje pomocí rozhraní JDBC, jak je běžné v aplikacích napsaných v jazyce Java. Ke správě databáze je určena speciální třída jménem `SQLiteOpenHelper`[8].

Pro manipulaci s daty jsem využil aplikační rámec pro objektově-relační mapování ORMLite (viz podkapitola 5.2.5). ORMLite využívá k vytváření, mapování a manipulaci s daty anotované třídy v jazyce Java. Lze použít jak anotace z balíku `javax.persistence`, tak

4. `http://developer.android.com/reference/java/net/HttpURLConnection.html`
5. `http://json.org/`
6. `http://jackson.codehaus.org/1.6.2/javadoc/org/codehaus/jackson/map/ObjectMapper.html`
7. `https://apiary.io/`
8. `http://developer.android.com/reference/android/database/sqlite/SQLiteOpenHelper.html`

i vlastní anotace knihovny ORMLite, které jsou doporučené. Namapované objekty jsou instancemi POJO[9] tříd.

Do verze Honeycomb včetně byla v kódu Androidu kritická chyba[10], která způsobovala neúměrně dlouhé zpracování anotací. V novějších verzích Androidu (od verze API 14) je tato chyba údajně opravena, ačkoli dle některých testů problém není stále zcela opraven ani ve verzi 4.0.4 [51].

ORMLite proto používá alternativní způsob zpracování anotací tříd pomocí vlastního nástroje. Ten lze spustit jako klasický program napsaný v jazyce Java, přičemž ten vygeneruje konfigurační soubor `ormlite_config.txt` v adresáři `/res/raw/`. Pokud v konstruktoru příslušné konfigurační třídy specifikován konfigurační soubor, ORMLite přeskočí zpracování anotací tříd.

### 5.2.3 Určení lokace a práce s mapou

Pro určení geografické lokace zařízení slouží třída `LocationManager`[11]. Pomocí třídy `LocationManager` lze získat jak poslední známou polohu zařízení, tak dostávat informace o jeho poloze v pravidelných intervalech. Příklad práce s geografickou polohou uvádí zdrojový kód 5.1.

Zdrojový kód 5.1: Práce s geografickou polohou

```
1 LocationManager locationManager = (LocationManager)
2   getSystemService(Context.LOCATION_SERVICE);
3 Location location = locationManager.getLastKnownLocation(
4     locationManager.getBestProvider(new Criteria, false)
5 );
6 locationManager.requestLocationUpdates(
7   locationManager.getBestProvider(criteria, false), 0, 0,
8     this
9 );
```

Android také poskytuje rozhraní pro přístup ke službě Google Maps. K tomu je možné využít Google Maps Android API v2[12].

9. z angl. *Plain Old Java Object* – tedy objekty, které nemusí implementovat žádné speciální rozhraní
10. `https://code.google.com/p/android/issues/detail?id=7811`
11. `http://developer.android.com/reference/android/location/LocationManager.html`
12. `https://developers.google.com/maps/documentation/android/`

Pro přístup k mapám je možné využít buďto připravený fragment `MapFragment` nebo grafickou komponentu `MapView`.

Referenci k objektu mapy je možné získat pomocí zpětného volání metody `getMapAsync`. Ta po přichystání a vykreslení mapy zavolá metodu `onMapReady` s připravenou instancí třídy `GoogleMap`. Do mapy je možné zakreslovat objekty pomocí tříd `Marker` a `MarkerOptions`. Práci s mapou ukazuje ukázka kódu 5.2.

Zdrojový kód 5.2: Vykreslení objektu na mapě

```
1 getMapAsync(new OnMapReadyCallback() {
2   public void onMapReady(GoogleMap googleMap) {
3     MarkerOptions options = new MarkerOptions();
4     options.draggable(true);
5     options.position(latLng);
6     googleMap.addMarker(options);
7   }
8 }
```

### Klastrování objektů na mapě

Pro pokročilou práci s mapami existuje knihovna Google Maps Android API utility library[13]. Díky ní lze vykreslovat velké počty objektů na mapě ve formě klastrů, tak aby vypadala přehledně a zbytečně nemátla uživatele.

Objekty zobrazované na mapě musí implementovat rozhraní `ClusterItem`[14], které předepisuje metodu `getPosition()`, která vrací polohu objektu v souřadnicích GPS. Namísto vykreslování objektů přímo mapou je možné je vykreslovat pomocí třídy `ClusterManager`. Práci s klastry ukazuje ukázka kódu 5.3.

Zdrojový kód 5.3: Klastrování objektů na mapě

```
1 ClusterManager clusterManager =
2   new ClusterManager<Joint>(getActivity(), map);
3 clusterManager.addItem(joint);
4 clusterManager.cluster();
```

13. http://googlemaps.github.io/android-maps-utils/
14. http://googlemaps.github.io/android-maps-utils/javadoc/

### Získání adresy objektu

V rámci Androidu je možné využít aplikační rámec pro převod adresy na geografické souřadnice a zpět pomocí třídy `Geocoder`[15]. Proces získání adresy z geografické souřadnice se nazývá *reverse geocoding*.

Příklad získání adresy ze souřadnice ukazuje zdrojový kód 5.4.

Zdrojový kód 5.4: Získání adresy pomocí třídy `Geocoder`

```
1  Geocoder geocoder = new Geocoder(this);
2  List<Address> addresses =
3    geocoder.getFromLocation(latitude, longitude, 1);
4
5  if (null != addresses && 0 < addresses.size()) {
6    Address address = addresses.get(0);
7
8    for (int i = 0; i < address.getMaxAddressLineIndex();
9      i++
10   ) {
11     String addressLine = address.getAddressLine(i);
12   }
13 }
```

### 5.2.4 Zpracování velkých bitmap

Každá aplikace pro Android může v čase využívat jen omezené množství paměti. Při překročení hraniční hodnoty dojde k vyhození nehlídané výjimky `OutOfMemoryException`, která obvykle vede k pádu aplikace. Konkrétní hodnoty se typicky pohybují mezi 16 a 64 MB a liší se dle použitého zařízení. Toto omezení je obzvláště nepříjemné při zobrazování velkých bitmap (např. fotografií). Například fotoaparát zařízení Galaxy Nexus pořizuje fotografie v rozlišení $2592 \times 1936$ pixelů, které při čtyřbajtovém kódování (jeden bajt pro každou ze základních barev a jeden bajt pro tzv. alfa kanál určující průhlednost) zabírá lehce přes 19 MB paměti.

Pouhý pokus o načtení fotografie na tomto zařízení do instance vestavěné třídy `Bitmap` by skončil právě výše zmíněnou výjimkou. Pro bezpečné a efektivní načítání bitmap je možné využít vestavě-

15. `https://developer.android.com/training/location/display-address.html`

nou třídu `BitmapFactory` z balíku `android.graphics`[16]. Dekódování bitmapy se sestává ze tří kroků.

Prvním krokem je zjištění celkových rozměrů obrázku. To je možné pomocí třídy `BitmapFactory.Options`, která má veřejný atribut `inJustDecodeBounds`, který pokud je nastaven na `true`, tak `BitmapFactory` přeskočí načítání bitmapy do paměti, ale pouze nastaví atributy `outWidth` a `outHeight` třídy `BitmapFactory.Options`.

Zdrojový kód 5.5: Určení velikosti obrázku

```
1 BitmapFactory.Options options
2   = new BitmapFactory.Options();
3 options.inJustDecodeBounds = true;
4 BitmapFactory.decode(filePath, options);
```

Druhým krokem je zjištění velikosti vzorkování. K tomu jsou potřeba rozměry komponenty uživatelského rozhraní, které se získají standardně pomocí metod `View#getWidth()` a `View#getHeight()`. Výpočet vzorkování probíhá dle jednoduchého algoritmu, který zdvojnásobuje velikost vzorkování a zároveň půlí rozměry obrázku, dokud alespoň jeden z rozměrů převyšuje velikost uvažovaného plátna (viz zdrojový kód 5.6).

16. `http://developer.android.com/reference/android/graphics/BitmapFactory.html`

Zdrojový kód 5.6: Výpočet vzorkování

```
1  public calculateInSampleSize(
2      BitmapFactory.Options options, int reqWidth,
3      int reqHeight
4  ) {
5    final int height = options.outHeight;
6    final int width = options.outWidth;
7    int inSampleSize = 1;
8    if (height > reqHeight || width > reqWidth) {
9      int halfHeight = height / 2;
10     int halfWidth = width / 2;
11
12     while ((halfHeight / inSampleSize) > reqHeight &&
13            (halfWidth / inSampleSize) > reqWidth)
14         {
15       inSampleSize *= 2;
16     }
17   }
18   return inSampleSize;
19 }
```

Třetím krokem je načtení samotné bitmapy dle vypočtené velikosti vzorkování a její následné zobrazení komponentě `ImageView`.

Zdrojový kód 5.7: Načtení bitmapy a zobrazení v `ImageView`

```
1 BitmapFactory.Options options =
2   new BitmapFactory.Options();
3 options.inSampleSize = sampleSize;
4 Bitmap bitmap = decodeFile(filePath, options);
5 imageView.setImageBitmap(bitmap);
```

`BitmapFactory` *není* kompatibilní s obrázky ve formátu JPEG se subtraktivním míchání barev (kódování CMYK). Toto ale není výraznější překážkou, protože fotoaparát v systému Android ukládá implicitně fotografie s aditivním mícháním barev (kódování RGB) [38].

### 5.2.5 Použité knihovny

#### Jackson JSON Processor

Jackson je knihovna v jazyce Java pro serializaci a deserializaci objektů ve formátu JSON (JavaScript Object Notation).

V roce 2011 provedl Martin Adámek sérii srovnání parserů formátu JSON na platformě Android, přičemž Jackson se prokázal jako nejvýkonnější z testovaných alternativ [2, 1, 3].

### ORMLite

ORMLite[17] je knihovna pro objektově-relační mapování napsána v jazyce Java americkým vývojářem Grayem Watsonem. Kromě podpory několika tradičních relačních databázových systémů pomocí rozhraní JDBC, ORMLite podporuje také SQLite na platformě Android. ORMLite je distribuován jako open-source pod licencí ISC.

### Google Play Services

Google Play Services je knihovna pro přístup k webovým službám provozovaným společností Google[18]. Pomocí Google Play Services je možné se připojit k aplikačnímu programovému rozhraní map (viz podkapitola 5.2.3).

### Google Maps Android API utility library

Google Map Android API utility library je rozšiřující knihovna pro vykreslování objektů v mapách. Google Maps Android API utility library je distribuovaná pod licencí Apache Licence 2.0.

### ActionBarSherlock a ViewPagerIndicator

ActionBarSherlock[19] a ViewPagerIndicator[20] jsou knihovny vyvinuté v roce 2012 americkým vývojářem Jakem Whartonem. Tyto knihovny poskytují podporu některých komponent a grafických prvků představených ve verzi 3.0 pro starší verze Androidu. Obě knihovny jsou distribuovány pod licencí Apache Licence 2.0.

---

17. `http://ormlite.com/`
18. `https://developer.android.com/google/play-services/index.html`
19. `http://actionbarsherlock.com/`
20. `http://viewpagerindicator.com/`

## 5.3 Server

Jako technologii pro vývoj serverové aplikace Hladových oken jsem použil programovací jazyk Java a platformu Java Enterprise Edition. Hlavní výhodou tohoto řešení je možnost znovuvyužitelnosti kódu při výměně dat mezi serverem a klientem (viz podkapitola 5.2.1).

Každá aplikace, která je hostovaná na OpenShiftu je zároveň i repozitář verzovacího systému git[21]. Pro nahrání nové verze aplikace na server je potřeba aplikaci nejprve zkompilovat na lokálním počítači a výsledný `.war` soubor umístit do adresáře `deployments`, který se nachází v kořenovém adresáři projektu.

Po nahrání změn do centrálního repozitáře je OpenShift nejprve ověří, zda se aplikace skutečně zkompiluje a spustí případné testy, jsou-li v aplikaci přítomny. Pokud vše proběhne bez chyb, zavede OpenShift všechny `.war` soubory z adresáře `deployments` do aplikačního serveru JBoss.

Pro automatizaci tohoto procesu existuje také nástroj pro vývojové prostředí Eclipse[22].

Nasazený server je k dispozici na adrese `http://ffmap-drekorian.rhcloud.com/`.

### 5.3.1 Použité knihovny

#### Enterprise JavaBeans

Enterprise JavaBeans jsou řízené serverové komponenty, které se často používají v serverových aplikacích napsaných v jazyce Java. Aplikační server dokáže instanciovat JavaBeans podle počtu požadavků a pomocí Context Dependency Injection vkládat instance do kódu aplikace.

21. `http://git-scm.com/`
22. `https://blog.openshift.com/day-28-openshift-eclipse-integration-for-java-developers/`

#### Hibernate

Hibernate je plnou implementací rozhraní JPA (Java Persistence API), kterou lze použít pro práci s řízenými datovými zdroji velice dobře spolupracuje s řízenými datovými zdroji serveru.

#### RESTEasy

RESTEasy je implementace rozhraní JAX-RS pro webové služby typu REST. RESTEasy umožňuje komunikaci ve formátu JSON pomocí integrace s parserem Jackson.

### 5.3.2 Nasazení na platformě OpenShift

Pro vzorovou implementaci jsem využil službu PaaS (z angl. *Platform as a Service*, doslova platforma jako služba) OpenShift[23] společnosti Red Hat. OpenShift umožňuje zdarma nahrání až tři aplikací pro jeden účet.

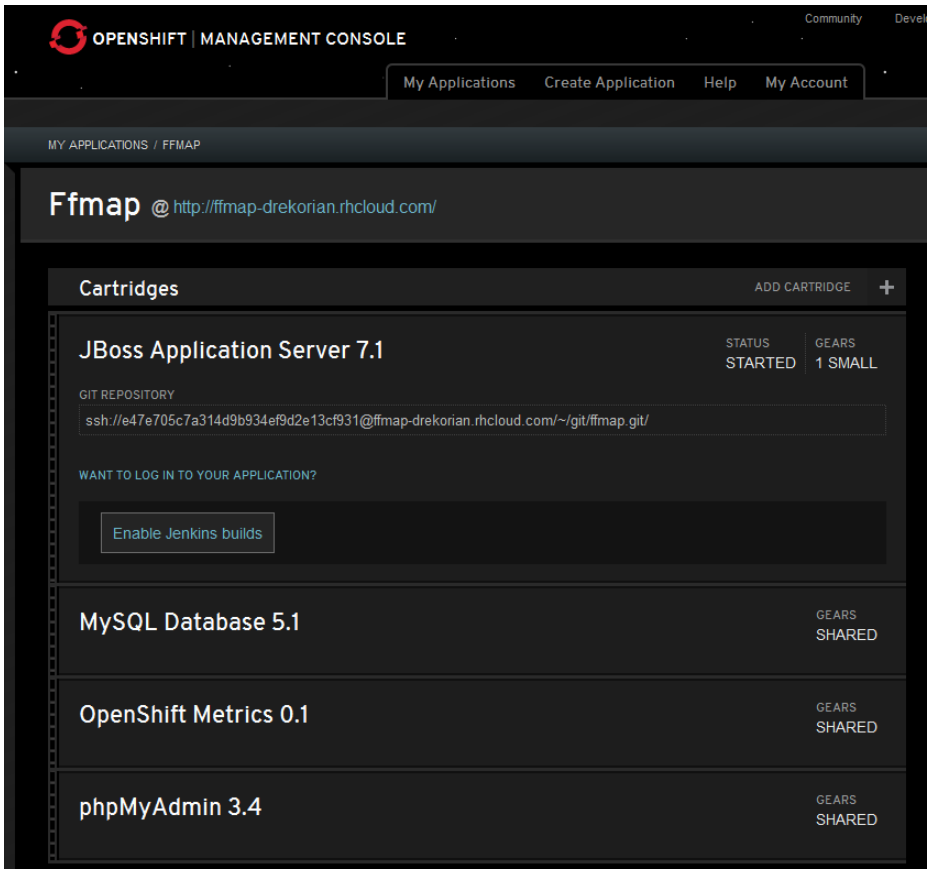
Vývojář si může zvolit z několika tzv. *cartridges*, předpřipravených zásuvných modulů s předinstalovanými interprety a aplikacemi pro virtuální server. Vývojáři, kteří si z nabídky zásuvných modulů nevyberou, si mohou vytvořit i zásuvný modul vlastní.

Serverová verze Hladových oken využívá čtyři cartridge – pro aplikační server JBoss, databázi MySQL, aplikaci Openshift Metrics a konečně pro aplikaci phpMyAdmin.

#### Cartridge JBoss Application Server 7.1

JBoss AS (JavaBeans Open Source Software Application Server) je aplikační server, který implementuje platformu Java Enterprise Edition, vyvíjený společností Red Hat a distribuovaný pod licencí GNU LGPL. Tato cartridge poskytuje tzv. webový profil Java EE, což je pro Hladová okna postačující. Cartridge s tzv. plným profilem je pro OpenShift také dostupná.

23. `https://openshift.redhat.com/app/`




Obrázek 5.5: OpenShift Management Console
Zdroj: autor

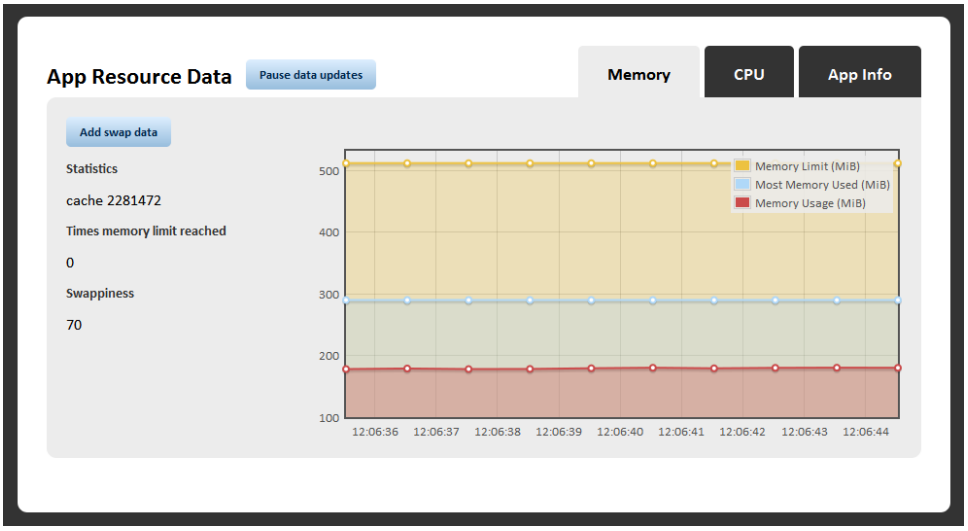
### Cartridge MySQL Database 5.1

MySQL je relační databázový systém vyvíjený firmou Oracle Corporation. Databáze se integruje s aplikačním serverem jako řízený datový zdroj (managed datasource) a je v aplikaci dostupná pomocí adresovacího rozhraní JNDI.

Pro OpenShift jsou dostupné i zásuvné moduly pro databáze PostgreSQL a nerelační databázi MongoDB.

### Cartridge OpenShift Metrics 0.1

OpenShift Metrics je jednoduchá webová aplikace, která ukazuje aktuální zatížení serveru z pohledu strojového času, paměti a swapování v reálném čase (viz obrázek 5.6). Aktuální výsledky aplikace je možné si prohlédnout na adrese `https://ffmap-drekorian.rhcloud.com/metrics/`.




Obrázek 5.6: Ukázka měření zatížení aplikace v OpenShift Metrics
Zdroj: autor

### Cartridge phpMyAdmin 3.4

PhpMyAdmin[24] je webová aplikace napsaná v PHP, která umožňuje správu databází MySQL. PhpMyAdmin je dostupný pod licencí GNU GPL.

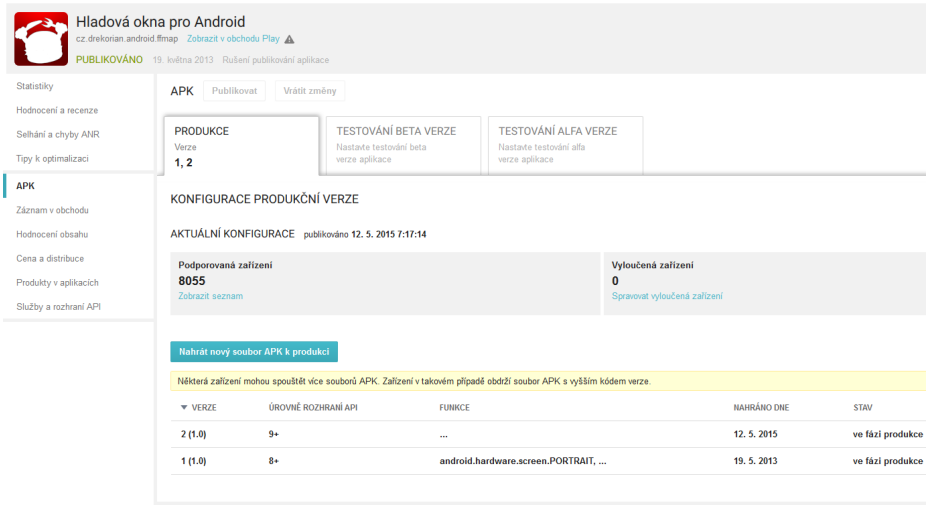
24. `http://www.phpmyadmin.net/`

# 6 Nasazení

Pro nasazení jsem vybral nejrozšířenější tržiště aplikací pro platformu Android – tržiště Google Play (viz podkapitola 3.5.1).

Před publikováním je nutné aplikace podepsat. To je možné buďto ručně pomocí nástrojů `keytool` a `jarsigner`[1], které jsou dodávány spolu s Java Development Kitem nebo aplikaci podepsat pomocí vývojového prostředí automaticky.

Podepsané APK je možné nahrát skrze webové stránky Google Play Developer Console[2].




Obrázek 6.1: Správa souborů APK v Google Play Developer Console
Zdroj: autor

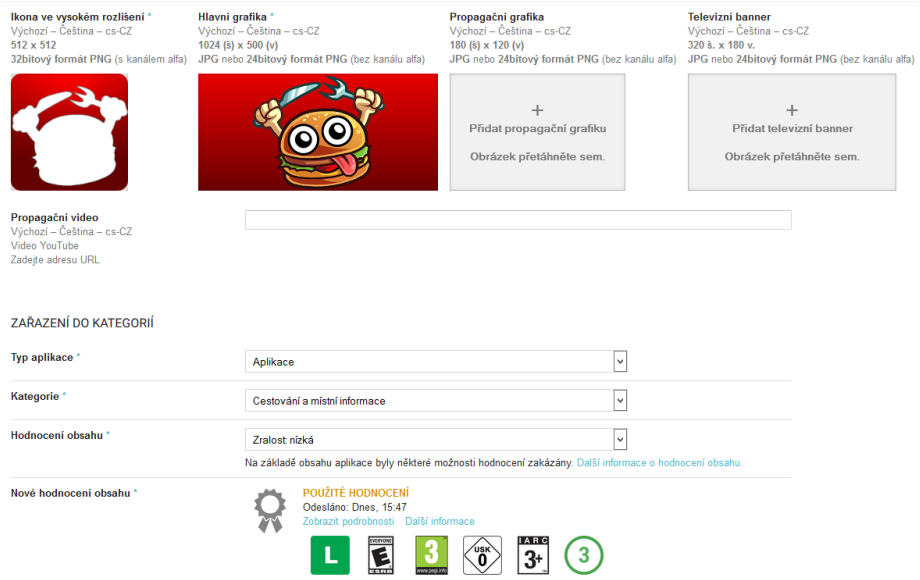
K publikaci aplikace jsou nutné nejméně dva obrázky ilustrující aplikaci, ikona ve vysokém rozlišení (512 × 512 pixelů) a tzv. hlavní grafika (obrázek o rozměrech 1024 × 500 pixelů).

Nově je také nutné vyplnit dotazník, který kalkuluje známku hodnocení obsahu dle standardů ratingových systémů jednotlivých zemí (např. pro evropské země je to hodnocení PEGI[3]).

---

1. `http://developer.android.com/tools/publishing/app-signing.html`
2. `https://play.google.com/apps/publish/`
3. `http://www.pegi.info/`

V Google Play Developer Console lze také ručně znepřístupnit publikaci v určitých zemích či pro konkrétní modely mobilních zařízení.




Obrázek 6.2: Grafika v Google Play Developer Console
Zdroj: autor

Po potvrzené publikaci je aplikace nasazena. Nasazení může trvat až několik hodin.

Nasazená aplikace je přístupná ve službě Google Play na adrese: `https://play.google.com/store/apps/details?id=cz.drekorian.android.ffmap`.

# 7 Závěr

V rámci bakalářské práce byla navržena a implementována aplikace pro platformu Android umožnující stahování databáze provozoven rychlého občerstvení z Internetu, její prohlížení a úpravy uživatelem.

Aplikace taktéž umí zodpovídat dotazy uživatele na základě jeho polohy zjištěné automaticky. Aplikace byla otestována jak na emulátoru, tak na fyzických telefonech, při čemž bylo ověřeno, že její běh je plynulý a nevyvolává pády aplikace.

Aplikace je připravena pro případná budoucí rozšíření.

# Literatura


[1] Adámek, M.: Adding Gson to Android JSON Parser Comparison. 2011, dostupné online [poslední revize 17. 5. 2015] http://martinadamek.com/2011/02/01/adding-gson-to-android-json-parser-comparison/.

[2] Adámek, M.: Comparison of JSON Parsers Performance on Android. 2011, dostupné online [poslední revize 17. 5. 2015] http://martinadamek.com/2011/01/31/comparison-of-json-parsers-performance-on-android/.

[3] Adámek, M.: JSON Parsers Performance on Android (With Warmup and Multiple Iterations). 2011, dostupné online [poslední revize 17. 5. 2015] http://martinadamek.com/2011/02/04/json-parsers-performance-on-android-with-warmup-and-multiple-iterations/.

[4] Bada Developers: Architecture of bada. 2011, dostupné online [poslední revize 17. 5. 2015] http://developer.bada.com/help_2.0/topic/com.osp.documentation.help/html/bada_overview/bada_architecture.htm.

[5] Berg, A.: The Complexities, Compromises of Battery Life. 2011, dostupné online [poslední revize 17. 5. 2015] http://www.wirelessweek.com/articles/2011/06/devices-complexities-compromises-battery-life-smartphones/.

[6] Bielinis, S.: Samsung Bada is a dud. Less than 10 million shipped, sales 50below expectations and falling. 2011, dostupné online [poslední revize 17. 5. 2015] http://www.unwiredview.com/2011/09/26/samsung-bada-is-a-dud-less-then-10-million-shipped-sales-50-below-expectations-and-falling/.

[7] Böhmer, M.; Hecht, B.; Schöning, J.; aj.: Falling Asleep with Angry Birds, Facebook and Kindle: A Large Scale Study on

Mobile Application Usage. In *Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services*, New York, NY, USA: ACM, 2011, ISBN 978-1-4503-0541-9, s. 47–56, dostupné online [poslední revize 17. 5. 2015]
http://doi.acm.org.ezproxy.muni.cz/10.1145/2037373.2037383.

[8] Carmody, T.: Fighting words: Apple's 'Post-PC' and Microsoft's 'PC Plus' were never that different. 2012, dostupné online [poslední revize 17. 5. 2015]
http://www.theverge.com/apple/2012/7/12/3151491/fighting-words-apple-post-pc-microsoft-pc-plus.

[9] Dediu, H.: How many Android phones have been activated? (Updated). 2011, dostupné online [poslední revize 17. 5. 2015]
http://www.asymco.com/2011/12/21/how-many-android-phones-have-been-activated/.

[10] Eclipse: Eclipse Indigo (3.7) Documentation. 2012, dostupné online [poslední revize 17. 5. 2015]
http://help.eclipse.org/indigo/index.jsp.

[11] Gartner: Gartner Says Worldwide Mobile Phone Sales Increased 14 Per Cent in First Quarter of 2008. 2008, dostupné online [poslední revize 17. 5. 2015]
http://www.gartner.com/newsroom/id/680207.

[12] Gartner: Gartner Says Worldwide Smartphone Sales Grew 16 Per Cent in Second Quarter of 2008. 2008, dostupné online [poslední revize 17. 5. 2015]
http://www.gartner.com/newsroom/id/754112.

[13] Gartner: Gartner Says Worldwide Smartphone Sales Reached Its Lowest Growth Rate With 11.5 Per Cent Increase in Third Quarter of 2008. 2008, dostupné online [poslední revize 17. 5. 2015]
http://www.gartner.com/newsroom/id/827912.

[14] Gartner: Gartner Says Grey-Market Sales and Destocking Drive Worldwide Mobile Phone Sales to 309 Million Units; Smartphone Sales Grew 13 Per Cent in Third Quarter of 2009. 2009, dostupné online [poslední revize 17. 5. 2015] http://www.gartner.com/newsroom/id/1224645.

[15] Gartner: Gartner Says Worldwide Mobile Phone Sales Declined 6 Per Cent and Smartphones Grew 27 Per Cent in Second Quarter of 2009. 2009, dostupné online [poslední revize 17. 5. 2015] http://www.gartner.com/newsroom/id/1126812.

[16] Gartner: Gartner Says Worldwide Mobile Phone Sales Declined 8.6 Per Cent and Smartphones Grew 12.7 Per Cent in First Quarter of 2009. 2009, dostupné online [poslední revize 17. 5. 2015] http://www.gartner.com/newsroom/id/985912.

[17] Gartner: Gartner Says Worldwide Smartphone Sales Reached Its Lowest Growth Rate With 3.7 Per Cent Increase in Fourth Quarter of 2008. 2009, dostupné online [poslední revize 17. 5. 2015] http://www.gartner.com/newsroom/id/910112.

[18] Gartner: Gartner Says Worldwide Mobile Device Sales Grew 13.8 Percent in Second Quarter of 2010, But Competition Drove Prices Down. 2010, dostupné online [poslední revize 17. 5. 2015] http://www.gartner.com/newsroom/id/1421013.

[19] Gartner: Gartner Says Worldwide Mobile Phone Sales Grew 17 Per Cent in First Quarter 2010. 2010, dostupné online [poslední revize 17. 5. 2015] http://www.gartner.com/newsroom/id/1372013.

[20] Gartner: Gartner Says Worldwide Mobile Phone Sales Grew 35 Percent in Third Quarter 2010; Smartphone Sales Increased 96 Percent. 2010, dostupné online [poslední revize 17. 5. 2015] http://www.gartner.com/newsroom/id/1466313.

[21] Gartner: Gartner Says Worldwide Mobile Phone Sales to End Users Grew 8 Per Cent in Fourth Quarter 2009; Market Remained Flat in 2009. 2010, dostupné online [poslední revize

17. 5. 2015]
http://www.gartner.com/newsroom/id/1306513.

[22] Gartner: Gartner Says 428 Million Mobile Communication Devices Sold Worldwide in First Quarter 2011, a 19 Percent Increase Year-on-Year. 2011, dostupné online [poslední revize 17. 5. 2015]
http://www.gartner.com/newsroom/id/1764714.

[23] Gartner: Gartner Says Sales of Mobile Devices Grew 5.6 Percent in Third Quarter of 2011; Smartphone Sales Increased 42 Percent. 2011, dostupné online [poslední revize 17. 5. 2015]
http://www.gartner.com/newsroom/id/1848514.

[24] Gartner: Gartner Says Sales of Mobile Devices in Second Quarter of 2011 Grew 16.5 Percent Year-on-Year; Smartphone Sales Grew 74 Percent. 2011, dostupné online [poslední revize 17. 5. 2015]
http://www.gartner.com/newsroom/id/1689814.

[25] Gartner: Gartner Says Worldwide Mobile Device Sales to End Users Reached 1.6 Billion Units in 2010; Smartphone Sales Grew 72 Percent in 2010. 2011, dostupné online [poslední revize 17. 5. 2015]
http://www.gartner.com/newsroom/id/1543014.

[26] Gartner: Gartner Says Worldwide Sales of Mobile Phones Declined 2 Percent in First Quarter of 2012; Previous Year-over-Year Decline Occurred in Second Quarter of 2009. 2012, dostupné online [poslední revize 17. 5. 2015]
http://www.gartner.com/newsroom/id/2017015.

[27] Gartner: Gartner Says Worldwide Sales of Mobile Phones Declined 2.3 Percent in Second Quarter of 2012. 2012, dostupné online [poslední revize 17. 5. 2015]
http://www.gartner.com/newsroom/id/2120015.

[28] Gartner: Gartner Says Worldwide Sales of Mobile Phones Declined 3 Percent in Third Quarter of 2012; Smartphone Sales Increased 47 Percent. 2012, dostupné online [poslední revize

17. 5. 2015]
http://www.gartner.com/newsroom/id/2237315.

[29] Gartner: Gartner Says Worldwide Smartphone Sales Soared in Fourth Quarter of 2011 With 47 Percent Growth. 2012, dostupné online [poslední revize 17. 5. 2015]
http://www.gartner.com/newsroom/id/1924314.

[30] Gartner: Gartner Says Asia/Pacific Led Worldwide Mobile Phone Sales to Growth in First Quarter of 2013. 2013, dostupné online [poslední revize 17. 5. 2015]
http://www.gartner.com/newsroom/id/2482816.

[31] Gartner: Gartner Says Smartphone Sales Accounted for 55 Percent of Overall Mobile Phone Sales in Third Quarter of 2013. 2013, dostupné online [poslední revize 17. 5. 2015]
http://www.gartner.com/newsroom/id/2623415.

[32] Gartner: Gartner Says Smartphone Sales Grew 46.5 Percent in Second Quarter of 2013 and Exceeded Feature Phone Sales for First Time. 2013, dostupné online [poslední revize 17. 5. 2015]
http://www.gartner.com/newsroom/id/2573415.

[33] Gartner: Gartner Says Worldwide Mobile Phone Sales Declined 1.7 Percent in 2012. 2013, dostupné online [poslední revize 17. 5. 2015]
http://www.gartner.com/newsroom/id/2335616.

[34] Google: Android Developer Tools. 2012, dostupné online [poslední revize 17. 5. 2015]
http://developer.android.com/guide/developing/tools/adt.html.

[35] Google: Android Developers: Licenses. 2012, dostupné online [poslední revize 17. 5. 2015]
http://source.android.com/source/licenses.html.

[36] Google: What is Android? 2012, dostupné online [poslední revize 17. 5. 2015]
http://developer.android.com/guide/basics/what-is-android.html.

[37] Google: Android Developers: Platform versions. 2013, dostupné online [poslední revize 17. 5. 2015]
http://developer.android.com/resources/dashboard/platform-versions.html.

[38] Google: Android Developers: Displaying Bitmaps Efficiently. 2015, dostupné online [poslední revize 17. 5. 2015]
http://developer.android.com/training/displaying-bitmaps/index.html.

[39] Google: Podporované lokality pro registraci vývojářů a obchodníků. 2015, dostupné online [poslední revize 17. 5. 2015]
https://support.google.com/googleplay/android-developer/table/3539140.

[40] HowStuffWorks.com: Why are the keys arranged the way they are on a QWERTY keyboard? 2000, dostupné online [poslední revize 17. 5. 2015]
http://computer.howstuffworks.com/question458.htm.

[41] ITU: Global mobile-cellular subscriptions, total and per 100 inhabitants, 2001-2013. 2011, dostupné online [poslední revize 17. 5. 2015]
http://www.itu.int/en/ITU-D/Statistics/Documents/statistics/2013/ITU_Key_2005-2013_ICT_data.xls.

[42] ITU: Key ICT indicators for developed and developing countries and the world (totals and penetration rates). 2011, dostupné online [poslední revize 17. 5. 2015]
http://www.itu.int/en/ITU-D/Statistics/Documents/statistics/2012/stat_page_all_charts.xls.

[43] ITU: ICT Facts and Figures. 2013, dostupné online [poslední revize 17. 5. 2015]
http://www.itu.int/en/ITU-D/Statistics/Documents/facts/ICTFactsFigures2013-e.pdf.

[44] Lupa.cz: Android Market: placené aplikace i pro české uživatele. 2010, dostupné online [poslední revize 17. 5. 2015] http://www.lupa.cz/zpravicky/android-market-placene-aplikace-i-pro-cr/.

[45] Mednieks, Z.; Dornin, L.; Meike, G.; aj.: *Programming Android, Second Edition*. O'Reilly, 2012, ISBN 978-1-449-31664-8, 90 s.

[46] Meier, R.: *Professional Android 2 Application Development*. Indianapolis, Indiana: Wiley Publishing, 2010, ISBN 9780470874516, 35 s.

[47] Meier, R.: *Professional Android 2 Application Development*. Wiley, 2010, ISBN 9780470637456, 480 s., Chapter 1: Hello, Android, s. 4-5.

[48] Murphy, M. L.: *Beginning Android*. New York: Apress, 2009, ISBN 9781430224198, 384 s., Chapter 9: Getting Fancy with Lists, s. 76-80.

[49] Pachal, P.: Apple vs. All the Rest: Every Mobile App Store Compared. 2011, dostupné online [poslední revize 17. 5. 2015] http://www.pcmag.com/article2/0,2817,2382944,00.asp.

[50] Perlow, J.: Post-PC era means mass extinction for personal computer OEMs. 2012, dostupné online [poslední revize 17. 5. 2015] http://www.zdnet.com/blog/perlow/post-pc-era-means-mass-extinction-for-personal-computer-oems/20514.

[51] Post, M.: Mythbuster: Android annotation performance unravelled. 2012, dostupné online [poslední revize 17. 5. 2015] http://eclipsesource.com/blogs/2012/09/28/mythbuster-android-annotation-performance-unravelled/.

[52] Preston-Werner, T.: Semantic Versioning 2.0.0. 2013, dostupné online [poslední revize 17. 5. 2015] http://semver.org/.

[53] Ujbányai, M.: *Programujeme pro Android*. Praha: Grada, 2012, ISBN 978-80-247-3995-3, Kapitola 1: Android – představení, s. 13-17.

[54] Ujbányai, M.: *Programujeme pro Android*. Praha: Grada, 2012, ISBN 978-80-247-3995-3, Kapitola 1: Android – představení, s. 23.

[55] Velazco, C.: Samsung To Make Bada OS Open Source And Part Of Your Smart TV? 2011, dostupné online [poslední revize 17. 5. 2015]
`http://techcrunch.com/2011/09/20/samsung-to-make-bada-os-open-source-and-part-of-your-smart-tv/`.

[56] Vogel, L.: Android BroadcastReceiver - Tutorial. 2013, dostupné online [poslední revize 17. 5. 2015]
`http://www.vogella.com/tutorials/AndroidBroadcastReceiver/article.html#broadcastreceiver_platformfilter`.

# A Přílohy

- CD obsahující:
  - zdrojové kódy serveru a klientské aplikace
  - diagramy ve formátu Visual Paradigm
  - výsledky průzkumu provedeného mezi potenciálními uživateli