UNIVERZITA PARDUBICE
Fakulta elektrotechniky a informatiky

Bluetooth vysílačka pro chytrý telefon

Michal Svoboda

Diplomová práce
2013

Univerzita Pardubice
Fakulta elektrotechniky a informatiky
Akademický rok: **2012/2013**

ZADÁNÍ DIPLOMOVÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Bc. Michal Svoboda**
Osobní číslo:	**I11411**
Studijní program:	**N2646 Informační technologie**
Studijní obor:	**Informační technologie**
Název tématu:	**Bluetooth vysílačka pro chytrý telefon**
Zadávající katedra:	**Katedra softwarových technologií**

Zásady pro vypracování:

Cílem diplomové práce je vytvoření aplikace pro chytrý telefon, která bude sloužit k obousměrnému přenosu hlasu mezi telefony, a to prostřednictvím technologie Bluetooth.
V úvodní části DP bude proveden rozbor technologie bluetooth (její možnosti, limity), rozbor operačních systémů pro chytré telefony ve vztahu k použití technologie bluetooth a dále bude provedena rešerše dostupných aplikací na trhu, které tuto problematiku řeší.
V aplikační části práce bude proveden návrh architektury aplikace a její vlastní implementace pro zvolený operační systém.

Rozsah grafických prací:

Rozsah pracovní zprávy:

Forma zpracování diplomové práce: **tištěná/elektronická**

Seznam odborné literatury:

MARK L. MURPHY. Android 2: Průvodce programováním mobilních aplikací, COMPUTER PRESS, 2011, 376 s., ISBN 978-80-251-3194-7
ING. MIROSLAV UJBÁNYAI. Programujeme pro Android. GRADA, 2012, 192 s., ISBN 978-80-247-3995-3
TKÁČ JOSEF. Jak na bluetooth v rekordním čase. GRADA, 2005, 84 s. ISBN 80-247-1081-1

Vedoucí diplomové práce: **Ing. Lukáš Čegan, Ph.D.**
Katedra informačních technologií

Datum zadání diplomové práce: **31. října 2012**
Termín odevzdání diplomové práce: **17. května 2013**

Univerzita Pardubice Česká republika

L.S.

prof. Ing. Simeon Karamazov, Dr.
děkan

prof. Ing. Antonín Kavička, Ph.D.
vedoucí katedry

V Pardubicích dne 15. listopadu 2012

Prohlášení autora

Prohlašuji, že jsem tuto práci vypracoval samostatně. Veškeré literární prameny a informace, které jsem v práci využil, jsou uvedeny v seznamu použité literatury.

Byl jsem seznámen s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorský zákon, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Souhlasím s prezenčním zpřístupněním své práce v Univerzitní knihovně.

V Pardubicích dne 23. 8. 2013 Michal Svoboda

Poděkování

Rád bych na tomto místě poděkoval svým blízkým, za projevenou podporu během studia. Dále bych rád poděkoval vedoucímu mé diplomové práce, panu Ing. Lukáši Čeganovi, Ph.D., za užitečné poznatky, rady, a čas, který mi věnoval.

Anotace

Diplomová práce je zaměřena na popis mobilních platforem a obousměrný přenos hlasu pomocí technologie Bluetooth.

Klíčová slova

Android, mobilní zařízení, vysílačka, Google Maps, Bluetooth, GPS

Title

Bluetooth transmitter for smart phone

Annotation

Diploma thesis is focused on description of mobile platforms and two-way voice transmission using Bluetooth technology.

Keywords

Android, mobile device, transmitter, Google Maps, Bluetooth, GPS

Obsah

Seznam zkratek

BT	Bluetooth
OS	Operační Systém
PAN	Personal Area Network
JVM	Java Virtual Machine
DVM	Dalvik Virtual Machine
SDK	Software Development Kit
ADK	Android Software Development Kit
DVM	Dalvik Vietual Machine
JVM	Java Virtual Machine
JSE	Java Standard Edition
AWT	Abstract Window Toolkit
ADT	Android Development Tools
AVD	Android Virtual Device

Seznam obrázků

Seznam tabulek

Úvod

Diplomová práce je zaměřena na problematiku přenosu hlasu mezi dvěma mobilními zařízeními prostřednictvím Bluetooth technologie. Cílem je vytvořit aplikaci pro chytrý telefon, která bude umožňovat plně duplexní přenos hlasu prostřednictvím technologie Bluetooth.

Účelově je tato aplikace vyvíjena pro motorkáře s cílem nahradit klasický interkom. Výsledek by však měl být univerzální. Sice bude mít funkce využitelné při komunikaci na motocyklu, bude se ale dát použít i v jiných oblastech. Měla by tedy sloužit při komunikaci mezi jezdcem a spolujezdcem, ale rovněž ji bude možné využít jako klasickou vysílačku na krátkou vzdálenost.

V úvodní části bude provedena rešerše, ve které se čtenář dozví něco o dosavadních řešeních přenosu hlasu, ať už prostřednictvím profesionálních interkomů, nebo prostřednictvím aplikace pro chytrý telefon. Budou zde popsány i konkrétní aplikace včetně mobilních platforem, pro které jsou určeny. Závěrem této části bude vzájemné srovnání jednotlivých řešení s aplikací, která je vyvíjena v rámci této práce. Následně budou popsány technologie, které byly použity během vývoje použity. Zvolená mobilní platforma a sama aplikace bude detailněji rozebrána a formou diagramů bude popsána její funkčnost. Následují úryvky zdrojových kódů aplikace a uživatelská příručka včetně obrázkového doprovodu.

Výsledek práce bude shrnut v závěru. Zde budou zároveň navrženy možné způsoby vylepšení aplikace.

1 Rešerše mobilních platforem a dostupných aplikací

V této části bude provedena rešerše již dostupných možných způsobů řešení přenosu hlasu mezi dvěma zařízeními, s funkcemi využitelnými spolujezdcem a řidičem motocyklu. Tato problematika se v praxi řeší hlavně rádiovými vysílačkami, které mají spoustu výhod. Jsou vytvářeny přímo pro účel přenosu hlas a často bývají zabudovány přímo do přileb. Jejich nevýhodou, kterou odstraňuje řešení prostřednictvím mobilní platformy, je cena. V kapitolách níže jsou proto popsány mobilní platformy a aplikace, které se na konkrétních platformách dají k přenosu hlasu přes Bluetooth použít. Tyto aplikace jsou následně porovnány s interkomy, které jsou k této činnosti určené.

1.1 iOS

iOS je jeden z nejrozšířenějších mobilních operačních systémů. Je vytvořený firmou Apple Inc. původně pro mobilní telefony iPhone. Nyní se však začal rozšiřovat i do ostatních zařízení této firmy, jako jsou iPod, iPad nebo Apple Tv. Pojmenování iOS se používá až od čtvrté verze tohoto systému, dříve se nazýval iPhone OS. V současné době je aktuální verze systému iOS 6.

iOS je odlehčená verze operačního systému Mac OS X, který se používá v počítačích společnosti Apple. Jedná se tedy o operační systém UNIXového typu, který však neobsahuje veškerou jeho funkcionalitu. Oproti OS X však přidává podporu dotykového ovládání.

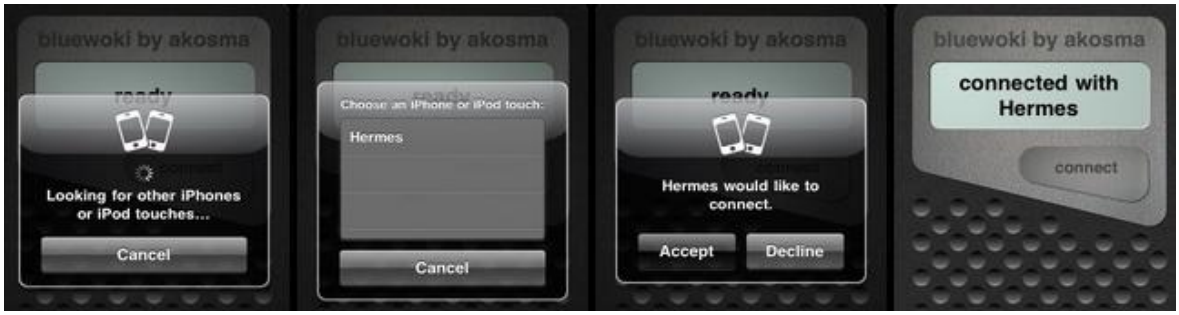
iOS je druhým nejpoužívanějším operačním systém, přičemž ho v současné době Android poráží zhruba v poměru 1:5. Nevýhodou iOS je oproti Androidu možné použít pouze v zařízeních od firmy Apple, které jsou často mnohokrát dražší než zařízení s operačním systémem Android, čímže se stává pro většinu lidí dostupnější.

Popularita tohoto systému spočívá především v zařízeních, která jsou na této platformě postavena. Velkou předností je existence mnoha aplikací, které jsou dostupné na App Store[1].

1.1.1 Bluewoki

Jednou z aplikací, která je schopna na operačním systému iOS přenést zvuk prostřednictvím technologie bluetooth, je aplikace bluewoki. Tato aplikace je podporována od verze iOS 3, a kromě bluetooth je od verze 2.0 schopná přenést hlas mezi dvěma telefony iPhone nebo iPady i prostřednictvím technologie Wifi přes lokální síť.

[1] App Store je obchod s aplikacemi pro zařízení s iOS provozován firmou Apple



Obrázek 1 - iOS: Bluewoki

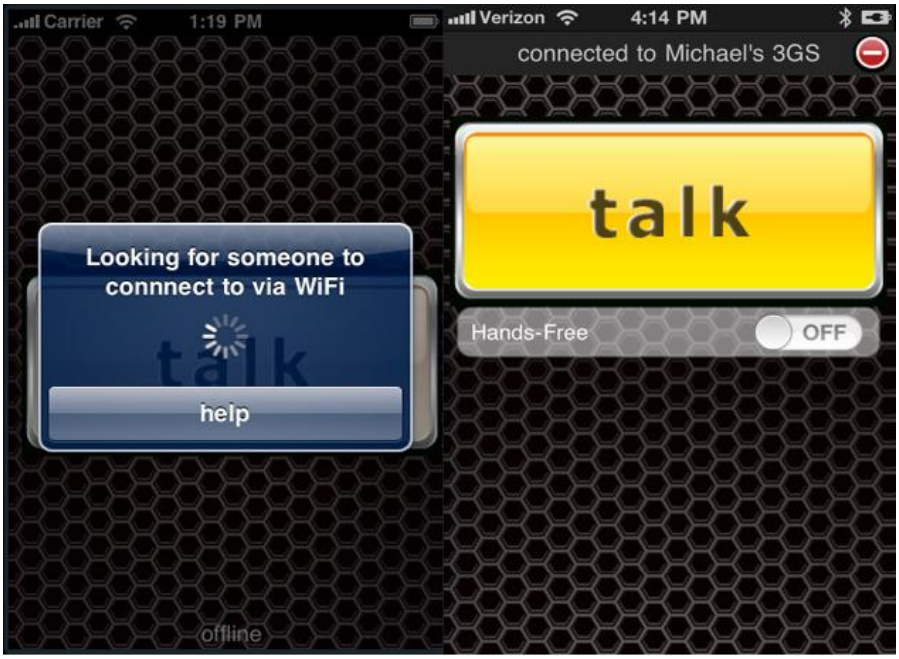
Na obrázku výše jsou 4 obrazovky aplikace. První vyhledává dostupná zařízení. Na druhé obrazovce je seznam vyhledaných zařízení. Po vybrání a kliknutí na název zařízení je na připojovaném zařízení nutné připojení povolit. Následně jsou obě zařízení spojeny. Tato aplikace je schopná peer-to-peer spojit dvě zařízení ať už prostřednictvím Bluetooth, či přes wifi v případě, že jsou dva telefony připojeny ve stejné wifi síti.

Příjemnou výhodou je dostupnost zdrojového kódu aplikace z https://github.com/akosmasoftware/bluewoki a možnost stažení zdarma přímo z AppStore. Oproti některým konkurenčním aplikacím dokáže od verze 2.0 přenést hlas i prostřednictvím wifi.

Nevýhodou však je schopnost současného propojení pouze dvou zařízení.

1.1.2 Walkie Talkie - WIFI & Bluetooth

Walkie Talkie - WIFI & Bluetooth je další aplikací pro zařízení s iOS, která je schopná přenést hlas prostřednictvím Bluetooth a podobně jako aplikace Bluewoki i přes wifi. Oproti Bluewoki se ale uživatel o nic nestará a po zapnutí aplikace na obou zařízeních se sama propojí. K tomu musí být zařízení opět ve stejné síti. Aplikace je dostupná pro Iphone, iPody touch a iPady. Zmíněná zařízení musí disponovat operačním systémem iOS alespoň ve verzi 4.0.



Obrázek 2 - iOS: Walkie Talkie – Wifi & Bluetooth

Na předchozím obrázku je ukázka aplikace ve chvíli, kdy se vyhledávají zařízení v lokální síti a následně, kdy je k zařízení připojeno jiné zařízení z této sítě.

Výhodou je, že je zdarma ke stažení z AppStore, a umožňuje přenos mezi telefony i prostřednictvím technologie wifi.

Nevýhodou pak, že dokáže prostřednictvím Bluetooth technologie spojit v jednom okamžiku pouze dvě zařízení.

1.2 Windows Phone

Windows Phone je nejmladším z popisovaných operačních systémů, nástupcem mobilního operačního systému Windows Mobile od firmy Microsoft, se kterým však není zpětně kompatibilní. Windows Phone byl vydán 21. 10. 2010 a v současné době jsou jako Windows Phone označovány verze Windows Phone 7 a Windows Phone 8. Nabízí nové uživatelské rozhraní Modem UI (Metro) stejně jako stolní verze Windows 8, kde jsou ikony na ploše nahrazeny takzvanými huby uspořádanými do dlaždic.

Verze Windows Phone 7 byla vydána v polovině roku 2010 a stále běžela, stejně jako Windows Mobile, na jádře Windows CE. Následující verze Windows 8 (vydáno 29. 10. 2012) už však obsahuje nové jádro systému přizpůsobené z řady Windows NT a nahrazuje tak zcela původní systém architektury Windows CE. Tím se ale přerušuje zpětná kompatibilita aplikací. Aplikace z Windows Phone 8 jsou však kompatibilní s Windows 8.

Aplikace pro Windows Phone jsou centralizovány na Windows Phone Store[2] (dříve Windows Phone Market).

Překvapivě tento mladý operační systém, vázající se na tradici slavného Windows Mobile, nemá zatím dostatečnou základnu aplikací a pravděpodobně není momentálně dostupná žádná aplikace, která by umožňovala přenos hlasu přes Bluetooth. Některé zdroje uvádějí, že na Windows Phone existuje stejně jako na Android aplikace Blue Talkie, to se však nepodařilo na oficiálním zdroji potvrdit. Jediné aplikace, umožňující podobný přenos hlasu fungují pouze prostřednictvím 3G, Wifi nebo jiného zdroje internetu.

1.3 Symbian OS

Symbian je svobodný operační systém, navržený pro mobilní zařízení. Jeho předchůdcem byl operační systém EPOC, který byl vytvořen pro kapesní počítače firmy Psion. V roce 1998 se ke zmíněné firmě Psion připojily další firmy mobilních technologií, jmenovitě Nokia, Ericsson a Motorola. Od tohoto roku se operační systém dál vyvíjel pod názvem Symbian. Dnes se Symbian používá převážně v mobilních telefonech značky Nokia, která také vlastní přes 99,9% všech akcií tohoto operačního systému. Symbian nestačil v poslední době držet krok s konkurenčními operačními systémy, a tak 11. února 2011 Nokia oznámila ústup od dalšího vývoje a přechod k platformě Windows Phone 7.

1.3.1 Bluetooth Walkie-Talkie

Jednou z aplikací, která dokáže pod tímto operačním systémem přenést hlas mezi dvěma a více telefony prostřednictvím technologie Bluetooth je Bluetooth Walkie-Talkie.

Uživatel se nejprve musí ujistit, že je na obrazovce zobrazeno „Active“. Je-li služba aktivní, může zobrazit seznam dostupných zařízení přes“Options“-„Refresh“. Chce-li uživatel odeslat hlasovou zprávu, stačí stisknout a držet zelené tlačítko.

[2]Windows Phone Store jecentrální databáze Aplikací, muziky a podobně od Microsoftu pro uživatele Windows Phone.

Obrázek 3 - Symbian: Bluetooth Walkie-Talkie

Výhodou je, že dokáže prostřednictvím Bluetooth spojí více než dvě zařízení.

Nevýhodou této aplikace je schopnost pouze half-duplex spojení, tzn., že zařízení mohou v jednom okamžiku pouze buď přijímat, nebo jen vysílat.

1.3.2 Walkietooth

Walkietooth je další aplikací umožňující přenos hlasu mezi dvěma chytrými telefony s operačním systémem Symbian S60.

Obrázek 4 - Symbian: Walkietooth

Aplikace má jednoduché ovládání. Spojí dvě zařízení v závislosti na jejich roli, kterou si vyberou na úvodní obrazovce, jak je vidět na obrázku 4.

1.4 Android

Android je open source platforma vyvíjená konsorciem OHA[3], zabývající se rozvojem mobilních technologií, které budou mít výrazně nižší náklady na vývoj a distribuci a zároveň uživatelům přinese uživatelsky přívětivé prostředí.

V říjnu roku 2003 byla založena společnost Android Inc, která byla jako nepříliš známá odkoupena v roce 2005 společností Google Inc. Tým Googlu vyvinul platformu založenou na Linuxovém jádře. V září 2007 získal několik patentů v oblasti mobilních technologií. 5. Listopadu 2007 bylo vytvořeno uskupení OHA, jehož cílem bylo vyvinutí otevřeného standardu pro mobilní zařízení. Tentýž den OHA ohlásila svůj první produkt, kterým byl Android, otevřená mobilní platforma postavená na jádře Linux verze 2.6.

V říjnu roku 2008 byl ve Spojených státech amerických uveden první komerční telefon vyrobený firmou HTC, který běžel s operačním systémem Android.

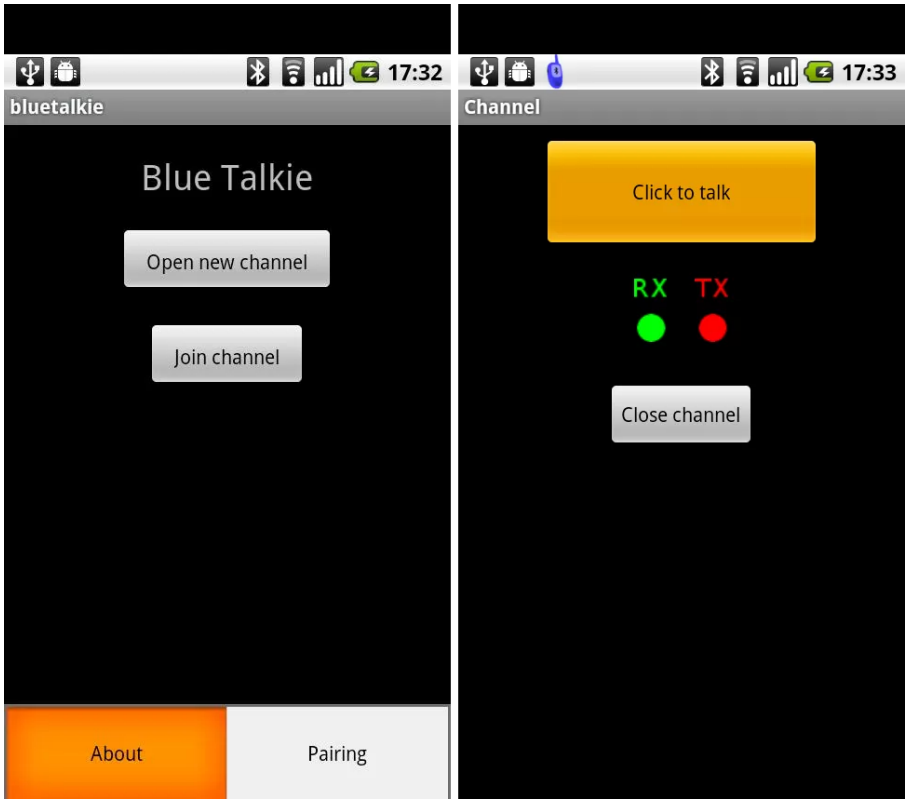
Android byl původně navržený pro mobilní zařízení, dnes již se ale vyskytuje například i v noteboocích. V současné době je operační systém Android nejoblíbenějším mobilním OS. Jeho velká rozšířenost spočívá především v nízkých pořizovacích cenách přístrojů s tímto operačním systémem. Tyto přístroje nejsou na rozdíl od jeho největšího konkurenta iOS instalována pouze do zařízení jedné jediné značky, a proto se dá vybírat z velkého množství modelů. Další výhodou je obrovská databáze placených i bezplatných aplikací centralizovaných na Google Play[4] a rychle se rozrůstající komunita kolem tohoto systému.[1]

1.4.1 Blue Talkie

Blue Talkie je aplikace řešící komunikaci mezi dvěma a více telefony pro operační systém Android. Umožňuje full-duplexově spojit libovolný počet zařízení a spojení probíhá v režimu Client-Server.

V úvodní obrazovce aplikace si uživatel zvolí, jestli chce vytvořit kanál (Server), nebo se připojit k vytvořenému kanálu (Client). Tato obrazovka je vidět vlevo na obrázku č. 5. Na stejném obrázku vpravo je obrazovka ve chvíli komunikace.

[3] Open Handset Aliance
[4] Google Play je centralizované úložiště aplikací pro operační systém Android



Obrázek 5 - Android: Blue Talkie

Aplikace je volně stažitelná z Google Play. Další výhodou je, že dokáže přes Bluetooth spojit v jednom okamžiku full duplexně více zařízení.

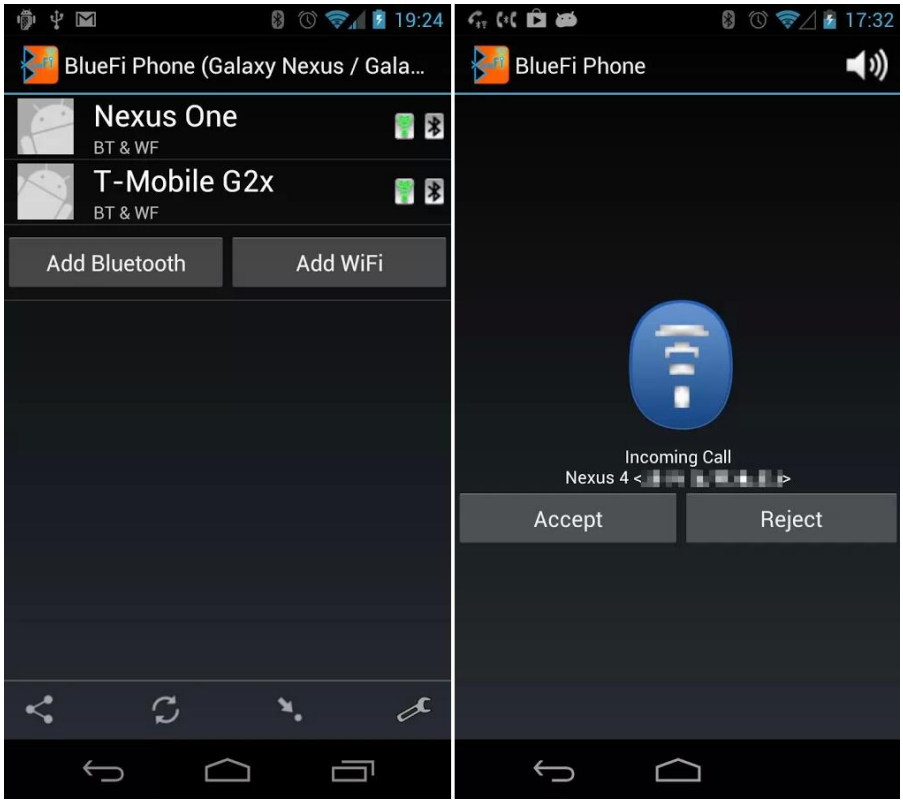
Nevýhodou je absence jakékoliv možnosti ovlivnění připojených zařízení a citlivosti snímání zvuku.

1.4.2 BlueFi Phone

BlueFi Phone umožňuje full duplexní spojení více zařízení mezi sebou v jednom okamžiku prostřednictvím Bluetooth, Wifi (připojení ke stejnému hot spot), nebo prostřednictvím obou technologií zároveň. Aplikace uživateli poskytuje seznam připojených kontaktů, které může podle specifik sdružovat.

Aby byla aplikace schopna komunikovat přes wifi, musí příslušný směrovač podporovat multicast paketů.

Vzhled aplikace v době připojení několika zařízení a v době přijímání hovoru je na následujícím obrázku č. 6.



Obrázek 6 - Android: BlueFi Phone

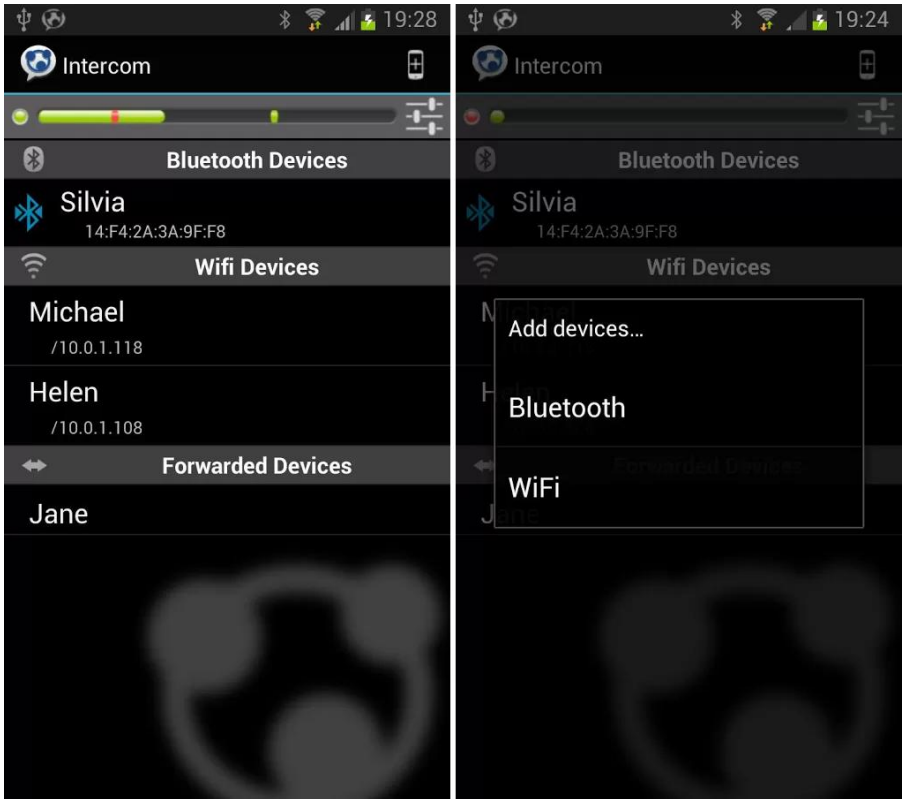
Výhodou aplikace je schopnost spojení i přes wifi. V případě tohoto spojení umožňuje ping na příslušný kontakt. Aplikace je volně stažitelná z Google Play.

Nevýhodou je absence možnosti řešení citlivosti mikrofonu a řešení automatické detekce hlasu.

1.4.3 Android Intercom

Umožňuje full duplexní spojení více telefonů prostřednictvím Bluetooth nebo wifi technologie. V případě užití Bluetooth aplikace funguje spolehlivěji, dokáže ale spojit pouze dvě zařízení. Využije-li uživatel wifi, může zařízení připojit více, a poskytuje tak možnost skupinové komunikace.

Vzhled aplikace je zachycen na obrázku č. 7. Na levé straně je obrazovka se třemi připojenými zařízeními. Nad nimi je graficky zobrazena hladina zachytávaného zvuku. Na pravé obrazovce si uživatel vybírá, jestli další zařízení připojí přes wifi či Bluetooth.



Obrázek 7 - Android: Android Intercom

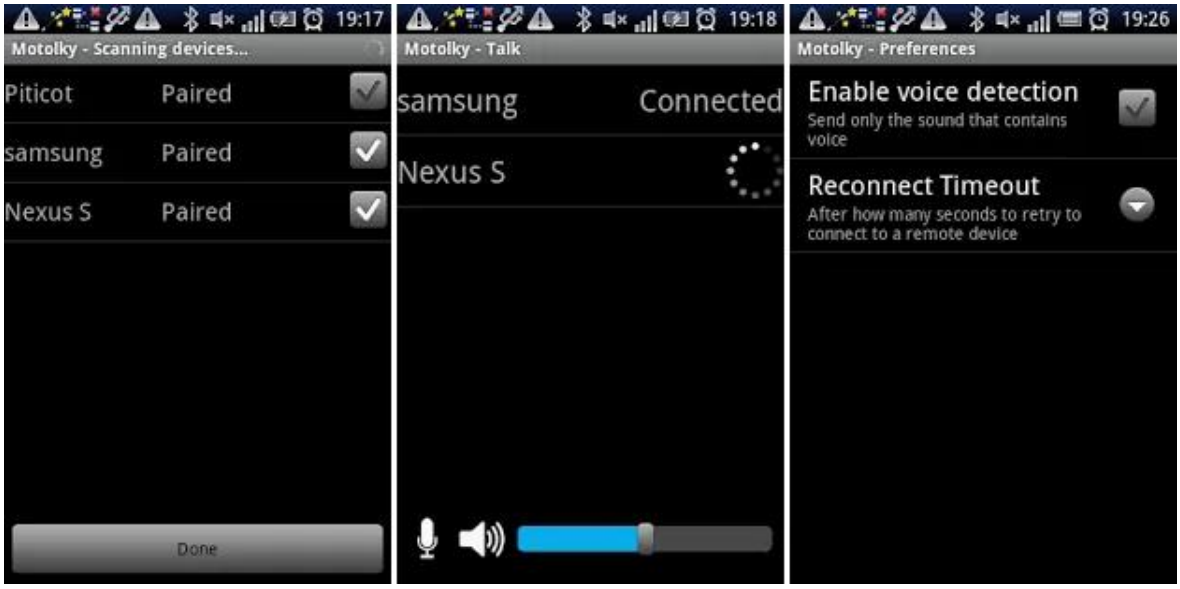
Umožňuje i spojení přes wifi. Další výhodou je schopnost automatického opětovného připojení v případě, že bylo zařízení z důvodu například příliš velké vzdálenosti odpojeno, a schopnost automaticky detekovat řeč. Aplikace je volně stažitelná z Google Play.

Nevýhodou je občasná nestabilita aplikace na starších zařízeních a schopnost přes Bluetooth spojit pouze dvě zařízení v jednom okamžiku.

1.4.4 Motolky

Aplikace Motolky je další z řad aplikací pro Android řešící komunikaci přes Bluetooth. Svými funkcemi, a zřejmě i účelem, je aplikace dost podobná předchozí aplikaci Android Intercom. Na rozdíl od ní ale umožňuje full-duplexně spojit několik zařízení v jednom okamžiku prostřednictvím technologie Bluetooth, a naopak postrádá možnost spojení přes wifi.

Vzhled aplikace je na obrázku č. 8. První obrazovka ukazuje stav, kdy si uživatel vybírá mezi spárovanými zařízeními ta, ke kterým se chce připojit. Druhá obrazovka zobrazuje aplikaci ve chvíli, kdy je s jedním zařízením spojená a ke druhému se připojuje. Na třetí obrazovce je nastavení, kde může uživatel ovlivnit detekci hlasu nebo interval, ve kterém se odpojené zařízení pokouší znovu připojit.



Obrázek 8 - Android: Motolky

Aplikace se dokáže opětovně připojit k jinému zařízení v případě výpadku spojení. Další výhodou je možnost automatické detekce hlasu. Aplikace je volně stažitelná z Google Play.

Nevýhodou je občasná nestabilita při testování převážně na starších zařízeních.

1.5 Intercom

V předcházejících bodech byly popsány aplikace, které se sice dají použít jako vysílačky, ale stále jde jen o náhražky profesionálních zařízení, které jsou za účelem plnění své funkčnosti vyráběny. Tyto vysílačky jsou schopny komunikovat přes Bluetooth, v případě potřeby dosažení větší vzdálenosti však lze využít vysílaček umožňujících přenos zvuku rádiovými vlnami. Některé vysílačky jsou schopny komunikovat přes rádiové vlny i přes Bluetooth. Díky rozhraní Bluetooth jsou schopny se spárovat prakticky s čímkoliv, co má rovněž Bluetooth rozhraní.

Výhodou těchto vysílaček je, že mohou disponovat dalšími vlastnostmi, kterých u mobilní aplikace nedosáhneme. Patří mezi ně například odolnost proti vodě, kompaktnější rozměry, pohodlnější manipulace, větší výdrž apod. Často disponují možností regulace hlasitosti podle druhu jejího využívání, takže když uživatel například poslouchá hudbu a začne přitom mluvit, hudba se automaticky ztiší. Obvykle disponují efektivním hlukovým filtrem, takže jsou i při vyšší hlučnosti schopny dosáhnout uspokojivé srozumitelnosti přeneseného hlasu.

Nespornou nevýhodou je ale jejich cena. Počet chytrých telefonů v dnešní době prudce roste, a tak vysílačka na krátkou vzdálenost jako aplikace do mobilního telefonu je pro běžného uživatele mnohem dostupnější.

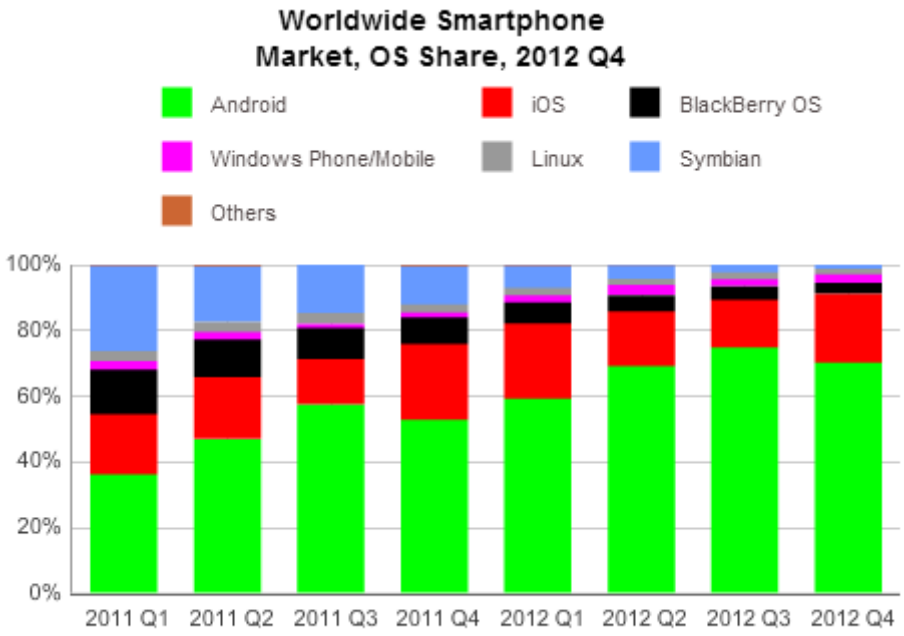
Obrázek 9 - Bluetooth vysílačka

Na obrázku 9 je ukázka vysílačky Albrecht A601 pro integrální přilbu.

1.6 Srovnání

V předchozích odstavcích byly shrnuty vlastnosti některých operačních systémů pro mobilní zařízení. Nebyly popsány systémy jako Bada, Windows Mobile a další, jelikož pro ně pravděpodobně není dostupná aplikace splňující požadavky této práce. Pokud si v dnešní době chce člověk pořídit mobilní telefon s operačním systémem, nemá při výběru lehký úkol, jelikož množství mobilních platforem čím dál více narůstá. Situace je o to složitější, že zatímco dříve se od sebe mobilní platformy dalece odlišovaly, dnes prakticky každá splní vše, co se dá od mobilního operačního systému očekávat.

Trh s mobilními zařízeními dnes přesvědčivě ovládají nováčci v oblasti mobilních operačních systémů iOS a Android. Zvláště v posledních letech ale začíná druhý jmenovaný ostatním systémům značně odbíhat, a to především díky většímu výběru zařízení a jejich ceně. Na následujícím obrázku č. 10 je graficky nastíněn vývoj podílu trhu konkrétních mobilních operačních systémů v roce 2011 a 2012 zveřejněné analytickou společností IDC.



Obrázek 10 - Podíl mobilních platforem

Počet prodaných zařízení a podíl operačních systémů na trhu mobilních zařízení je uveden v následující tabulce.

Tabulka 1 - Podíl mobilních platforem

Operační systém	2012 [mil. ks]	2012 [%]	2011 [mil. ks]	2011 [%]	Rozdíl [%]
Android	497,1	68.8%	243,5	49.2%	104.1%
iOS	135,9	18.8%	93,1	18.8%	46.0%
BlackBerry	32,5	4.5%	51,1	10.3%	-36.4%
Symbian	23,9	3.3%	81,5	16.5%	-70.7%
Windows Phone a Windows Mobile	17,9	2.5%	9	1.8%	98.9%
Ostatní	15,1	2.1%	16,3	3.3%	-7.4%
Celkem	**722,4**	**100.0%**	**494,5**	**100.0%**	**46.1%**

Z obrázku č. 10 a tabulky č. 2 je zřetelně vidět pokles starších platforem s tradicí a viditelný nárůst nováčka - operačního systému Android.

Srovnat mezi sebou dnešní mobilní operační systémy je složité a ještě složitější je srovnat jejich aplikace. V následující tabulce je seznam výše popsaných řešení a základních vlastností, kterými mohou vysílačky disponovat.

Tabulka 2 - Přehled vlastností řešení

Vlastnost / Řešení	W	N	F	R	U	I	O	Z	C	D
Blue Talkie	a	n	n	n	n	a	n	n	a	a
Blue-Fi Phone	a	a	n	n	a	n	n	n	a	a
Android Intercom	a	wa	a	a	a	n	n	n	a	a
Motolky	n	a	a	a	a	a	a	n	a	a
Bluewoki	a	-	n	n	-	n	n	n	a	a
Walkie Talkie – Wifi & Bluetooth	a	wa	n	n	n	a	n	n	a	a
Bluetooth Walkie-Talkie	n	-	n	n	-	-	-	n	a	n
Walkietooth	n	n	n	n	-	-	-	-	a	a
Intercom	a	a	a	a	-	-	-	-	n	a
Vlastní aplikace	n	a	a	a	a	a	a	a	a	a

Vysvětlivky ke zkratkám v tabulce:

- a – řešení disponuje touto vlastností
- n – řešení nedisponuje touto vlastností
- p – za poplatek disponuje touto vlastností
- wa – pouze v režimu wifi
- W – možnost spojení prostřednictvím wifi
- N – spojení mezi více než dvěma zařízeními
- F – nějaký způsob filtrování hluku (filtr, citlivost mikrofonu apod.)
- R – reconnect – možnost se při výpadku spojení automaticky znovu připojit
- U – ovlivnění připojených uživatelů
- I – ovlivnění vstup (mikrofon)
- O – ovlivnění výstupu (reproduktor)
- Z – záznam komunikace
- C – dostupnost řešení zdarma
- D – full-duplex

Z tabulky je přehledně vidět, jakými vlastnostmi disponují konkrétní řešení. V případě, že kolonka obsahuje znak "-", znamená to, že k danému řešení nebyla z nějakého důvodu jeho funkčnost potvrzena. V posledním řádku je popsána aplikace, která je vyvíjena v rámci této práce. Vzhledem k tomu, že je vyvíjena pro konkrétní účel, ke kterému jsou postaveny i parametry tabulky, očekává se, že v porovnání s ostatními dopadne nejlépe. Popis

jednotlivých vlastností není konečný a funkce aplikace budou rozšířeny o další funkce užitečné v této problematice.

2 Analytická část

2.1 Úvod do analytické části

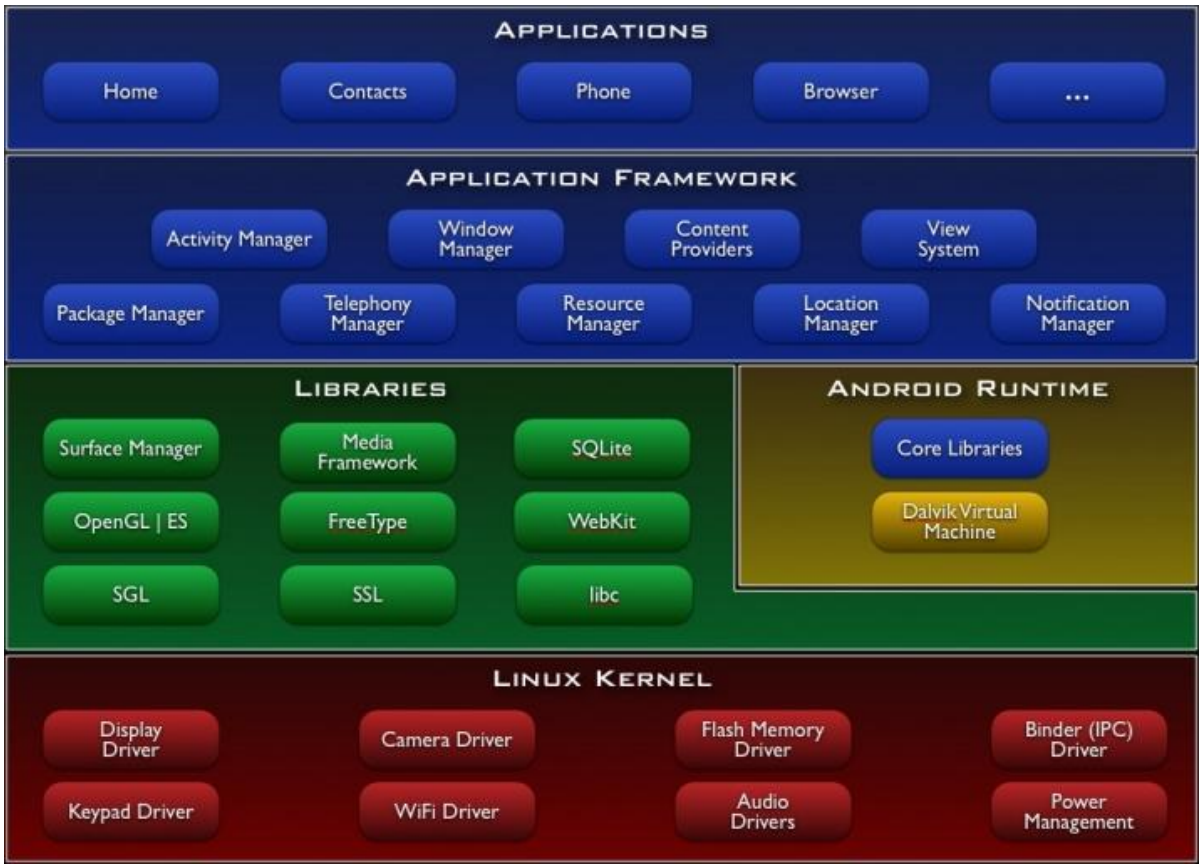
Úvodem je dobré říci, pro koho je aplikace vlastně určena. Tato aplikace je vyvíjena zejména pro příznivce motorismu. Aplikace zvládne přenos hlasu minimálně mezi řidičem a spolujezdcem motocyklu. Dále by aplikace měla být schopná řešit možnost odfiltrování okolního šumu a hluku, jakým je například zvuk motocyklu. Aplikace bude zaznamenávat informace o odjeté trati, mezi které patří hlavně záznam trasy a rychlost v jednotlivých místech, a bude poskytovat informaci o tom, v jakých místech se zaznamenávala komunikace, kterou je možné zpětně přehrát.

2.2 Volba platformy

Z porovnání jednotlivých mobilních platforem v úvodní části je patrné, že operační systém Android má v současnosti obrovský potenciál a aplikace pro něj tak mohou rychle najít uplatnění. Navíc se programování pro Android na první pohled jeví jako snadné. Z těchto důvodů je vyvíjená aplikace navržena právě pro něj.

2.2.1 Android architektura

Architektura operačního systému Android je rozdělena do pěti vrstev a každá vrstva plní svůj vlastní účel. Rozdělení těchto vrstev je viditelné na následujícím obrázku 11.



Obrázek 11 - Vrstvy OS Android (převzato z [2])

Popis jednotlivých vrstev:

Linux Kernel, jádro operačního systému, je na obrázku ve spodní části zobrazen červenou barvou. Jádro operačního systému Android je nejnižší vrstvou, představující abstrakci mezi hardwarem a softwarem ve vyšších vrstvách. Je postaveno na Linuxu verze 2.6, kde Google provedl jen pár úprav, takže s hardwarem a zbytkem komunikuje Linux i s jeho správou pamětí, procesů a sítě, ovladači apod. Linuxové jádro je dobré i vzhledem k jeho snadnému sestavení na různých zařízeních, čímž je zaručena přenositelnost.

Libraries, česky knihovny. Tyto knihovny jsou psány v jazyce C a C++, a jsou využívány různými komponentami systému. Vývojáři jsou tyto knihovny zpřístupněny prostřednictvím vrstvy Android Application Framework.

Android Runtime obsahuje Virtuální Stroj Dalvik (DVM) a Java knihovny. Dalvik je vyvinut speciálně pro Android týmem Googlu. DVM vznikl především ze dvou důvodů. Prvním bylo, že zatímco Java a její knihovny jsou volně šiřitelné, Java Virtuální Stroj (JVM) není. Druhým důvodem byla optimalizace virtuálního stroje pro mobilní zařízení, které kladou větší důraz především na úsporu energie a mají podstatně nižší výkon. Java knihovny jsou svým obsahem podobné platformě Java Standard Edition (JSE). Rozdíl je především v nahrazení knihoven uživatelského rozhraní (AWT[5], Swing[6]) knihovnami uživatelského rozhraní pro Android. Přidána byla knihovna Apache pro práci se sítí.

Application Framework je nejdůležitější vrstva pro vývojáře. Poskytuje přístup ke službám, které mohou být použity přímo v aplikaci. Tyto služby mohou například zpřístupňovat data jiným aplikacím, umožňují běh aplikace na pozadí apod. Mezi tyto služby patří především následující:

- View - slouží k vytvoření uživatelského rozhraní. Jedná se o prvky jako je tlačítko, textové pole atd.
- Content Provider - zprostředkovává přístup k obsahu jiných aplikací (například kontakty).
- Notification Manager - poskytuje aplikacím přístup ke stavovému řádku.
- Activity Manager - řídí životní cyklus aplikací a poskytuje orientaci v zásobníku s aplikacemi.

Applications je nejvyšší vrstva Android architektury. Jde o předinstalované i dodatečně instalované aplikace a je využívána běžnými uživateli.[1][2]

2.2.2 Vývoj aplikace

Aplikace pro Android jsou psány převážně v jazyce Java. Tento zdrojový kód aplikace je zkompilován do Java byte kódu, který je následně překompilován v Dalvik kompilátoru. Výsledný Dalvik kód je spouštěný na DVM. Každá aplikace běží ve svém vlastním procesu s vlastní instancí DVM.

[5] Abstract Window Toolkit je část Java Core Api, umožňující tvorbu grafického uživatelského rozhraní.
[6] Swing je knihovna uživatelských prvků psaná v Javě pro ovládání počítače pomocí grafického rozhraní.

Oficiálně podporované vývojové prostředí pro Android aplikace je Eclipse, ale samozřejmě lze aplikace vyvíjet i v ostatních vývojových prostředích, nebo ji lze psát jednoduše v textovém editoru a kompilovat v příkazové řádce, což ale v dnešní době nepatří mezi nejpohodlnější způsoby vývoje. Do eclipse, či jiných IDE, lze doinstalovat plugin Android Development Tools (ADT), který výrazně ulehčuje vývoj Android aplikace[7].

Aplikace se dají vyvíjet buď na fyzicky připojeném zařízení, nebo na emulátoru, který je součástí Android SDK. V obou případech se aplikace chovají většinou stejně. Existují však situace, jako je přijetí hovoru, funkce Bluetooth apod., které se simulovat nedají, a je nutné provádět vývoj na připojeném fyzickém zařízení.

Nástroje potřebné pro vývoj Android aplikace jsou obsaženy v Android Software Development Kit (ADK). Ten je dostupný pro všechny hlavní platformy operačních systémů. ADK je rozdělen na následující části:

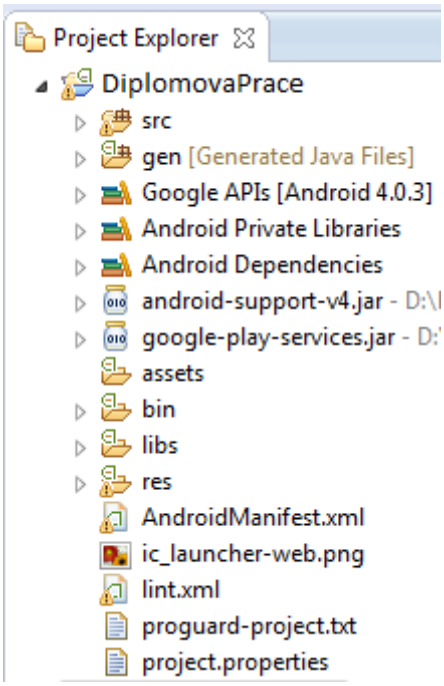
- Základní konfigurace - obsahuje například nástroje pro debugging a testování aplikace, správu virtuálních zařízení Android Virtual Device (AVD), Android emulátor nebo Android Debug Bridge, který umožňuje nahrávání souborů do zařízení. Dále například Android SDK platforms, kde si uživatel musí nainstalovat minimálně jednu platformu, pro kterou chce vyvíjet. Každá platforma je rozdělena na několik částí, jako třeba vzorové příklady pro konkrétní platformu, knihovny pro platformu a podobně.
- Doporučená konfigurace - obsahuje například USB ovladače. Ty jsou potřebné pouze na platformu Windows a umožňují ladění aplikace přímo na fyzicky připojeném zařízení.
- Plná konfigurace - obsahuje například knihovny pro běh Google Maps.

Android projekt se skládá ze spousty rozdílných souborů. Ve chvíli, kdy se buildne, zabalí se veškerý přiložený kód spolu s reource soubory (obrázky, styly, layouty, …) a deskriptorem aplikace do podepsaného zip souboru s příponou ".apk". Tento soubor je připraven pro spuštění buď v emulátoru, nebo na fyzicky připojeném zařízení.[1]

2.2.3 Struktura projektu

Android projekt je složen z několika důležitých částí. Struktura takového projektu je zobrazena na následujícím obrázku č. 12 a popsána níže.

[7]Eclipse IDE už s vestavěným ADT pluginem lze stáhnout ze stránky http://developer.android.com/sdk/index.html

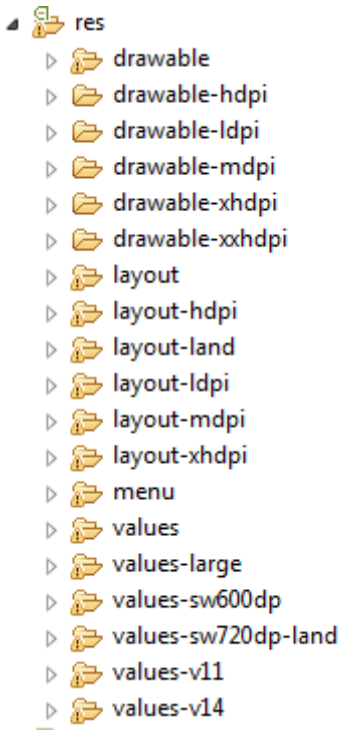


Obrázek 12 - Adresářová struktura projektu

Na obrázku č. 12 je vidět adresářová struktura projektu DiplomovaPrace. Projekt je rozdělen do několika adresářů, každý adresář má svůj vlastní druh obsahu.

- src – obsahuje zdrojové kódy psané v Javě strukturované do balíčků. Kořenový balíček je deklarován v souboru AndroidManifest.xml a je použit jako unikátní identifikátor aplikace.
- gen – adresář, ve kterém se automaticky generuje třída R.java. Tato umožňuje odkazovat se ze souborů v adresáři src na prostředky v adresáři res. Třída by se neměla ručně měnit.
- přilinkované soubory ".jar".
- assets - slouží k umístění různých datových souborů, které bude aplikace využívat.
- bin - obsahuje soubory vzniklé kompilací projektu.
- lib - obsahuje knihovny.
- res - obsahuje prostředky aplikace. Pro všechny soubory se automaticky vytváří index v podobě statického klíče ve třídě R.java.
- AndroidManifest.xml – popisuje aplikaci a její komponenty.
- project.properties - slouží pro úpravu properties používaných programem ant při sestavování aplikace.
- proguard-project.txt – soubor, kterým lze definovat jakým způsobem proběhne optimalizace výsledného kódu.

Adresář res (resources) obsahuje další podsložky, které vývojář může většinou přidávat a odebírat jak potřebuje. Příklad struktury adresáře je zobrazen na následujícím obrázku č. 13.



Obrázek 13 - Struktura adresáře res

Adresář res obsahuje prostředky aplikace, jako například obrázky, xml soubory a další. Nejdůležitější jsou tři základní druhy prostředků (složky):

- drawable – obsahuje převážně obrázky aplikace.
- layout – obsahuje xml soubory, popisující vzhled jednotlivých komponent aplikace.
- values – obsahuje xml soubory, jako je soubor lokalizačních řetězců, nadefinované barvy, styly apod.

Ze struktury je zřejmé, že některé složky mají své alternativy lišící se pouze koncovkou. Koncovka určuje, jaká složka se použije pro konkrétní zařízení v závislosti na rozlišení jeho displeje a podobně. To znamená, že například při překlopení displeje se spustí nová aktivita se sadou resource souborů příslušících překlopenému displeji. Nepřiřadí-li se vhodná koncovka, použije se složka bez koncovky. Koncovky složek mohou být například následující:

- ldpi – nízké rozlišení displeje (~120dpi).
- mdpi – střední rozlišení displeje (~160dpi).
- hdpi – vysoké rozlišení displeje (~240dpi).
- xhdpi – extra vysoké rozlišení displeje (~320dpi).
- land, port – určuje orientaci displeje.
- small, normal, large – velikost displeje.
- cs,en, …- jazyk telefonu.

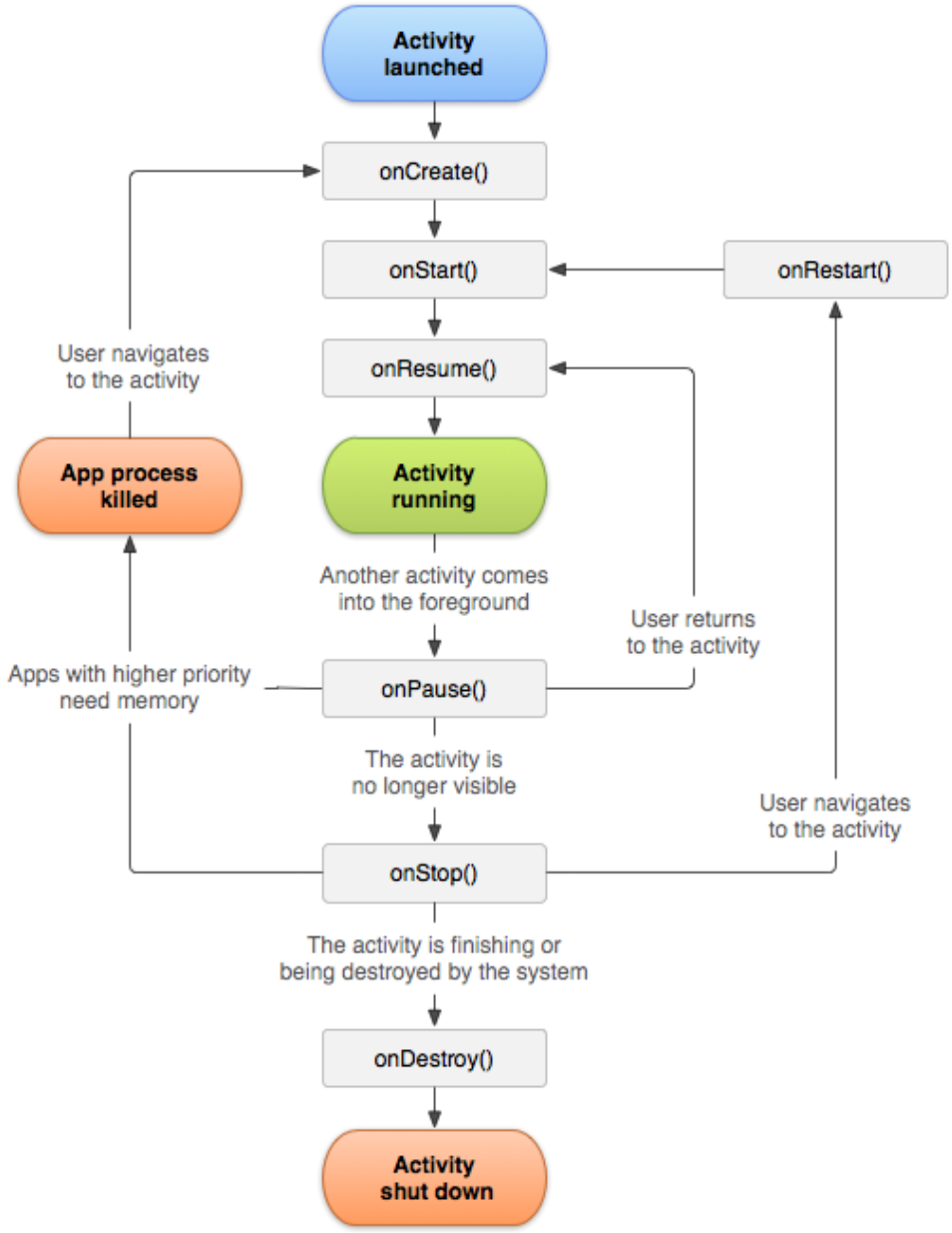
2.2.4 Základní prvky aplikace

Na rozdíl od programování většiny jiných aplikací se zde programátor nesetká se statickou metodou main, implementovanou v rámci hlavní třídy, která ovlivňuje běh aplikace. Místo toho se Android stará o vytváření, ničení instancí aktivit apod. sám a v průběhu života jim pomocí životního cyklu určuje, co mají dělat. Architektura android aplikací je stavěna tak, že nepoužívané části aplikace jsou odklízeny a uvolňují tak paměť. K tomu, aby probíhalo odklízení korektně, musí programátor patřičně reagovat na změny v životním cyklu aplikace a v pravý okamžik například uložit rozepsaný formulář apod.

Android má několik stavebních prvků, bez kterých by se vytvořená aplikace neobešla. Mezi základní patří především Activity (aktivita), Service (služba), Content provider a Broadcast receiver. Všechny tyto komponenty musejí být deklarovány v souboru Android Manifest.xml, který je v každé aplikaci uložen v kořenovém adresáři a poskytuje informace o aplikaci. Podrobněji jsou tyto komponenty popsány níže.

Activity představuje jednu obrazovku aplikace, která uživateli poskytuje informace. Při vytváření Activity se vytváří nový proces a alokuje se paměť pro objekty uživatelského rozhraní. Za správu životního cyklu Activity zodpovídá Activity Manager. Ten pracuje se zásobníkem s uloženými informacemi o spuštěných aktivitách. Na vrcholu tohoto zásobníku je poslední spuštěná aktivita.

Aktivita má životní cyklus, který je znázorněný na následujícím obrázku, a pro vytvoření dobré aplikace je nezbytné jeho pochopení.



Obrázek 14 - Životní cyklus Activity (převzato z [3])

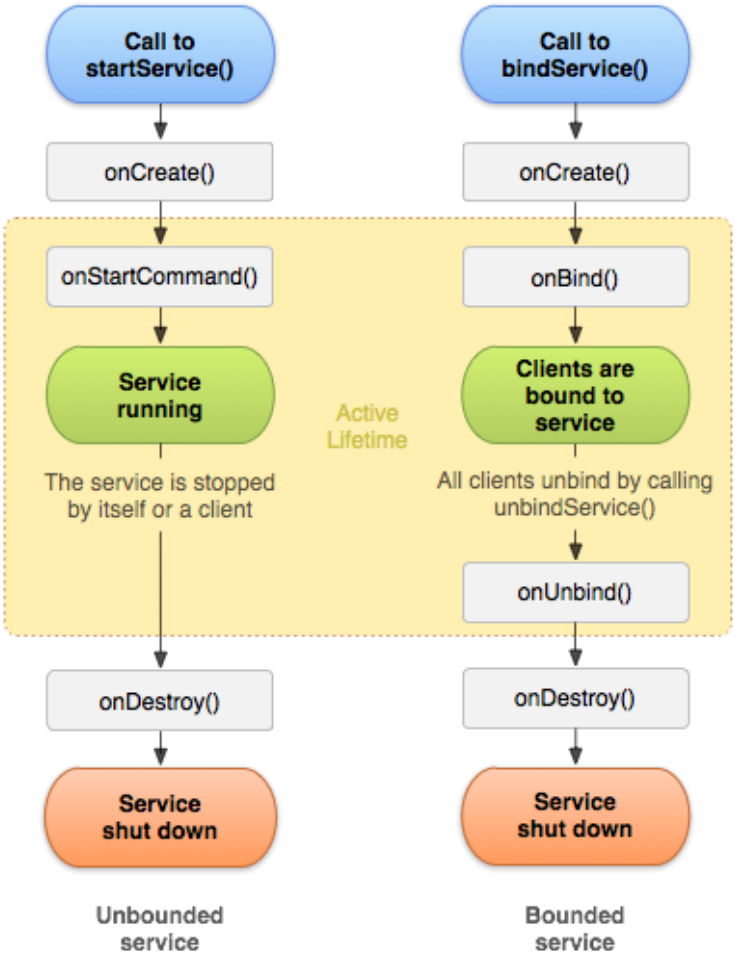
Z obrázku je patrné, že se aktivita může nacházet v následujících stavech:

- Launched – započetí (inicializace) aktivity.
- Running – v tomto stavu může být v jednom okamžiku právě jedna aktivita. Je to aktivita, která může mít interakci s uživatelem. Všechny aktivity, s nimiž uživatel pracuje, jsou v tomto stavu.
- Killed – do tohoto stavu se aktivita dostane, pokud je aktivita viditelná, ale nedostane se k ní uživatelský vstup (je částečně překryta jiným, například průhledným oknem apod.). Při kritickém nedostatku paměti může být taková aktivita zrušena.

- Shut down – aktivita je již ukončena Activity Managerem a nezabírá žádnou paměť.

Mezi těmito stavy aktivita přechází pomocí metod, nicméně i metody na obrázku 14 jsou pouze základní. Při programování běžné aplikace se programátor nevyhne přepsání těchto metod vlastními, protože aktivita se ruší a znovu vytváří už například při změně orientace displeje či vysunutí hardwarové klávesnice, a proto je potřeba na původní aktivitu správně navázat. Například k uložení vlastního stavu aktivity slouží další metoda onSaveInstanceState() a k získání stavu v nové aplikaci metoda onCreate() nebo onRestoreInstanceState(). [3]

Service umožňuje aplikaci běh na pozadí. I když bývá služba často spouštěna z aktivit, má svůj vlastní cyklus, který se nijak neváže na životní cyklus aktivity, jež ji spustila.



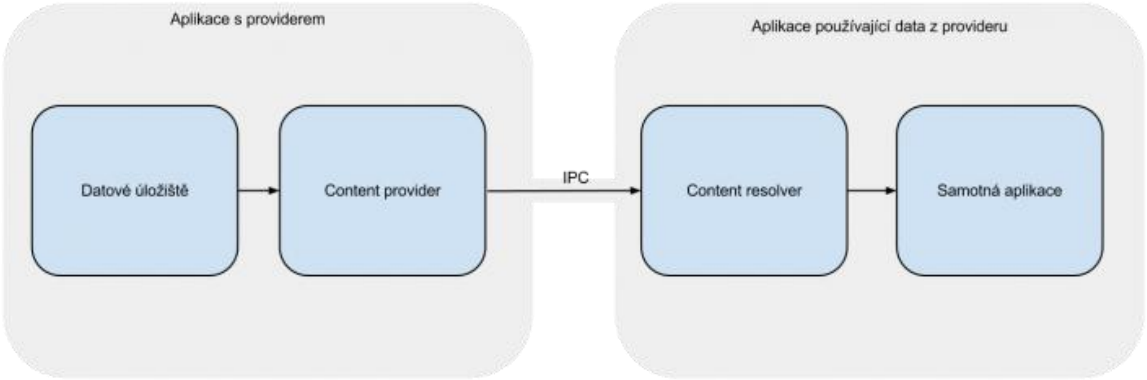
Obrázek 15 - Životní cyklus služby (převzato z [4])

Z obrázku č. 15 je patrné, že služba může vznikat a zanikat dvěma způsoby, podle jejího druhu.

- Spuštěná služba – služba se spouští voláním metody startService(). Tato služba může běžet na pozadí do nekonečna nezávisle na komponentě, která ji spustila. Tato služba nevrací výsledek volajícímu.
- Vázaná služba – služba se sváže s komponentou voláním metody bindService(). Taková služba pak klientovi nabízí rozhraní, jehož prostřednictvím ji může ovládat, získávat od ní informace apod. Tato služba běží pouze po dobu, dokud není zrušena komponenta, jež je s ní svázána, nebo dokud s ní komponenta neukončí vazbu.

Ačkoliv se služba obecně odděluje na dva typy, lze ji spustit v obou variantách současně. Takže se služba dá nejprve spustit metodou startService() a následně na ní navázat metodou bindService(). To umožňuje získání služby, která může běžet do nekonečna a přitom poskytovat své rozhraní.

Content provider je způsob, jak v Androidu sdílet data mezi aplikacemi či jednotlivými aktivitami. Data v content provideru jsou jako v databázi uspořádána ve formě tabulky a tak aplikace i k těmto datům přistupují, podobně jako k databázi, pomocí standardních metod insert, delete apod., které mají stejnou funkci jako databázové metody. Jelikož content providery bývají z jiných aplikací, a každá aplikace běží ve svém vlastním procesu, musí se řešit komunikace mezi procesy. Tuto komunikaci mezi procesy zajišťuje třída ContentProvider společně s třídou ContentResolver, která má stejné rozhraní jako ContentProvider. V aplikaci se vytvoří ContentResolver a ten se postará o spuštění ContentProvideru s požadavkem na získání či přidání dat. ContentProvider získá tyto data z nějakého datového úložiště a předá je ContentResolveru, který je předá samotné aplikaci. Content provider se navenek tváří jako databáze s jednou nebo více tabulkami. Předávání dat mezi aplikacemi pomocí Content provideru a Content resolveru je znázorněn na následujícím obrázku 16.[6]



Obrázek 16 - Content provider (převzato z [6])

Broadcast reciver globálně naslouchají na nějakou událost a umožňují tak aplikacím reagovat například na připojení k internetu, nebo na příchozí hovor. K tomu stačí vytvořit událost, předat ji frameworku, a ten upozorní všechny broadcast receivery, které deklarovali, že se o danou událost zajímají.

2.3 Použité technologie

V této části budou blíže popsány technologie, které byly použity v praktické části této práce.

2.3.1 Google Maps

Google Maps je integrovaná aplikace a technologie, poskytovaná společností Google, která je pro nekomerční účely dostupná zdarma. Pro Android jsou poskytnuty SDK knihovny Google Api. Tyto knihovny zpřístupňují rozhraní Google Maps, které je možné použít v aplikacích.

Všechny aplikace pro Android musí být podepsány pomocí digitálního certifikátu, pro kterou vývojář drží soukromý klíč. Tímto klíčem je v případě map Google Maps API Key, který je jedinečný a poskytuje jednoduchý způsob jak identifikovat autora aplikace. Pro použití Google Maps je tedy třeba získat klíč Google Maps API Key. Tento klíč může být buď pro účel vývoje, nebo pro publikaci a je generován ke konkrétnímu počítači pro konkrétní projekt, což znamená, že pro vývoj jednoho projektu na dvou počítačích jsou zapotřebí dva klíče. Klíč ke všem dostupným verzím Google Maps API lze získat na adrese https://code.google.com/apis/console. Od 3. 12. 2012 se oficiálně „přestalo používat“ Google Maps Anroid v1 API, což znamená, že od 18. 3. 2013 již nelze získat klíč pro toto API, a jelikož Google Maps Android API v2 používá odlišný systém správy klíčů, stávající klíče z verze 1 nelze ve druhé verzi použít. Nicméně klíče, které byly získány, budou platit i nadále. Zatímco první verze API umožňovala bez klíče, nebo se špatným klíčem vykreslování komponent a veškerou práci s mapou s tím, že pouze nevykreslila podkladová data, novější verze bez zadání správného klíče neumožní nic. Novější API je o poznání jednodušší a práce s ním je mnohem rychlejší a efektivnější, ztrácí ale podporu starších zařízení s nižší verzí androidu.

Klíč se do aplikace zadává na různá místa podle verze. Zatímco v první verzi se klíč zadával přímo do layoutu aplikace, u novější se vyplňuje pouze do AndroidManifestu.[8]

2.3.2 Sqlite databáze

Sqlite databáze je Open Source databáze, která je dostupná v každém zařízení s operačním systémem Android. Pro její využití není třeba žádné nastavení ani administrace a standartní uživatel se k ní nedostane.

V operačním systému Android není žádná předpřipravená databáze, proto si ji musí programátor sám vytvořit. Ideální cestou k vytvoření či aktualizaci databáze je vytvoření potomka třídy SQLiteOpenHelper, ve které je třeba přepsat metody onCreate() a onUpgrade(). První slouží k vytvoření databázových tabulek a jejich naplnění, druhá pak pro změnu jejich struktury. Dotazy lze provádět za pomocí třídy SQLiteDatabase díky metodám execSQL(), kterých je hned několik variant. Ta nejjednodušší přijímá jeden parametr, který reprezentuje SQL dotaz. Ten je metodě předán ve formě proměnné typu String.

Dotaz do databáze může být následující:

```
db.execSQL("create table zaznam (_id INTEGER PRIMARY KEY AUTOINCREMENT,
nazev TEXT NOT NULL);");
```

Kde objekt s označením `db` reprezentuje instanci třídy SQLiteDatabase. V tomto případě je instance předávána metodě onCreate() třídy, která je potomkem třídy SQLiteOpenHelper. Metoda execSQL() přijímá řetězec, který vytváří tabulku "zaznam". Tato tabulka obsahuje následující sloupce:

_id – sloupec typu INTEGER. Reprezentuje primární klíč, což je jedinečný identifikátor jednoho řádku tabulky. Tento klíč je automaticky inkrementován o jedničku při každém vložení řádku do tabulky.

nazev – sloupec typu TEXT. Reprezentuje název záznamu. Při vyplnění řádku nesmí atribut "nazev" zůstat nevyplněn.

[5][9]

2.3.3 Bletooth

Bluetooth je otevřený standard pro bezdrátovou komunikaci propojující dvě a více elektronických zařízení. Vytvořen byl firmou Ericsson v roce 1994 jako bezdrátová náhrada za sériové drátové rozhraní RS-232.

Bluetooth je definován standardem IEEE 802.15.1. Patří do kategorie osobních počítačových sítí PAN. Vyskytuje se v několika verzích, z nichž je dnes nejvíce zastoupená verze 2.0. V poslední době bývají ale čím dál častěji zastoupeny i novější verze.

Zařízení se dělí dle výkonnosti následujícím způsobem:

Tabulka 3 - Bluetooth - třídy výkonosti

Třída	Maximální povolený výkon		Dosah
	mW	dBm	
Třída 1	100	20	~100
Třída 2	2,5	4	~10
Třída 3	1	0	~1

Dle standardů mají zařízení následující přenosové rychlosti:

Tabulka 4 - Bluetooth - verze

Bluetooth	
Verze	Přenosová rychlost [Mb/s]
1.2	1
2.0 + EDR	3
3.0 + HS	24
4.0	24

V nejnovější verzi 4.0 zůstala stejná přenosová rychlost, zlepšil se však dosah, který je až 100 m a snížila spotřeba elektrické energie. Dále přibyla podpora šifrování AES-128.[13]

Android Framework poskytuje přístup k funkcionalitě Bluetooth pomocí Android Bluetooth API a pro použití v aplikaci se do souboru AndroidManifest.xml musí přidat právo využívat Bluetooth přidáním řádku

```
<uses-permission android:name="android.permission.BLUETOOTH" />
```

Android Bluetooth API poskytuje následující funkce:

- Vyhledávání ostatních Bluetooth zařízení.
- Připojení k ostatním zařízením.
- Přenos dat prostřednictvím Bluetooth.
- Správu více spojení.
- Podpora RFCOMM kanálu.
- Dotázat se na lokální spárovaná zařízení.

K tomu, aby se mohla dvě zařízení v Androidu spojit, musí být z bezpečnostních důvodů nejprve spárována.

Bluetooth API je dostupné v balíčku *android.bluetooth* a obsahuje například následující důležité třídy:

- BluetoothAdapter - reprezentuje lokální Bluetooth adaptér. Díky tomuto adaptéru se může zařízení vyhledávat ostatní zařízení s Bluetooth, nebo poskytnout již spárovaná zařízení.
- BluetoothDevice - představuje vzdálené Bluetooth zařízení. Prostřednictvím BluetoothScoket lze požádat o spojení s tímto zařízením.
- BluetoothSocket - podobně jako TCP soket představuje rozhraní pro správu Bluetooth spojení. Prostřednictvím objektů InputStream a OutputStream umožňuje výměnu dat mezi Bluetoth zařízeními.
- BluetoothServerSocket - podobně jako TCP Server soket představuje otevřený serverový soket., který sleduje příchozí požadavky.[14]

2.4 Diagramy

V této části práce bude formou diagramů představena samotná aplikace. Tvorba diagramů by měla předcházet programování každé aplikace, kvůli komplexnějšímu a promyšlenějšímu pohledu na problematiku, kterou má řešit.

2.4.1 Analýza požadavků

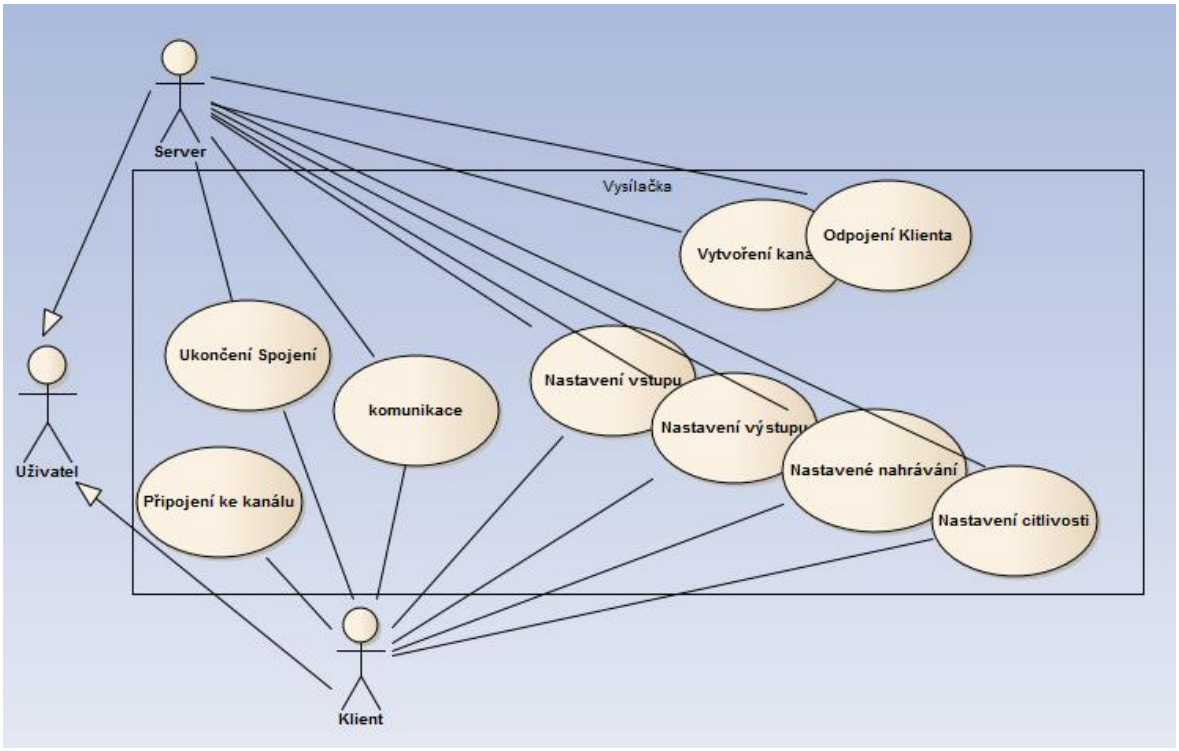
Začátkem návrhu aplikace je sběr požadavků, které aplikaci specifikují. Požadavky mohou být funkční či nefunkční. Funkční požadavky specifikují, co by měla aplikace obsahovat a jak by se měla v určitých situacích chovat. Nefunkční požadavky jsou vlastnosti a omezení služeb, které jsou aplikací poskytovány. Oba druhy požadavků jsou shrnuty v následující tabulce.

Tabulka 5 - Tabulka požadavků

Funkční požadavky	Nefunkční požadavky
Přenos hlasu	Spojení přes Bluetooth
Audiozáznam konverzace	Spojení alespoň 2 zařízení
Zap/Vyp vstupu (vlastní odchozí zvuk)	Běh aplikace na pozadí
Zap/Vyp výstupu (příchozá zvuk)	Intuitivní ovládání
Záznam trasy	Běh na Android 2.2 a vyšších
Zobrazení uložené trasy na mapě i grafem	Spolehlivý přenos dat na vzdálenost BT
Zobrazení grafu rychlosti v bodech trasy	Funkčnost v režimu Client-Server
Volba citlivosti zachycení zvuku	
Automatický Re-Connect	
Odpojení zařízení	
Přehrání záznamu v konkrétním bodě trasy	
Zobrazení uložených tras	
Zobrazení uložených záznamů	
Možnost smazání uložené trasy	
Možnost smazat uložený záznam	

2.4.2 Use Case diagram

Use Case diagram (Diagram užití) je diagram chování, který je definovaný v UML. Zachycuje vnější pohled na modelovaný systém a je dobrý k zobrazení funkčnosti aplikace, odhalení hranic systému a vztahu jednotlivých aktérů vůči aplikaci.



Obrázek 17 - Use Case Vysílačka

Z obrázku č. 17 je snadno viditelná část aplikace ovládající vysílačku. Je patrné, že uživatelé se tu dělí do dvou rolí, což vypovídá o tom, že vysílačka funguje v režimu Client - Server. K upřesnění slouží následující popis jednotlivých případů užití.

Vytvoření kanálu – Uživatel vytvoří kanál, který čeká na připojení ostatních zařízení. Automaticky se tak stává serverem.

Připojení ke kanálu – Uživatel se může připojit k nějakému zařízení, které otevřelo kanál a čeká na připojení jiných zařízení. Automaticky se tak stává klientem.

Ukončení spojení – Uživatel opustí konverzaci. Je-li uživatel klient, pouze se informace o jeho odpojení předá ostatním klientům, kteří jí patřičně zpracují. Je-li server, znamená to odpojení všech zařízení a zánik spojení.

Komunikace – Je-li navázáno spojení, může uživatel komunikovat s ostatními zařízeními v závislosti na aktuálním nastavení konverzace.

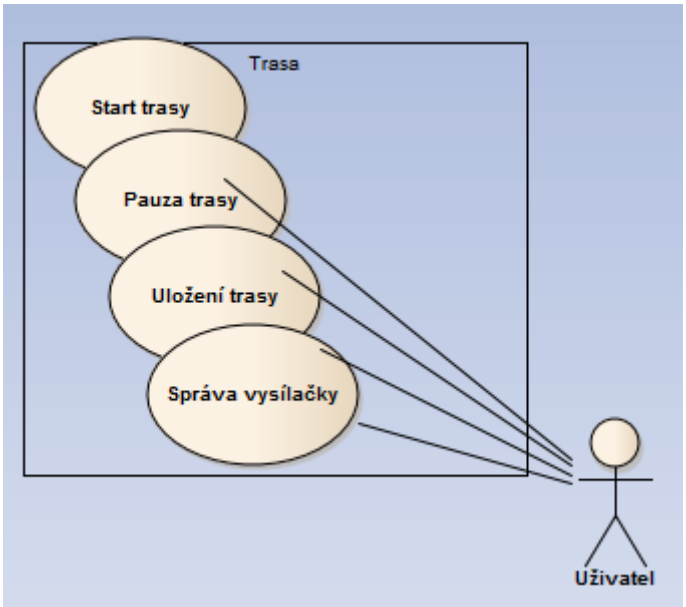
Odpojení klienta – Server může násilně odpojit připojeného uživatele. Klient může odpojení účastníka komunikace serveru pouze navrhnout a ten se o jeho odpojení rozhodne dle svého uvážení.

Nastavení vstupu – Uživatel volí, zda se chce účastnit komunikace vlastním vstupem. Touto volbou ovlivňuje, zda se mají odesílat data z mikrofonu ostatním uživatelům.

Nastavení výstupu – Uživatel volí, zda se chce účastnit komunikace vlastním výstupem. Touto volbou ovlivňuje, zda se mají přijímat data od ostatních účastníků komunikace.

Nastavení nahrávání – Uživatel volí, zda chce nahrávat konverzaci mezi účastníky konverzace.

Nastavení citlivosti – Uživatel si zvolí citlivost mikrofonu, takže při okolním hluku sníží citlivost, aby okolí zbytečně neovlivňovalo konverzaci.



Obrázek 18 - Use Case Trasa

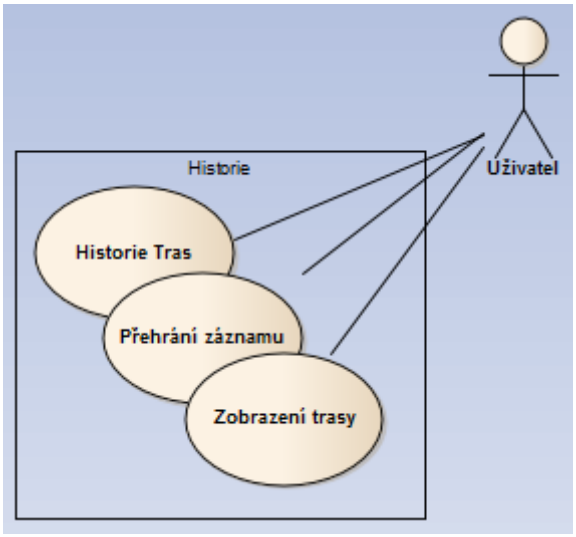
Obrázek 18 popisuje možnost zaznamenávání trasy, což je druhou funkcí aplikace. V této části je pouze jedna role uživatele, což však bývá u většiny aplikací pro mobilní zařízení. Popis jednotlivých případů užití je popsán níže.

Start trasy- Uživatel odstartuje zaznamenávání trasy.

Pauza trasy – Uživatel dočasně přeruší záznam trasy.

Uložení trasy – Po ukončení trasy je uživateli nabídnuto uložení zaznamenané trasy. Uživatel vyplní její podrobnosti a trasa je uložena do databáze.

Správa vysílačky – Uživatel může v průběhu záznamu trasy spustit vysílačku, jejíž prostředí je identické se samostatnou funkcí vysílačky. Navíc do prostředí záznamu trasy přibydou tlačítka pro rychlé ovládání konverzace.



Obrázek 19 - Use Case Historie

Obrázek č. 19 zobrazuje Use Case diagram, popisující historii. To je funkce aplikace, která uživateli umožňuje procházet zaznamenané trasy a konverzace. Konkrétní případy užité jsou popsány níže.

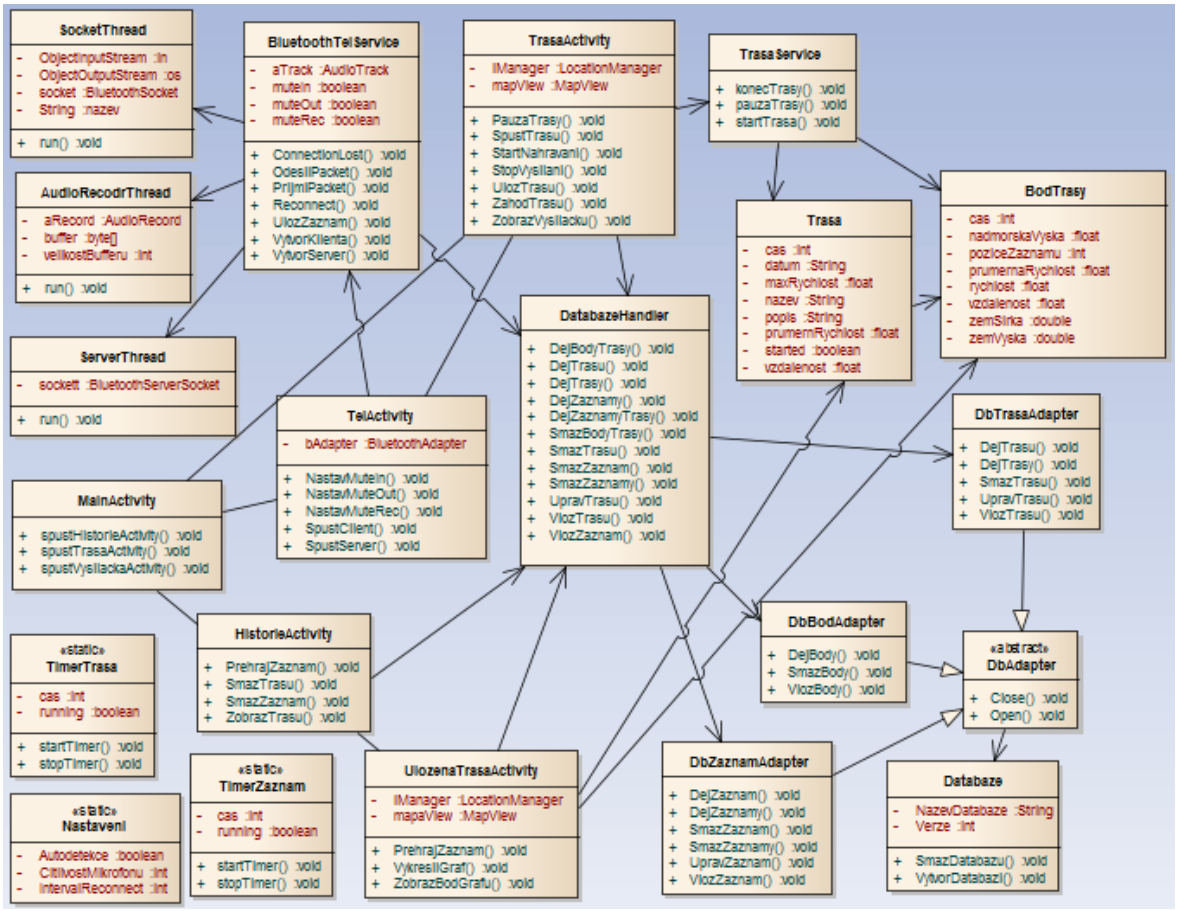
Historie tras – Uživatel má možnost nahlédnutí na historii tras a záznamů uložených uživatelem.

Přehrání záznamu – Uživatel má možnost si uložený záznam konverzace zpětně přehrát.

Zobrazení trasy – Uživatel si může zobrazit uloženou trasu. Zde je trasa zobrazena jak formou záznamu na mapě, tak grafem. Stisk do prostoru grafu automaticky odkáže na místo na mapě s podrobnými informacemi o konkrétním bodě.

2.4.3 Diagram analytických tříd

Diagram tříd představuje „statický pohled na modelovaný systém“. Zobrazuje strukturu objektových tříd a jejich vzájemné vazby v modelovaném systému.



Obrázek 20 - Diagram analytických tříd

Diagram tříd z obrázku č. 8 je dosti zjednodušený. Jak bylo uvedeno v kapitole Základní stavební prvky, každá komponenta, ať už se jedná o aktivitu, službu a podobně, prochází životním cyklem, který je doprovázen řadou metod. Těchto metod je velké množství, programátor je přepisuje a na diagramu nejsou kvůli jeho přehlednosti zobrazeny.

V levém spodním rohu jsou tři statické třídy. Tyto třídy jsou dostupné pro celou aplikaci.

- Nastavení - nastavení aplikace.
- TimerZaznam – časovač měřící dobu nahrávaného záznamu.
- TimerTrasa – časovač měřící dobru zaznamenávané trasy.

MainActivity je aktivita reprezentující úvodní obrazovku. Ta slouží jako rozcestník do dalších tří aktivit.

- TelActivity – reprezentuje prostředí vysílačky a váže se se službou TelService.
- TrasaActivity – reprezentuje prostředí záznamu trasy a váže se se službou TrasaService.
- HistorieActivity – reprezentuje prostředí uložených záznamů a tras. Umožňuje přehrávání záznamu a přepnutí na aktivitu s uloženou trasou.

Konkrétní data jsou reprezentována objekty následujících tříd:

- Trasa – představuje jednu ujetou trasu.
- BodTrasy – představuje jeden bod na ujeté trase.

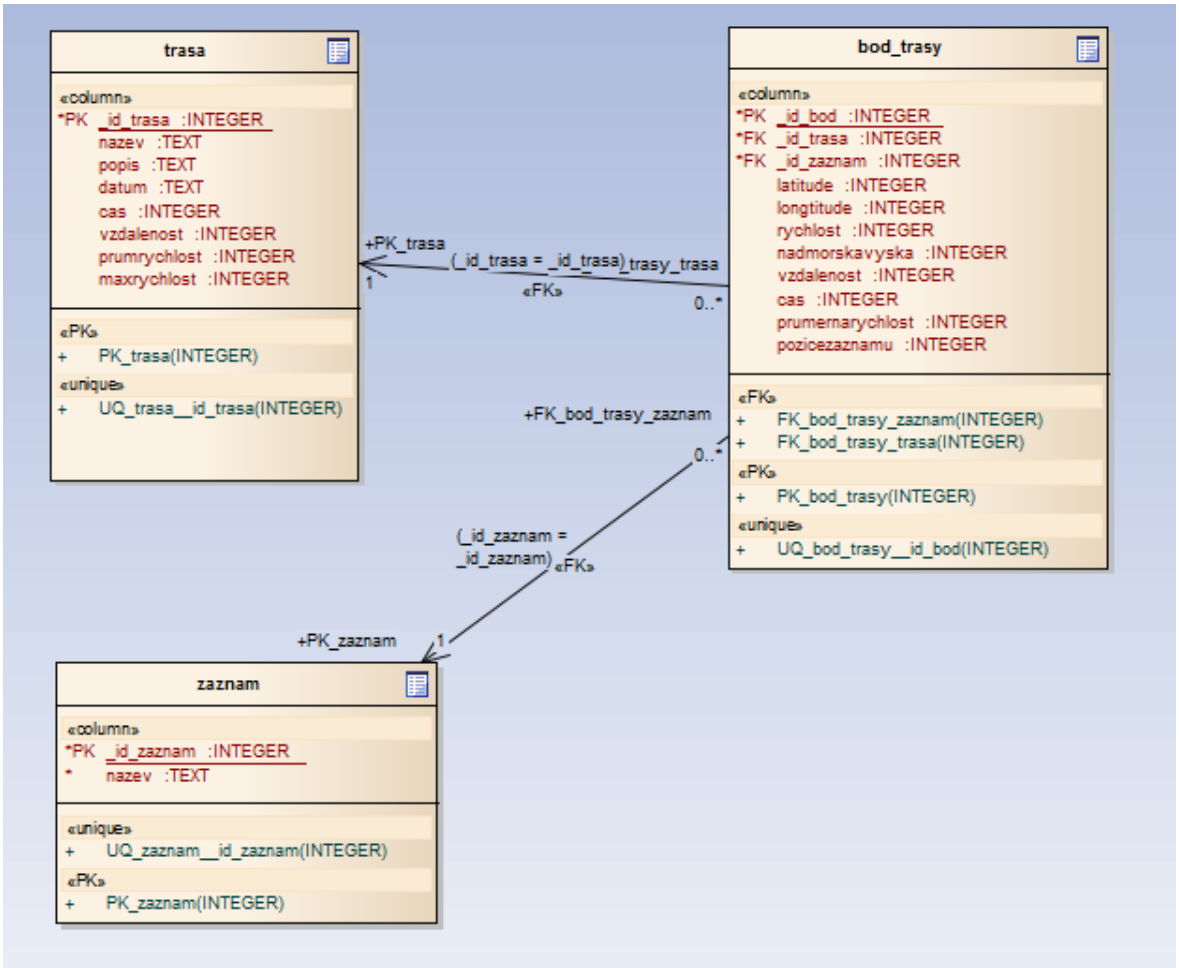
Třída `Databaze` se stará o vytváření a změnu struktury databáze. Abstraktní třída `DbAdapter` umožňuje otevření a uzavření databáze. Její potomci mají předepsané metody pro konkrétní databázové dotazy. Instanci těchto potomků vytváří jedna třída, `DatabazeHandler`, která volí jejich metody. Díky tomu lze snadno přistupovat k databázi prostřednictvím jedné jediné třídy.

Stěžejní část vysílačky je obsažena třídách `SocketThread`, `ServerThread` a `AudioRecordThread`. Všechny tři jsou potomky třídy `Thread` a umožňují běh ve svém vlastním vlákně, aby nebrzdili chod vlákna hlavního.

- ServerThread – naslouchá na portu a čeká na připojení nových uživatelů.
- SocketThread – při vytvoření spojení naslouchá na soketu na příchozí data.
- AudioRecordThread – naslouchá na mikrofonu, a když zachytí dostatečnou zvukovou hladinu, odešle paket se zvukem.

2.4.4 Datový model

Datový model představuje databázi, v níž jsou uložena data potřebná pro správný běh aplikace. Tabulky jsou zobrazeny na následujícím obrázku.



Obrázek 21 - Datový model

Z obrázku č. 21 je zřejmé, že datový model aplikace je velice jednoduchý a skládá se pouze ze tří tabulek. Vzhledem k použité databázi je zde využito jen pár základních datových typů, což znamená, že například položka datum musí být převáděna na text a podobně. Zato je však databáze celkově velice malá a je tedy možné ji použít i na mobilních zařízeních.

Tabulka `trasa` obsahuje uložené konkrétní zaznamenané trasy. Je v ní uložen název trasy, popis trasy, datum pořízení trasy, celkové trvání trasy, vzdálenost trasy, její průměrná a maximální rychlost. Primárním klíčem a zároveň jedinečným identifikátorem trasy je atribut `id_trasa`. Tento atribut je rostoucí číslo, automaticky generované při ukládání trasy.

Tabulka `bod_trasy` představuje záznamy o konkrétních záznamech trasy. Mezi tabulkou `trasa` a touto tabulkou platí vazba 1 k n, takže jedna trasa může obsahovat více bodů, zatímco jeden bod může být pouze v jedné trase. Stejná vazba je i mezi tabulkou `zaznam` a touto tabulkou, takže jeden uložený záznam může být ve více bodech. Tabulka `bod_trasy` obsahuje informace o zeměpisné šířce, výšce, aktuální rychlosti, nadmořské výšce a vzdálenosti od startu trasy. Tyto hodnoty jsou získávány z GPS senzoru. Navíc jsou ale dopočítávány informace o aktuální průměrné rychlosti v bodě, aktuální čas trasy

a aktuální pozice nahrávaného záznamu v tomto bodě. Cizí klíč `id_zaznamu` a hodnota pozice záznamu společně identifikují, v jakém záznamu a v jakém čase záznamu se tento bod zaznamenal, takže je pak snadné přehrát záznam od konkrétní pozice na trase. Cizí klíč `id_trasa` identifikuje trasu, do které tento bod patří a primární klíč `id_bod` je obdobně jako u tabulky `trasa` jedinečné, automaticky se zvětšující číslo jednoznačně identifikující konkrétní bod.

Poslední je tabulka `zaznam`. Ta uchovává informace o uložených zvukových záznamech. Pro tuto aplikaci postačuje jedinečný identifikátor záznamu, který se generuje obdobně jako identifikátory v předchozích dvou tabulkách, a název uloženého záznamu.

3 Implementační část

Základní struktura projektu je vidět na obrázcích 12 a 13 v analytické části. Pro Android aplikaci je stěžejní obsah souboru AndroidManifest.xml, který ji popisuje. Tento soubor se generuje automaticky při založení nového projektu a postupně se do něj připisují elementy dle potřeby aplikace.

3.1 AndroidManifest

Následující kus zdrojového kódu je kořenovým elementem manifestu.

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.diplomovaprace"
    android:versionCode="1"
    android:versionName="1.0" >
</manifest>
```

Atribut `package` představuje hlavní balíček, který je zároveň identifikátorem aplikace. Díky této definici se u popisu dalších komponent z tohoto balíčku může místo celé jeho cesty využít pouze. Další dva atributy se týkají aktuální verze aplikace, což je užitečné například při aktualizaci (nahrazování starší verze novější) aplikace z Google Play.

Následně je do manifestu nutné vepsat element, určující minimální verzi SDK a verzi, pro kterou je zařízení navrženo. Jako `targetSdkVersion` je dobré zadávat co nejvyšší možnou verzi. Je-li uvedena nižší, používají i novější zařízení například nevhodně starý vzhled. Je-li `minSdkVersion` příliš nízké číslo, nedají se v aplikaci používat technologie pro novější systém.

```
<uses-sdk
        android:minSdkVersion="5"
        android:targetSdkVersion="17" />
```

AndroidManifest dále obsahuje elementy <permission>, které jsou potřebné pro správný běh aplikace. Běžný uživatel tyto práva potvrzuje při stahování aplikace z Google Play a získává díky tomu přehled o tom, k čemu má aplikace přístup. Aby aplikace uživatele neodradila, měla by těchto oprávnění mít co nejméně. Manifest aplikace obsahuje následující práva:

```
<permission
android:name="com.example.diplomovaprace.permission.MAPS_RECEIVE"
android:protectionLevel="signature" />
<uses-permission
android:name="com.google.android.providers.gsf.permission.READ_GSERVICES"
/>
<uses-permission
android:name="com.example.diplomovaprace.permission.MAPS_RECEIVE"/>
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"
/>
```

```
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"
/>
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION"
/>
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"
/>
<uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />
<uses-permission android:name="android.permission.BLUETOOTH" />
<uses-permission android:name="android.permission.RECORD_AUDIO" />
<uses-permission android:name="android.permission.WAKE_LOCK" />
```

Někdy je omezení množství práv obtížné, protože jen Google Maps vyžadují prvních 9 z těchto práv. Další práva jsou potřeba pro vysílačku a často je podle atributu `name`, jehož hodnotou je název daného oprávnění jasné, k čemuž slouží. Právo WAKE_LOCK umožňuje uchovat aplikaci „probuzenou“, takže displej nezhasne, a aplikace je udržována aktivní. Toho se využívá při záznamu trasy.

Google mapy pro svůj chod potřebují ještě další dva elementy.

```
<uses-feature
        android:glEsVersion="0x00020000"
        android:required="true" />
```

a

```
<meta-data
            android:name="com.google.android.maps.v2.API_KEY"
            android:value="AIzaSyCKX2Z7TXrS_8wtekicNlmmyDxJPaRisJg" />
```

Druhý z elementů je právě API klíč, popsaný v kapitole o Google Maps a musí být pod elementem elementu `<application>`.

Element `<application>` definuje obsah aplikace. Je v něm definována ikona aplikace (`icon`), popisek aplikace (`label`), a motiv aplikace (`theme`).

```
<application
        android:icon="@drawable/ikona_img"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
</application>
```

V elementu `<application>` jsou zároveň definovány všechny aktivity, služby, content provery a broadcast receivery, jako jeho podelementy. Tato aplikace obsahuje následující aktivity.

```
<aktivity android:name=".activity.MainActivity"
android:label="@string/app_name" >
   <intent-filter>
      <action android:name="android.intent.action.MAIN" />
      <category android:name="android.intent.category.LAUNCHER" />
   </intent-filter>
</activity>
<activity
   android:name=".activity.DeviceListActivity"
   android:configChanges="orientation|keyboardHidden"
   android:label="@string/select_device"
```

```
   android:theme="@android:style/Theme.Holo.Dialog" />
<activity android:name=".activity.BluetoothTelActivity" ></activity>
<aktivity android:name=".activity.GpsActivity"
   android:label="@string/app_name"
   android:theme="@android:style/Theme.NoTitleBar" ></activity>
<activity android:name=".activity.TrasaActivity" ></activity>
<activity android:name=".activity.TrasyListActivity" ></activity>
<activity android:name=".activity.TrasaUlozenaActivity" ></activity>
<activity android:name=".activity.mapa" ></activity>
<activity android:name=".activity.MapaLocation" ></activity>
```

Jako název aktivity se do atributu `name` používá název aktivity, včetně balíčků, ve kterých je umístěna. Jelikož je v úvodní části manifestu deklarován hlavní balíček a všechny aktivity jsou umístěny v něm, mohou se jejich jména psát bez tohoto balíčku. Aktivita MainActivity obsahuje navíc element Intent-filter. Tento element je způsob, jak může nějaká aktivita (service, broadcast receiver) dát najevo, že je schopna splnit nějaký implicitní typ intentu. Kombinace atributů `"android.intent.action.MAIN"` a `"android.intent.category.LAUNCHER"` je součást automaticky generovaného intent filtru generované aktivity při vytváření projektu. Tato kombinace znamená, že se tato aktivita zobrazí v seznamu aplikací.

Posledními elementy v souboru AndroidManifest jsou následující dvě služby.

```
<service android:name=".service.GpsTrasaService" ></service>
<service android:name=".service.BluetoothTelService" ></service>
```
[15]

3.2 res

Adresář "res" obsahuje grafickou část aplikace. Kromě obrázků jsou v něm xml soubory, popisující vzhled jednotlivých komponent. Vzhled aktivit je popsán xml soubory v podadresáři layout. Následující kód popisuje jednoduchý layout popisující obrazovku, na které je zaznamenávána běžící trasa. Kořenovým elementem je LinearLayout, obsahující element fragment, který obsahuje mapu. Vertikálně pod ním je další vertikální LinearLayout s pod elementem `<include>`, který umožňuje nahrání vzdáleného layoutu na místo tohoto elementu.

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="match_parent" android:layout_height="match_parent"
android:orientation="vertical" >
      <fragment android:id="@+id/mapFragment"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            class="com.google.android.gms.maps.SupportMapFragment"/>
        <LinearLayout android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:orientation="vertical" >
            <include layout="@layout/trasa_run_info_layout" />
        </LinearLayout>
</LinearLayout>
```

Dalším typickým podadresářem adresáře res je adresář `values`. Ten obsahuje například soubor `strings.xml`, ve kterém jsou texty, používané v aplikaci. Ten může mít různé

koncovky a telefon si podle své lokalizace vybere soubor s jazykem, který potřebuje. V následujícím kódu je krátká ukázka souboru strings.xml z této práce.

```
<resources>
    <string name="vysilacka">Vysílačka</string>
    <string name="start">Start</string>
    <string name="stop">Stop</string>
    <string name="pause">Pauza</string>
</resources>
```

3.3 R.java

Ve složce gen je soubor R.java, který je automaticky generován a programátor by do něj neměl zasahovat. Soubor R.java obsahuje automaticky vygenerované klíče všech komponent deklarovaných v xml souborech adresáře `res`, proto tyto soubory nesmí obsahovat chybu a v opačném případě není soubor vygenerován. Zjednodušeně by se dal popsat jako most mezi programovou částí, reprezentovanou Java třídami a grafickou částí, reprezentovanou obrázky a xml soubory. Následující kód ukazuje část tohoto souboru, který identifikuje některé obrázky.

```
public static final class drawable {
        public static final int pause_img=0x7f020022;
        public static final int phone_green_img=0x7f020023;
        public static final int phone_red_img=0x7f020024;}
```

3.4 src

Obsahem složky `src` jsou Java soubory, představující funkční část aplikace. V následujících odstavcích budou shrnuty některé konstrukce, které byly v aplikaci použity.

Pro spojení dvou zařízení prostřednictvím Bluetooth musí serverová část aplikace naslouchat na objektu třídy `BluetoothServerSocket` a při připojení jiného zařízení se vytvoří soket pro komunikaci s ním. Tento proces je vidět na následující části kódu.

```
BluetoothServerSocket serverSocket = null;
try {
serverSocket=bAdapter.listenUsingRfcommWithServiceRecord(NAME, MY_UUID);
} catch (IOException e) {
e.printStackTrace();
}
BluetoothSocket socket = serverSocket.accept();
```

Na klientské části aplikace se odehrává následující část kódu.

```
BluetoothSocket bSocket;
try {
bSocket = device.createRfcommSocketToServiceRecord(MY_UUID);
} catch (IOException e) {
e.printStackTrace();
}
bAdapter.cancelDiscovery();
```

```
try {
bSocket.connect();
} catch (IOException e) {
e.printStackTrace();
}
```

MY_UUID je 128 bitový univerzálně jedinečný identifikátor aplikace. Existuje několik způsobů, jak tento identifikátor získat, ne všechny jsou však kompatibilní se všemi typy zařízení. V aplikaci se osvědčil následující způsob generování.

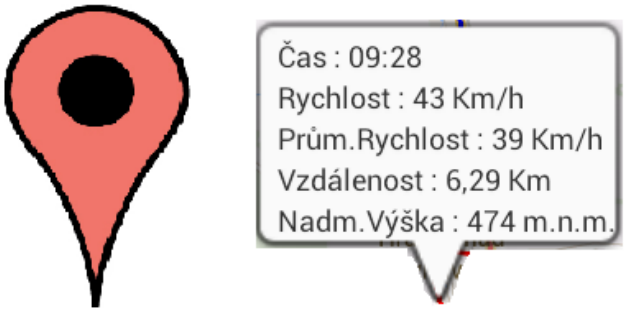
```
private static final UUID MY_UUID = new UUID(-1375618032L, -890933216L);
```

Další důležitou částí pro vysílačku jsou kódy řešení auto detekce zvuku a převodu ".raw" na ".wav". Kódy těchto procesů jsou ve zjednodušené formě v příloze A a B.

V režimu prohlížení uložené trasy bylo složitější vyřešit zobrazování komponenty Marker, s vlastním vzhledem, na mapě. Tento Marker zobrazuje na mapě informace o konkrétním bodu trasy a bylo kvůli němu nutné převést layout na bitmap, a následné použití na místo ikony Markeru. Tento převod je řešen v následující metodě.

```
public static Bitmap createDrawableFromView(Context context, View view) {
DisplayMetrics displayMetrics = new DisplayMetrics();
((Activity)context).getWindowManager().getDefaultDisplay().getMetrics(dis
playMetrics);
view.setLayoutParams(new LayoutParams(LayoutParams.WRAP_CONTENT,
LayoutParams.WRAP_CONTENT));
view.measure(displayMetrics.widthPixels, displayMetrics.heightPixels);
view.layout(0, 0, displayMetrics.widthPixels,
displayMetrics.heightPixels);
view.buildDrawingCache();
Bitmap bitmap = Bitmap.createBitmap(view.getMeasuredWidth(),
view.getMeasuredHeight(), Bitmap.Config.ARGB_8888);
Canvas canvas = new Canvas(bitmap);
view.draw(canvas);
return bitmap;
}
```

Díky tomuto kódu bylo možné nahradit standardní ikonu markeru layoutem, převedeným na obrázek. Původní a nová ikona jsou na následujícím obrázku č. 22.



Obrázek 22 - Ikona Markeru

4 Uživatelská část

V této části bude popsána aplikace z uživatelského pohledu. Aplikace je z důvodů popsaných v rešerši výše vytvořena pro operační systém Android, bohužel však kvůli poplatku za Obchod Play nebude na oficiálním úložišti aplikací pro Android dostupná.

4.1 Rozbor aplikace

Jak ze zadání a v úvodní části vyplývá, bylo cílem vyvinout aplikaci umožňující hlasovou komunikaci přes Bluetooth. Tohoto cíle bylo dosaženo. Aplikace je schopna komunikovat mezi dvěma a více zařízeními technikou Klient-Server, což znamená, že jedno zařízení musí nejprve vytvořit kanál, ke kterému se ostatní připojují. Zařízení mezi sebou komunikují pomocí paketů. Ty obsahují informace o druhu paketu, zvuková data a další informace potřebné ke komunikaci. Každé zařízení si musí pro komunikaci s ostatními vytvořit soket, s jehož pomocí jsou přeposílána data. Klient má otevřen soket pro komunikaci se serverem, server pak má soket pro každého klienta. Komunikace mezi všemi funguje tak, že klient odešle paket serveru a ten ho podle potřeby rozešle ostatním klientům. Server může navíc filtrovat účastníky komunikace a podle potřeby zařízení odpojovat. Oproti tomu klient může pouze navrhnout nějaké zařízení na odpojení a poslat požadavek serveru, který odpojení musí provést sám. Každé zařízení může jednoduše ovlivňovat, zda chce odesílat vlastní zvuková data ostatním zařízením, či jestli chce zvuková data od ostatních zařízení reprodukovat. Navíc umí aplikace snadno komunikaci nahrávat do souboru s koncovkou `".wav"`, takže jde jednoduše přehrát v aplikaci i mimo ni.

4.2 Rozšíření aplikace

Předchozím odstavcem bylo naplněno zadání této práce, která však byla původně cílena především pro skupinu motorkářů, z důvodu ceny profesionálního řešení interkomu. Samotná vysílačka byla rozšířena o možnost regulovat citlivost, se kterou se snímá zvuk. Užitečnou funkcí je i automatické znovu připojování, takže se aplikace sama pokouší znovu připojit k zařízení v roli serveru v pravidelných intervalech v případě, že je, z důvodu například překročení maximální vzdálenosti mezi zařízeními, násilně odpojena. Dále byla, jak už bylo zřejmé z analytické části, aplikace rozšířena o možnost zaznamenávání ujeté trasy. K tomu je třeba mít zapnuté získávání polohy ze satelitů GPS. Každou vteřinu se pak zaznamenává aktuální poloha. Původní myšlenkou bylo zaznamenávat trasu při každé změně pozice, zde by však nastal problém se ztrátou GPS signálu. Navíc může uživatel kdykoliv během jízdy spustit vysílačku a komunikovat se spolujezdcem i v tomto režimu. Tuto komunikaci lze pak snadno zaznamenávat. Záznamy uložené prostřednictvím vysílačky lze zpětně přehrávat a uložené trasy prohlížet. Při prohlížení uložené trasy se zobrazí souhrnné informace, jako je vzdálenost, čas apod. Navíc je vykreslena do mapy a grafu závislosti rychlosti na čase, do kterého lze klinknout a vybraný bod se přenese do mapy, kde se na místě jeho polohy zobrazí i s podrobnějšími

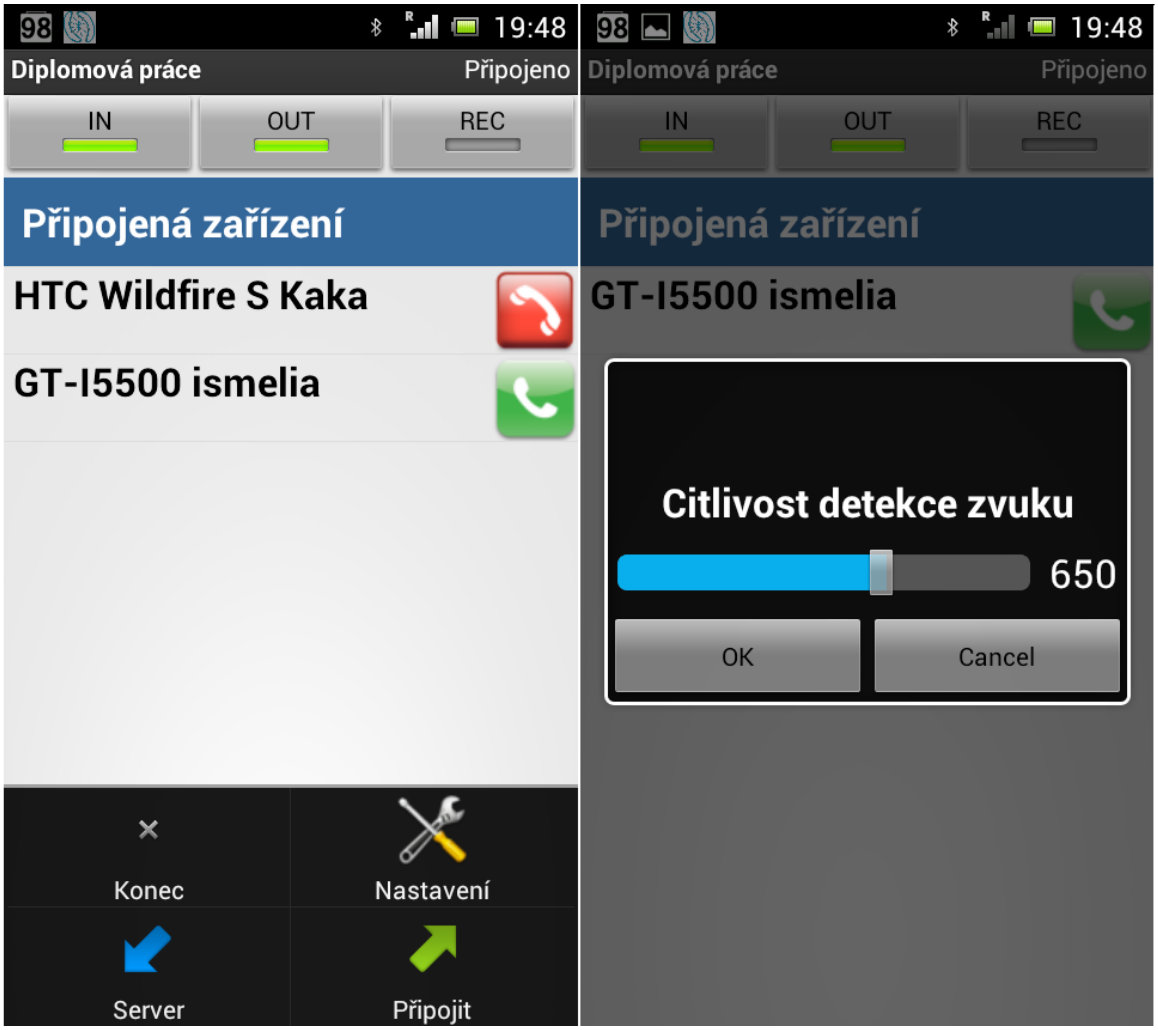
informacemi. Byl-li někde během trasy nahráván záznam, je tato část označena v mapě i grafu červenou barvou a záznam lze od pozice zvolené pozice přehrát.

4.3 Uživatelská příručka

V této kapitole bude popsáno uživatelské prostředí včetně ukázek aplikace.

Obrázek 23 - Úvodní obrazovka

Po spuštění aplikace se dostane do stavu na obrázku 23, na kterém je zobrazena úvodní obrazovka. Ta je zároveň rozcestník do dalších funkcionalit aplikace. Jsou na ní pouze 3 tlačítka, jejichž titulek napovídá o jejich funkci. První tlačítko spouští stěžejní funkci aplikace, kterou je samotná vysílačka. Prostředí vysílačky bude podrobněji popsáno níže. Druhé tlačítko se vztahuje na rozšířenou funkcionalitu aplikace a přesměruje nás na prostředí záznamu ujeté trasy. Nejspodnější tlačítko uživatele odkáže na obrazovku s historií.



Obrázek 24 - Vysílačka

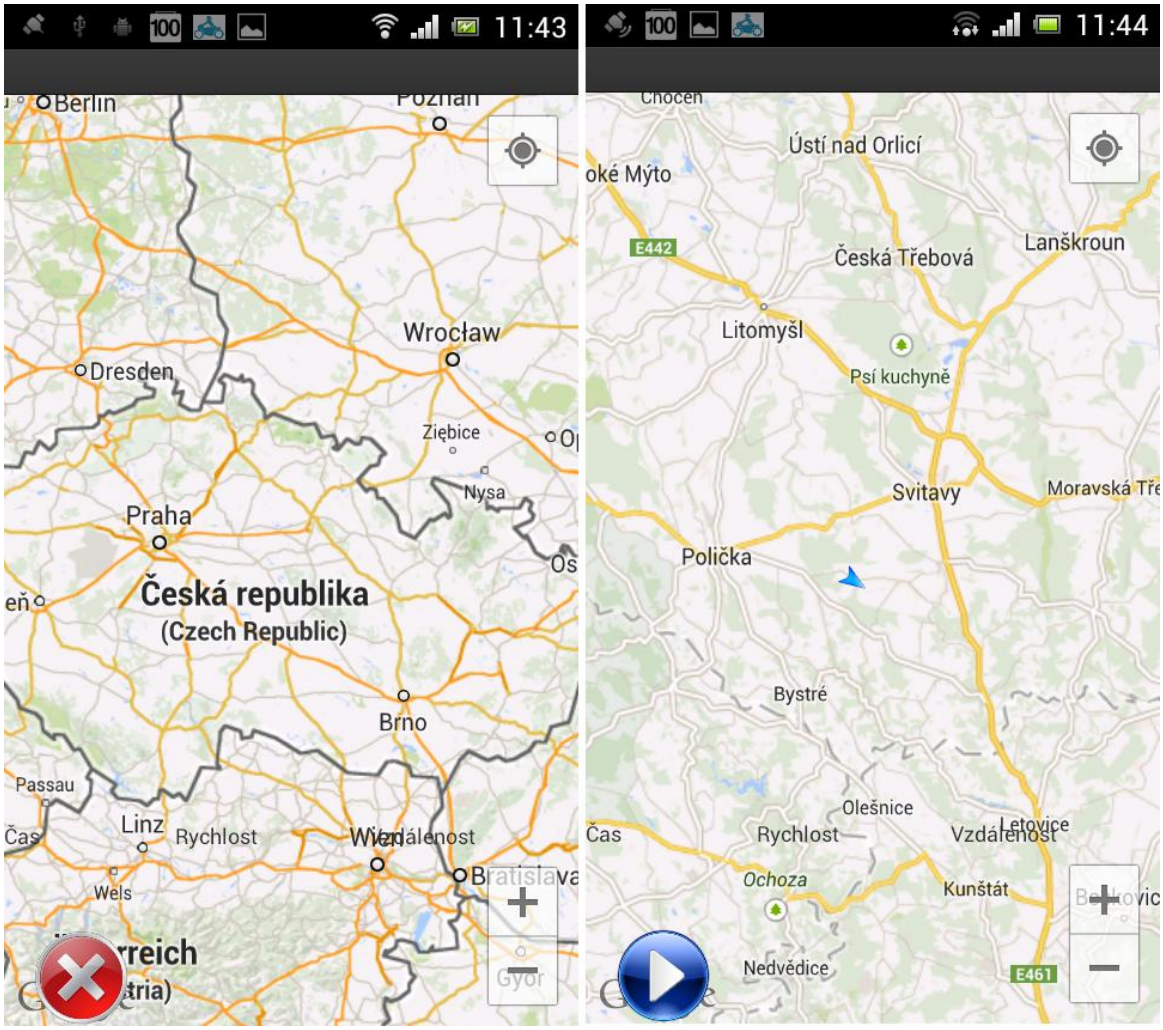
Ve chvíli, kdy uživatel spustí z úvodní obrazovky aplikace vysílačku, v první chvíli se ověřuje, zda má zapnuté Bluetooth. Je-li zapnuté, pokračuje se dále do vysílačky, v opačném případě je dotázán, zda chce Bluetooth zapnout. Nepotvrdí-li zapnutí, je nasměrován zpět na úvodní menu, v opačném případě se vysílačka spustí. Automaticky s ní se spouští i služba, která běží na pozadí a lze tak využívat funkcí vysílačky i ve chvíli, kdy zrovna není tato obrazovka na popředí zařízení.

Na Obrázku 24 je v horní části text s aktuálním stavem připojení. Pod tímto titulkem je trojice tlačítek. První zleva, s popiskem "IN" slouží k ovládání vstupu. Je-li aktivní, odesílá se audio zachycení mikrofonem ostatním účastníkům komunikace. Druhé tlačítko, s titulkem "OUT", obdobně ovládá výstup. Potvrzuje se jím, zda se má audio zachycené od ostatních účastníků komunikace reprodukovat. Poslední tlačítko, které je oproti ostatním aktuálně neaktivní, ovládá záznam audia. Aktivuje-li uživatel toto tlačítko, začne se zaznamenávat audio, které je zachytávané mikrofonem, i audio příchozí od ostatních zařízení do jednoho souboru na externí paměťovou kartu. Pod trojicí tlačítek je seznam s titulkem "Připojená zařízení". V tomto seznamu jsou názvy aktuálně připojených zařízení včetně ikony s telefonním sluchátkem. Pokud je ikona červená, jako u prvního připojeného

zařízení, znamená to, že zařízení aktuálně nevysílá žádný zvuk. Je-li ikona zelená, jako u druhého zařízení, tak toto zařízení aktuálně zvuk vysílá a pokud je aktivní tlačítko "OUT", je tento zvuk reprodukován. Na obrázku je zachycené i kontextové menu se čtyřmi položkami. První položka "Konec" ukončí vysílačku, včetně služby běžící na pozadí. Pokud uživatel opustí aplikaci jiným způsobem, zůstává služba aktivní a vysílačka běží na pozadí. Druhé tlačítko vyvolá dialog s možností nastavení citlivosti snímání zvuku. Tak lze snadno filtrovat hluk okolí, který uživatel nechce přenášet. Třetí položka "Server" spustí vysílačku v režimu server, takže se od této chvíle mohou ostatní zařízení připojit. Poslední položka "Připojit" vyvolá obrazovku se seznamem spárovaných zařízení a dostupných zařízení, které je v této části aplikace možné spárovat. Kliknutím na spárované zařízení se aktivuje připojování. Pokud je zařízení v dosahu, má zapnutou aplikaci, je navázáno spojení.

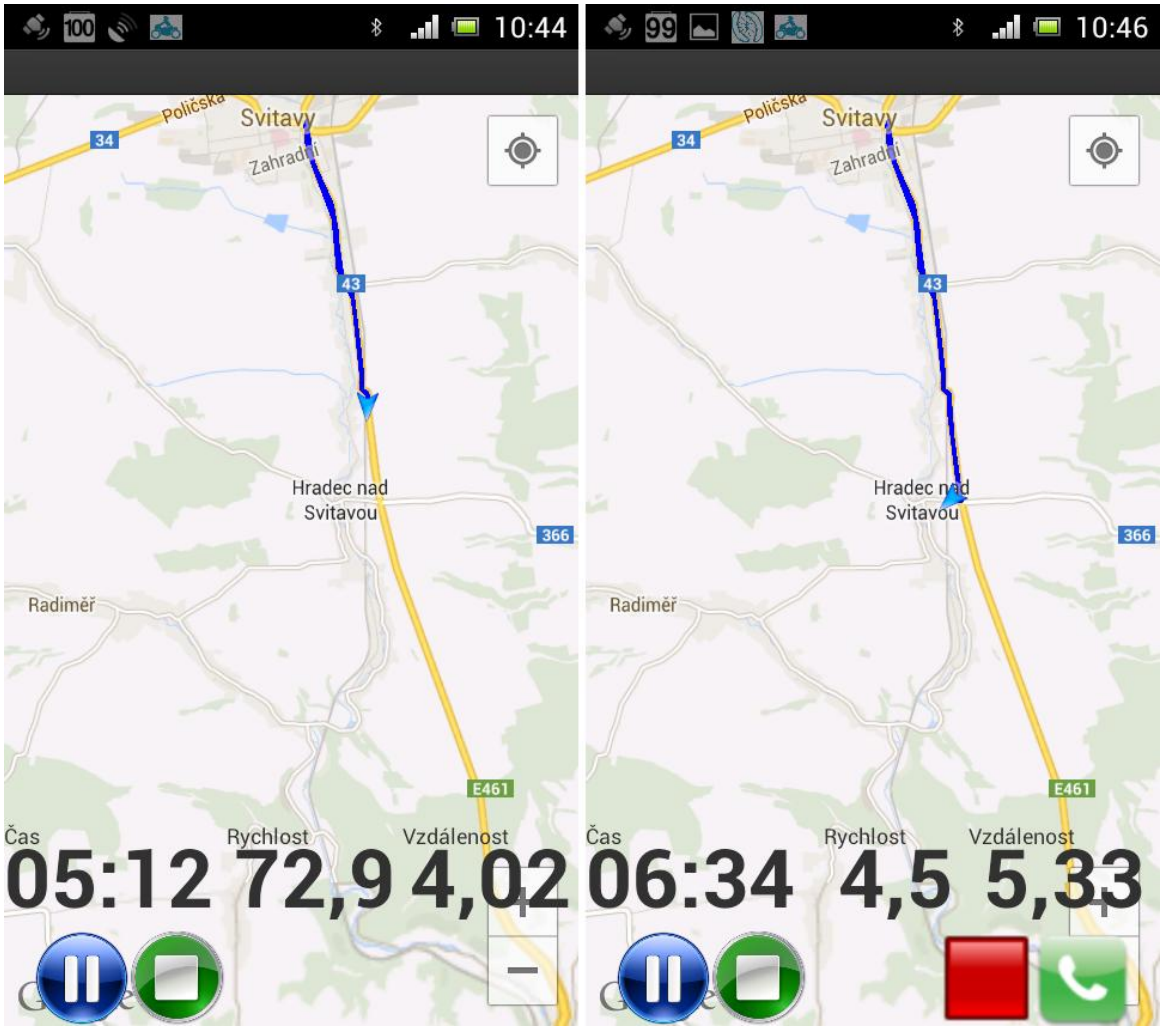
Po celou dobu, po kterou je aktivní služba vysílačky, je aktivní i notifikační ikona aplikace, přes kterou je možné se k vysílačce odkudkoliv navrátit.

Druhé tlačítko v úvodním menu uvede uživatele do prostředí trasy. Hned po kliknutí je vyzván k zapnutí GPS a v případě souhlasu je přesměrován na obrazovku Služeb určování polohy, kde může aplikaci povolit určování polohy pomocí systému GPS. Jelikož je určení polohy pro tuto část aplikace vyžadováno, je v případě odmítnutí zapnutí GPS aplikace ukončena. Obrazovka po zapnutí režimu trasy je vidět na následujícím obrázku č. 25.



Obrázek 25 - Úvodní obrazovka trasy

Po aktivaci režimu trasy se objeví levá obrazovka na obrázku č. 25. Ve stavovém řádku automaticky objeví notifikační ikona, signalizující běžící službu trasy a lze se díky ní do této obrazovky kdykoliv vrátit. Po dobu, kterou je uživatel v prostředí trasy, zůstává displej rozsvícený a sám nezhasne ani neztmavne. V levém spodním menu je ikona, signalizující stav připojení GPS. Ve chvíli, kdy má zařízení dostatečný GPS signál, změní se tlačítko v levém spodním rohu na ikonu, která je vidět na obrazovce vpravo. Toto tlačítko slouží ke spuštění záznamu trasy. Dále se obrazovka přesune automaticky na aktuální pozici telefonu, jež je signalizována modrou šipkou. V případě, že se telefon nepohybuje, změní se šipka na modrý kroužek. Odstartuje-li uživatel trasu stisknutím modrého tlačítka, objeví se obrazovka na následujícím obrázku č. 26 vlevo.

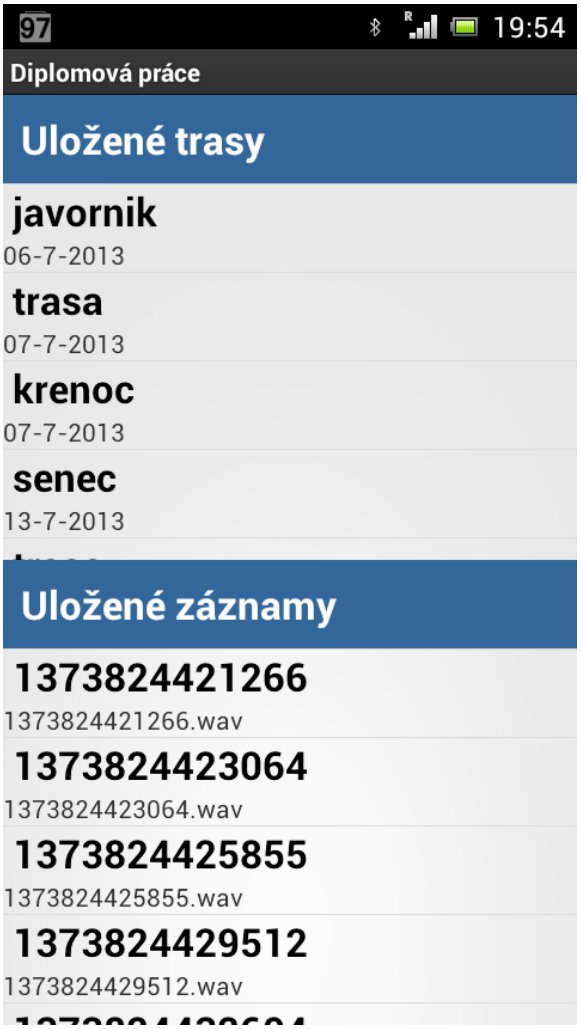


Obrázek 26 - Průběh trasy

Na obrázku č. 26 je na levé obrazovce vidět odstartovaná trasa. Tuto trasu lze přerušit a následně na ní navázat, nebo rovnou ukončit. Přes dialogové menu se uživatel může dostat do rozhraní vysílačky. Toto prostředí je identické s vysílačkou, spustitelnou z hlavního menu, takže má i stejné funkce a možnosti. Po jejím obsloužení a stisknutí tlačítka zpět je navrácen zpět na rozhraní trasy, ve kterém přibyla dvojice tlačítek, což je vidět na pravé obrazovce obrázku 26. V tuto chvíli je spuštěné zaznamenávání komunikace, proto je vedle tlačítka se sluchátkem červený čtvereček, který slouží k stopnutí tohoto záznamu. Není-li nahrávání aktivní, lze jej aktivovat stisknutím červeného kolečka, které se místo něj objeví. Tlačítko se sluchátkem souží pro navigaci do rozhraní spuštěné vysílačky.

Nad skupinou tlačítek je trojice textů. První z leva je čas ve formátu "hodiny:minuty:vteřiny". Druhý text je rychlost v Km/h a třetí je vzdálenost v kilometrech.

Stiskne-li uživatel poslední tlačítko na úvodní obrazovce aplikace, objeví se následující rozhraní historie.

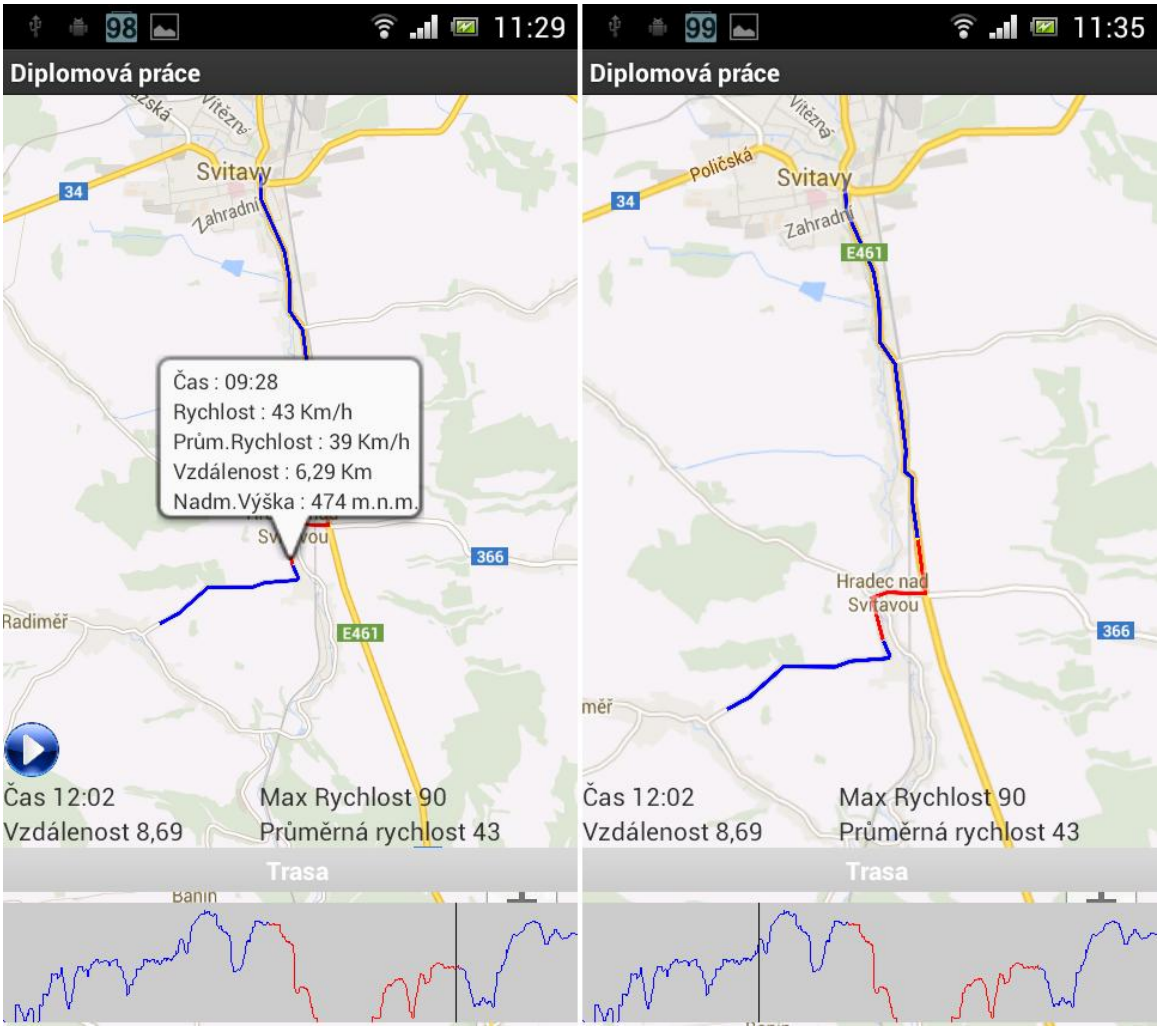


Obrázek 27 - Historie

Na obrázku 27 je vidět obrazovka historie se dvěma seznamy. První seznam obsahuje uložené trasy. Kliknutím na konkrétní trasu je uživatel přesměrován na obrazovku s trasou, stiskne-li uživatel trasu dlouze, je mu nabídnuto její odebrání. V případě, že s odebráním souhlasí, je odstraněna včetně jejich zvukových záznamů. Druhý seznam obsahuje záznamy, které byly uloženy v rámci trasy i při užívání samotné vysílačky. Při stisku se záznam začne přehrávat, případně se přehrávání přeruší. Při delším stisku je uživateli nabídnuto odstranění záznamu, a to jak z databáze, tak z paměťové karty, kde je záznam fyzicky uložen.

Na následujícím obrázku č. 28 je dvojice obrazovek, zobrazujících uloženou trasu, ke které se uživatel dostane po vybrání trasy v rozhraní historie. V pravé obrazovce je vidět čistá mapa s vykreslenou trasou. Modře je vykreslena klasicky ujetá trasa, červeně pak ta část, po kterou byla zaznamenávána konverzace z vysílačky. Na spodní části mapy jsou informace o uložené trase a pod těmito informacemi je vykreslen graf závislosti rychlosti

na čase. Ten je stejně jako trasa v mapě barevně odlišen. Klikne-li uživatel do prostoru grafu, objeví se u odpovídajícího bodu na mapě bublina s detailními informacemi o konkrétním bodu a mapa je kolem tohoto bodu vycentrována. Obsahuje-li daný bod nějaký záznam konverzace, automaticky se v levé části nad informacemi o trase objeví tlačítko, umožňující jeho přehrání. Po kliknutí do oblasti informační bubliny se automaticky zavře.



Obrázek 28 - Historie tras

Závěr

Cílem diplomové práce bylo vytvořit mobilní aplikaci, umožňující hlasovou komunikaci mezi dvěma chytrými telefony prostřednictvím Bluetooth. Pro aplikaci byl zvolen operační systém Android především pro jeho současnou oblíbenost a dostupnost. Proto bylo za osobní cíl kladeno detailnější seznámení se s vývojem pro tuto platformu. Práce se dotknula Bluetooth, GPS, Google Maps a SQLite databáze. Se všemi těmito technologiemi se v Androidu pracovalo poměrně dobře.

Prvotní seznamování s platformou bylo velice příjemné, díky velmi dobře zpracované dokumentaci a komunitě kolem Androidu, který se rychle rozrůstá, takže o inspiraci většinou nebyla nouze. Obrovským zdrojem informací a inspirace byl kromě dokumentace na stránkách http://developer.android.com/index.html server http://stackoverflow.com/.

Během vývoje se řešilo několik větších problémů. První se týkal vysílačky. Bluetooth se nedařilo rozchodit na všech zařízeních úplně korektně. Po vyřešení tohoto problému bylo potřeba vyřešit samotný princip fungování vysílačky a zvládnutí možnost komunikace mezi větším počtem zařízení.

Další problém se objevil při ukládání zvukového záznamu. Pro potřebu přistupovat k nahrávce na libovolnou pozici nestačilo ukládání v defaultním formátu ".raw", přizpůsobeným k záznamu hlasu, a musel se soubor programově převést do souboru ".wav".

Poslední větší problém nastal, když se z technických důvodů muselo přejít z Google Map verze 1 na novější verzi 2. Rozdíl mezi verzemi je tak obrovský, že to ovlivnilo značnou část aplikace. Naštěstí se s Google Maps verze 2 pracuje většinou o poznání rychleji a intuitivněji, a díky velké komunitě kolem je dostupná spousta návodů a příkladů.

Cíl práce byl naplněn vytvořením aplikace, jež disponuje funkcemi, které jsou potřebné pro plánovanou oblast použití. Navíc byla vysílačka rozšířena o schopnost záznamu ujetých tras, což je v této problematice využitelné rozšíření. Rozšíření aplikace bylo dobrovolné a z důvodů technických a časových nemusí být vždy úplně ideálně zpracované. Vysílačka však byla otestována a v porovnání s konkurencí je funkčně na dobré úrovni. Vhodné by bylo rozšířit ji o možnost fungování prostřednictvím wifi. Tato problematika však byla vynechána z důvodu vysokých nároků na verzi systému. Až na pár výpadků se pro Android programovalo velice pěkně a se zvládnutou angličtinou je pro programátora snadné získat si o své problematice potřebné informace.

Literatura

[1] Android (operační systém). In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2013-07-12]. Dostupné z: http://cs.wikipedia.org/wiki/Android_(opera%C4%8Dn%C3%AD_syst%C3%A9m)

[2] Android Architecture: The Key Concepts of Android OS. *Android-app-market* [online]. 2012 [cit. 2013-07-12]. Dostupné z: http://www.android-app-market.com/android-architecture.html

[3] Activity. *Developers* [online]. 2013 [cit. 2013-07-12]. Dostupné z: http://developer.android.com/reference/android/app/Activity.html

[4] Services. *Developers* [online]. 2013 [cit. 2013-07-12]. Dostupné z: http://developer.android.com/reference/android/app/Activity.html

[5] Android SQLite Database and ContentProvider: Tutorial. *Vogella* [online]. 16.06.2013 [cit. 2013-07-12]. Dostupné z: http://www.vogella.com/articles/AndroidSQLite/article.html

[6] KONEČNÝ, Matěj. Vyvíjíme pro Android: Content providery. *Zdrojak* [online]. 2012 [cit. 2013-07-13]. Dostupné z: http://www.zdrojak.cz/clanky/vyvijime-pro-android-content-providery/

[7] Bluetooth. *Developers* [online]. 2013 [cit. 2013-07-13]. Dostupné z: http://developer.android.com/guide/topics/connectivity/bluetooth.html

[8] Google Maps: Android API v2. *Developers* [online]. 2013 [cit. 2013-07-17]. Dostupné z: https://developers.google.com/maps/documentation/android/start?hl=cs-CZ#the_google_maps_api_key

[9] UJBÁNYAI, Miroslav. *Programujeme pro Android*. Vyd. 1. Praha: Grada, 2012, 187 s. Průvodce (Grada). ISBN 978-80-247-3995-3.

[10] PETŘEK, Pavel. Vývoj pro Android-II. *Zdroják* [online]. 2010 [cit. 2013-07-17]. Dostupné z: http://www.zdrojak.cz/clanky/vyvoj-pro-android-ii/

[11] KYPTA, Tomáš. Vyvíjíme pro Android: tvoříme aktivity. *Abclinux* [online]. 2011 [cit. 2013-07-17]. Dostupné z: http://www.abclinuxu.cz/clanky/vyvijime-pro-android-tvorime-aktivity#!/-1/

[12] IDC: Press Release. *Idc* [online]. 2013 [cit. 2013-07-22]. Dostupné z: http://www.idc.com/getdoc.jsp?containerId=prUS23946013#.US4Fvx0z2rG

[13] Bluetooth. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001-2013 [cit. 2013-07-22]. Dostupné z: http://cs.wikipedia.org/wiki/Bluetooth

[14] Bluetooth. *Developers* [online]. 2013 [cit. 2013-07-22]. Dostupné z: http://developer.android.com/guide/topics/connectivity/bluetooth.html

[15] <manifest>. *Developers* [online]. 2013 [cit. 2013-07-22]. Dostupné z: http://developer.android.com/guide/topics/manifest/manifest-element.html

Příloha A – Metoda převodu .raw na .wav

```
private void WriteWaveFileHeader(FileOutputStream out,
                long totalAudioLen, long totalDataLen, long
longSampleRate,
                int channels, long byteRate) throws IOException {

            byte[] header = new byte[44];

            header[0] = 'R';
            header[1] = 'I';
            header[2] = 'F';
            header[3] = 'F';
            header[4] = (byte) (totalDataLen & 0xff);
            header[5] = (byte) ((totalDataLen >> 8) & 0xff);
            header[6] = (byte) ((totalDataLen >> 16) & 0xff);
            header[7] = (byte) ((totalDataLen >> 24) & 0xff);
            header[8] = 'W';
            header[9] = 'A';
            header[10] = 'V';
            header[11] = 'E';
            header[12] = 'f';
            header[13] = 'm';
            header[14] = 't';
            header[15] = ' ';
            header[16] = 16;
            header[17] = 0;
            header[18] = 0;
            header[19] = 0;
            header[20] = 1;
            header[21] = 0;
            header[22] = (byte) channels;
            header[23] = 0;
            header[24] = (byte) (longSampleRate & 0xff);
            header[25] = (byte) ((longSampleRate >> 8) & 0xff);
            header[26] = (byte) ((longSampleRate >> 16) & 0xff);
            header[27] = (byte) ((longSampleRate >> 24) & 0xff);
            header[28] = (byte) (byteRate & 0xff);
            header[29] = (byte) ((byteRate >> 8) & 0xff);
            header[30] = (byte) ((byteRate >> 16) & 0xff);
            header[31] = (byte) ((byteRate >> 24) & 0xff);
            header[32] = (byte) (2 * 16 / 8);
            header[33] = 0;
            header[34] = RECORDER_BPP;
            header[35] = 0;
            header[36] = 'd';
            header[37] = 'a';
            header[38] = 't';
            header[39] = 'a';
            header[40] = (byte) (totalAudioLen & 0xff);
            header[41] = (byte) ((totalAudioLen >> 8) & 0xff);
            header[42] = (byte) ((totalAudioLen >> 16) & 0xff);
            header[43] = (byte) ((totalAudioLen >> 24) & 0xff);

            out.write(header, 0, 44);
    }
```

Příloha B – Princip auto detekce zvuku

```
while(true){
	float totalAbsValue = 0.0f;short sample = 0;
	pocetBytu = this.aRecord.read(buffer, 0, bufferSize);
	if (pocetBytu < 0)continue;
	//analýza zvuku
	for (int i = 0; i < bufferSize; i += 2) {
	sample = (short) ((buffer[i]) | buffer[i + 1] << 8);
	totalAbsValue += (float) Math.abs(sample)/((float)
	pocetBytu/(float) 2);}
	// analýza bufferu
	tempFloatBuffer[tempIndex % 3] = totalAbsValue;
	float temp = 0.0f;
	for (int i = 0; i < 3; ++i)
	temp += tempFloatBuffer[i];
	if ((temp >= 0 && temp <= citlivostMic) && recording == false) {
	tempIndex++;
	continue;}
	 (temp > citlivostMic && recording == false) {
	recording = true; }
	if ((temp >= 0 && temp <= citlivostMic) && recording == true) {
	if (prenos) {
	prenos = false;
	//konec komunikace
	}
	tempIndex++;
	break;}
	if (!prenos) {
	prenos = true;
	//začátek komunikace
	}
	//průběh komunikace
	tempIndex++;
	}
```

Příloha C – Přiložené CD

Přiložené CD obsahuje

- Zdrojové kódy aplikace
- Spustitelnou aplikaci
- Tuto práci ve formátu PDF a DOC