VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

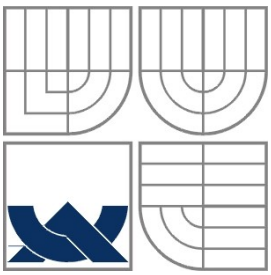
MOBILNÍ APLIKACE VYUŽÍVAJÍCÍ GPS MODUL

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

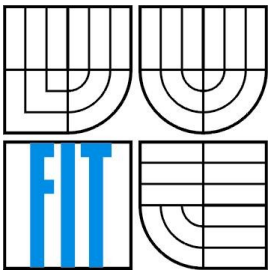
AUTOR PRÁCE
AUTHOR

PETR BLATNÝ

BRNO 2012



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

MOBILNÍ APLIKACE VYUŽÍVAJÍCÍ GPS MODUL

MOBILE APPLICATION USING GPS MODULE

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

PETR BLATNÝ

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. RADEK BARTOŇ

BRNO 2012

Abstrakt

Tato bakalářská práce se zabývá popisem návrhu a tvorby mobilní aplikace využívající GPS modul. Touto aplikací je lokalizační hra, kde hráčovým úkolem je projít za co nejkratší čas na mapě zobrazené body. Jako cílová mobilní platforma byl zvolen operační systém Android. Mapové podklady v aplikaci jsou volně šířené Open Street Maps. Práce se proto zabývá popisem platformy Android, mapových podkladů a systému GPS. Aplikace dále využívá SQLite databázi pro uložení perzistentních dat. Pro hru více hráčů na více zařízeních je využito platformy Gamooga, která umožňuje vzájemnou komunikaci zařízeních.

Abstract

This bachelor's thesis deals with the description of the design and creation of mobile application using GPS module. Application is location game where the player's task is to pass points shown on mam in the shortest possible time. As the target mobile platform was chosen operation system Android. Map data in the application are freely distributed Open Street Maps. The thesis describes Android platform, maps and GPS system. The application also uses SQLite database to store persistent data. For multiplayer game is used Gamooga platform which allows network communication between devices.

Klíčová slova

Android, Google, mobilní aplikace, databáze, SQLite, GPS, Java, OpenStreetMap.

Keywords

Android, Google, mobile application, database, SQLite, GPS, Java, OpenStreetMap.

Citace

Petr Blatný: Mobilní aplikace využívající GPS modul, bakalářská práce, Brno, FIT VUT v Brně, 2012

Mobilní aplikace využívající GPS modul

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením Ing. Radka Bartoně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

……
Petr Blatný
16. Května 2012

Poděkování

Rád bych poděkoval vedoucímu práce Ing. Radku Bartoňovi za poskytování užitečných nápadů a rad a za jeho trpělivost. Také bych rád poděkoval všem, kteří mi poskytli jejich mobilní telefon pro otestování aplikace.

Obsah

1 Úvod

S rozvojem techniky dochází k narůstání výkonu a minimalizace zařízení. Příkladem tohoto jsou dnes chytré mobilní telefony, které mají výkon jako klasické počítače před deseti lety. Proto také dochází k častému nahrazování klasických počítačů právě těmito zařízeními. Jejich výhodou je převážně mobilita, ale navíc často poskytují jednoduché a intuitivní ovládání, dnes často dotykové.

Součástí výbavy takovýchto zařízení bývá často GPS modul, který je využívám aplikacemi z mnoha odvětví. Mezi nejčastější použití patří bezesporu navigace a dále pak sportovní deníky měřící výkony uživatele. Dalším využitím jsou například aplikace pro pořizování fotek, jenž přikládají údaj o poloze právě pořízeného snímku.

Cílem této práce je vytvořit aplikaci využívající GPS modul pro mobilní platformu. Touto aplikací je lokalizační hra, kde hráčovým úkolem je projít na mapě zobrazené body za co nejkratší čas. Aplikace by měla poskytovat kompletní správu tratí, tedy jejich vytváření, editování i mazání. Dále umožní hrát hru pro jednoho i více hráčů. Aplikace využívá při zobrazení mapy podkladů Open Street Maps, které jsou volně dostupné. Aplikace by měla ukládat statistiky her a umožnit je uživateli zobrazit.

V následující kapitole je uveden popis systému Android. Jsou zde popsány součásti aplikace a jejich životní cyklus, jehož pochopení je klíčové pro správnou funkčnost aplikace. Dále je v této kapitole uveden popis systému GPS a projektu Open Street Map, které aplikace využívá. Na závěr kapitoly je popsán herní systém Gamooga použitý v aplikaci pro podporu hry více hráčů.

V následujících kapitolách je uveden postup návrhu a implementace aplikace. Je zde uveden popis databáze, kterou aplikace využívá pro uložení dat, popis návrhu a tvorby uživatelského rozhraní. Mimo jiné je zde uvedena funkčnost zobrazování mapových podkladů a využití prvků map a dále určování polohy zařízení pomocí systému GPS. Na závěr této kapitoly je popsána síťová komunikace pomocí zpráv.

V závěrečných kapitolách je popsáno testování výsledné aplikace a budoucí vývoj projektu – možnost rozšíření funkčnosti aplikace a její distribuce na Google Play.

2 Použité technologie

Tato kapitola se zabývá popisem součástí použitých pro vývoj aplikace. Popsána je platforma Android, na které je výsledný produkt postaven. Dále se kapitola zabývá popisem použitých mapových podkladů Open Street Map a GPS systémem.

2.1 Platforma Android

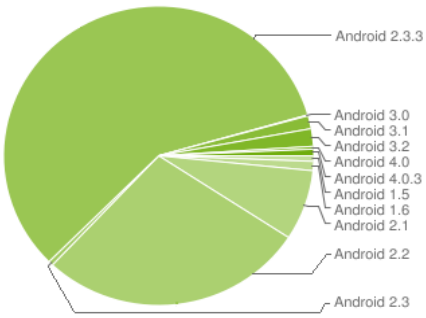
Android je open-source softwarová platforma určená převážně pro mobilní zařízení jako mobilní telefony, navigace, tablety. Obsahuje na linuxovém jádře založený operační systém, middleware[1] a uživatelské rozhraní a aplikace.

Android SDK[2] poskytuje nástroje a API potřebné pro vývoj aplikací pro platformu Android a využívá programovacího jazyka Java.

Android byl od roku 2003 vyvíjen společností Android Inc., kterou v roce 2005 odkoupila společnost Google. Dne 5. listopadu 2007 vzniklo uskupení Open Handset Aliance (OHA) s cílem vyvinout otevřený standard pro mobilní zařízení. Jeho členy jsou společnosti zabývající se výrobou mobilních čipů, telefonů nebo mobilních aplikací, jako Google, HTC, Intel, LG, Motorola, Nvidia, Qualcomm, Samsung a další. Stejný den byl ohlášen Android a o týden později bylo vydáno první SDK pro vývojáře.

Verze systému android

Od první verze systému bylo vydáno několik aktualizací opravující chyby a přinášející vylepšení a nové funkce. Pojmenování verzí je podle zákusků a názvy navíc začínají postupně na písmena abecedy. Aktuální rozložení všech uvolněných verzí na trhu je vidět na obrázku 2.1. Další předpokládaná verze Jelly Bean má být uvedena v polovině roku 2012.



Platform	Codename	API Level	Distribution
Android 1.5	Cupcake	3	0.6%
Android 1.6	Donut	4	1.0%
Android 2.1	Eclair	7	7.6%
Android 2.2	Froyo	8	27.8%
Android 2.3 - 2.3.2	Gingerbread	9	0.5%
Android 2.3.3 -2.3.7		10	58.1%
Android 3.0	Honeycomb	11	0.1%
Android 3.1		12	1.4%
Android 3.2		13	1.9%
Android 4.0 - 4.0.2	Ice Cream Sandwich	14	0.3%
Android 4.0.3		15	0.7%

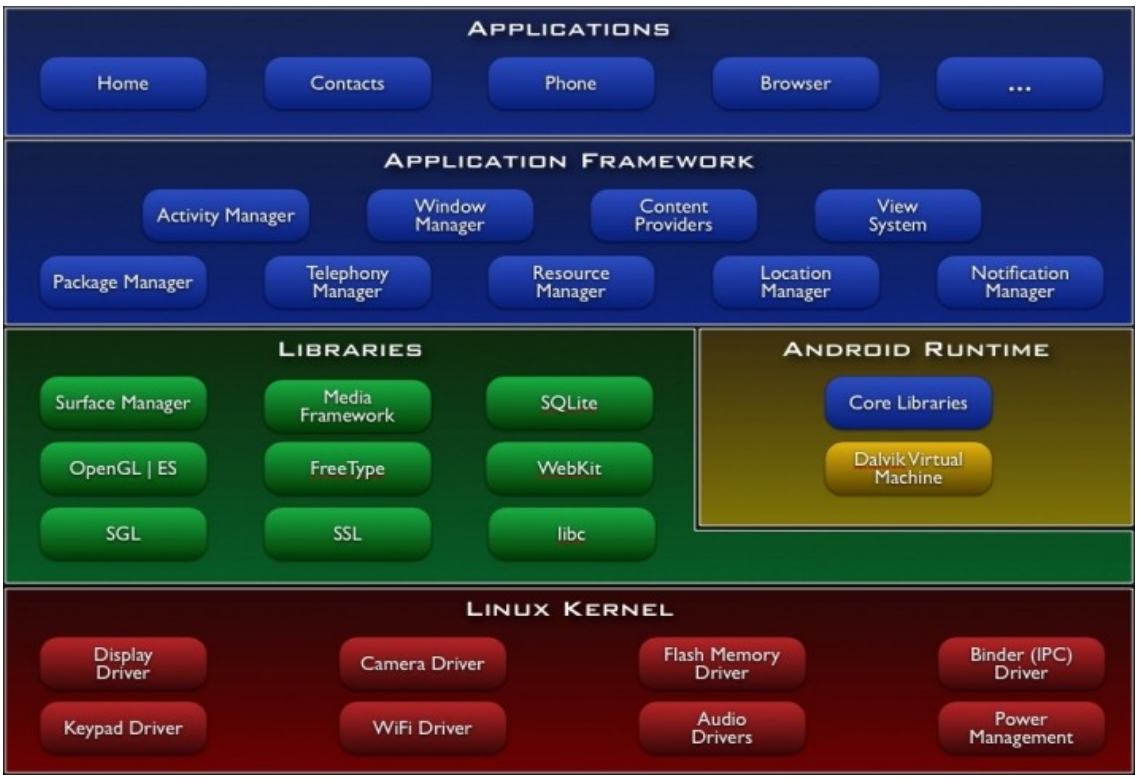
Obrázek 2.1: Graf a tabulka podílu verzí systému android k datu 21.2.2012 [3]

1 Vrstva mezi operačním systémem a aplikacemi.

2 Zkratka pro Software Development Kit

2.1.1 Architektura

Architektura Androidu sestává z vrstev zobrazených na obrázku 2.2. Kapitola čerpá z [2][4].



Obrázek 2.2: Architektura Android [2]

Linux kernel

Tato vrstva tvoří abstraktní vrstvu mezi hardware a software na vyšších vrstvách. Jádro operačního systému Android je založené na Linuxovém jádře 2.6 a využívá systémové služby jako zabezpečení, správu paměti a procesů, síťové služby, hardwarové ovladače a správu napájení.

Knihovny

Další vrstvou jsou knihovny napsané v programovacím jazyce C/C++. Nejedná se o samostatné aplikace, ale jsou dostupné přes vrstvu Application framework.

Některé základní knihovny:

- Systémové knihovny jazyka C – z BSD odvozené knihovny jazyka C upravené pro použití ve vestavěných systémech založených na Linuxu.
- Multimediální knihovny – knihovny pro přehrávání hudby a videa a zobrazování obrázků. Např. MPEG4, H.264, MP3, AAC, AMR, JPG a PNG.

- Surface manager – spravuje přístup k displeji zařízení a spojuje 2D a 3D grafické vrstvy více aplikací, čímž umožňuje vytváření grafických efektů (animace obrazovek, posunů nebo průhlednost)
- Grafické knihovny – obsahují knihovny SGL (základní 2D grafický engine) a OpenGL ES (knihovna vykreslování 3D grafiky)
- SQLite – odlehčená relační databázová knihovna
- LibWebCore – knihovna webového prohlížeče.

Android Runtime

Android Runtime se skládá z knihoven poskytujících funkcionalitu jazyka Java, ve které jsou aplikace pro Android vyvíjeny.

Každá aplikace v systému Android běží ve svém vlastním procesu, který je instancí virtuálního stroje. *Dalvik virtual machine* je pro mobilní zařízení optimalizovanou verzí virtuálního stroje Java. Optimalizace spočívá v efektivním využívání systémových prostředků. K tomu účelu poskytuje jádro systému podporu vláken a správu paměti na nízké úrovni.

Aplication Framework

Tato vrstva poskytuje základní bloky pro vytváření aplikací. Všechny aplikace, včetně standardních, používají stejné API. Je tedy možné plně nahradit výchozí aplikace (jako správku kontaktů, editor zpráv SMS/MMS a další) aplikacemi vlastními.

Základní sada služeb poskytuje:

- Systém prvků View – tyto prvky slouží pro sestavení uživatelského rozhraní aplikace. Jsou jimi například textová pole, tlačítka, seznamy, checkboxy a další.
- Activity manager (správce aktivit) – poskytuje správu životního cyklu aplikace a orientaci v zásobníku s aplikacemi. Více v kapitole 2.1.3.
- Content providers (poskytovatelé obsahu) - jsou aplikace dovolující sdílet data mezi aplikacemi (například kontakty).
- Resource manager (správce zdrojů) – poskytují přístup k datům, které nejsou součástí aplikačního kódu, jako jsou jazykové mutace, grafika aplikace nebo definice uživatelského rozhraní.
- Notification manager (správce upozornění) – umožňuje všem aplikacím zobrazit oznámení ve stavovém řádku systému.
- Location manager (správce polohy) – poskytuje přístup do systémové polohové služby. Tato služba poskytuje aplikacím zjišťování geografické polohy.

Aplikace

Operační systém Android v základu obsahuje několik aplikací jako emailový klient, program pro správu SMS, kalendář, mapy, prohlížeč, kontakty a další. Všechny aplikace jsou napsány a využívají programovací jazyk Java.

2.1.2 Komponenty aplikace

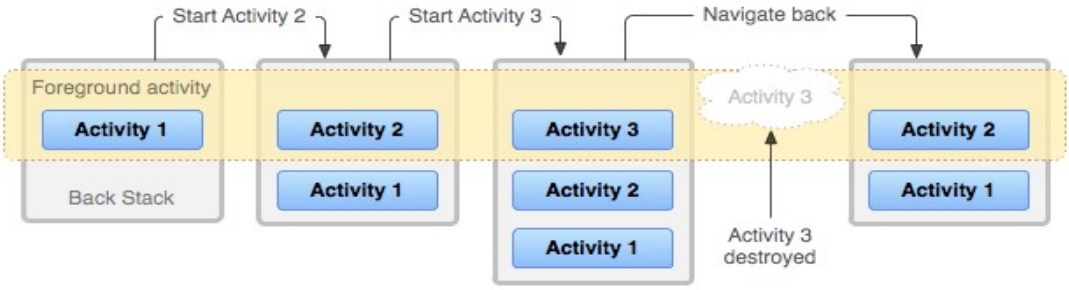
Tato kapitola vychází z [4][5][6][7][8]. Android SDK kompiluje kód aplikace, včetně dat a ostatních zdrojů do jednoho archivu (Android package) s příponou *.apk*. V základu každá aplikace běží ve vlastním procesu. Systém spustí proces, pokud nějaká součást aplikace vyžaduje provádění a ukončí proces, pokud již dále není potřeba, nebo systém potřebuje paměť pro jiné aplikace. Každý proces se spouští ve vlastním virtuálním stroji, proto je aplikační kód oddělen od jiných aplikací.

Existují 4 základní stavební komponenty aplikace: *Activity, Service, Brodcast Receiver, Content Provider*. Každá komponenta má svoji specifickou roli a poskytuje unikátní blok, kterým je možné specifikovat chování aplikace.

Activity

Activity, neboli česky Aktivita, představuje jednu obrazovku uživatelského rozhraní. Definuje se ve zdrojovém kódu jako podtřída třídy *Activity*. GUI je popsáno hierarchií objektů třídy View a aktivitě je předán kořenový prvek pomocí metody *Activity.setContentView()*. Každý objekt ovládá určitou obdélníkovou oblast okna.

Aplikace obvykle sestává z více aktivit, kde každá je nezávislá na ostatních. Jedna z aktivit je označena jako výchozí, což zajistí její zobrazení při spuštění aplikace. Každá aktivita umožňuje spustit další aktivitu, pro poskytnutí jiné akce. Jakmile je spuštěna nějaká aktivita, předchozí aktivita je zastavena, ale systém ji uchovává v zásobníku (back stack). Při startu aktivity je uložen její záznam na vrchol zásobníku a aktivita získává fokus (aktuálně zobrazená a komunikuje s uživatelem). Jakmile je stisknuto tlačítko zpět, je aktivita z vrcholu zásobníku odstraněna a dojde k obnovení předchozí aktivity. Princip zásobníku je zobrazen na obrázku 2.3.



Obrázek 2.3: Ilustrace funkce zásobníku aktivit [6]

Service

Service, neboli Služba, je komponenta, která běží na pozadí bez interakce s uživatelem. Z toho důvodu neposkytuje služba uživatelské rozhraní. Ke spuštěné službě se lze připojit operací *bind* nebo může být služba spuštěna. Po spojení je možné se službou komunikovat pomocí rozhraní které poskytuje.

Příkladem využití služby může být aplikace hudebního přehrávače. Po stisku tlačítka play se spustí služba a přehrávání pokračuje i po opuštění aplikace.

Brodcast Receiver

Přijímač broadcastu je komponenta, která naslouchá a reaguje na broadcastová oznámení. Většina těchto oznámení jsou systémových, jako například nízký stav baterie, doručení SMS zprávy nebo zapnutí obrazovky. Aplikace mohou být také zdrojem broadcastových oznámení.

Podobně jako služba nemá broadcast receiver uživatelské rozhraní, ale může například zobrazit oznámení ve stavové liště nebo spustit jinou aktivitu jako reakci na přijaté oznámení.

Content Provider

Content provider (Poskytovatel obsahu) slouží pro sdílení dat mezi Aktivitami aplikace, ale i mezi aplikacemi. Data mohou být uložena v souborovém systému, v SQLite databázi nebo i na webu. Content provider poskytuje metody (insert, update, delete a query) s podobnou funkcí jako u databází.

Intent

Intent je asynchronní zpráva, sloužící pro aktivování komponent. Intent je objekt typu *Intent*, který nese obsah zprávy. Obsahem zprávy může být akce a cesta k předávaným datům. Stejně tak aktivita může Intentem vracet požadované informace (například vybraný kontakt).

Aplikační zdroje

Aplikace pro android sestává z více součástí než jen ze zdrojových kódů. Tím může dojít k oddělení grafiky aplikace, zvukových souborů, uživatelského rozhraní a jazykových balíčků od zdrojových kódů. Tyto zdroje je pak možné v kódu používat díky třídě *R.java*, do níž se generují identifikátory těchto zdrojů seskupené do skupin podle určení. Toho je využito zejména pro definování uživatelského rozhraní pomocí XML souborů, překlad textových řetězců v aplikaci, nebo volba grafiky podle velikosti displeje zařízení.

Grafické uživatelské rozhraní

Uživatelské rozhraní lze v android aplikaci definovat dvěma způsoby. Prvním způsobem je definovat je přímo v kódu aplikace. Tato varianta však znepřehledňuje funkční kód. Druhým způsobem je možno využít definování GUI pomocí XML souborů. Tento způsob využívá XML elementy, které odpovídají objektům třídy *View*. Atributy elementu jsou nastavitelné parametry objektů. Příklad této struktury je na následujícím výpisu 2.1.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
              android:layout_width="fill_parent"
              android:layout_height="fill_parent"
              android:orientation="vertical" >
    <TextView android:id="@+id/text"
              android:layout_width="wrap_content"
              android:layout_height="wrap_content"
              android:text="Hello, I am a TextView" />
    <Button android:id="@+id/button"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Hello, I am a Button" />
</LinearLayout>
```

Výpis 2.1: Příklad deklarace uživatelského rozhraní pomocí XML souboru

Android Manifest

Soubor *AndroidManifest.xml* je součástí balíčku každé aplikace pro Android. Jeho úkolem je převážně definování komponent aplikace, vymezení oprávnění, která aplikace vyžaduje, určení verze SDK, pro kterou byla aplikace přeložena a minimální verze s níž je kompatibilní. Mohou zde být uvedeny knihovny, které pro svůj běh aplikace vyžaduje.

Systém hledá v manifestu definici komponenty před jejím spuštěním. Součástí definice komponenty jsou *intent-filtry*. Intent-filtr slouží systému pro určení, na které *Intenty* daná komponenta může odpovědět. Komponenta může mít libovolný počet intent-filtrů, pokud nemá žádný, jediný způsob jak ji aktivovat je pomocí intentu, který explicitně jmenuje cílovou komponentu.

2.1.3 Životní cykly komponent

Komponenty aplikace mají svůj životní cyklus. Od vytvoření, přes vykonávání své činnosti až po ukončení. Tato kapitola popisuje stavy, ve kterých se komponenty mohou nacházet a metody sloužící k reakci na změnu těchto stavů.

Životní cyklus Aktivity

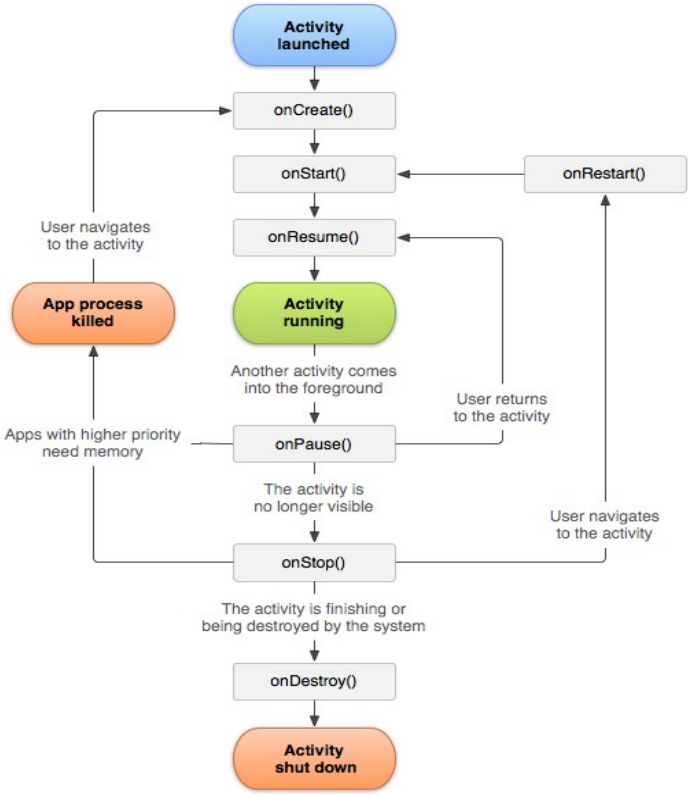
Aktivita se může nacházet ve třech stavech:

- Resumed (obnovena) – Aktivita je v popředí a může komunikovat s uživatelem (má fokus). Tento stav bývá označován jako Běžící.
- Paused (pozastavena) – Aktivita nemá fokus, ale je částečně viditelná (je zčásti překryta jinou aktivitou nebo zcela překryta průhlednou aktivitou). Aktivita je stále uchována v paměti, ale může být systémem ukončena v případě extrémního nedostatku paměti.
- Stopped (zastavena) – Aktivita je úplně překryta jinou neprůhlednou aktivitou a není tedy pro uživatele viditelná. Zastavená aktivita je stále uchovávána v paměti, ale může být systémem ukončena při nedostatku paměti.

Při přepínání mezi těmito stavy je aktivita informována pomocí speciálních metod. Tyto metody provedou základní reakci, ale mohou být nahrazeny nebo rozšířeny o akce, které uživatel vyžaduje. Těmito metodami jsou:

- onCreate() - Volána při prvním vytváření aktivity. Může mít parametr *savedInstanceState,* který obsahuje předchozí uložený stav aktivity. Definuje se zde uživatelské rozhraní.
- onStart() - Volána předtím, než se aktivita stane viditelnou. Může být volána vícekrát za život aktivity.
- onResume() - Volána těsně předtím, než aktivita začne komunikovat s uživatelem.
- onPause() - Volána, když jiná aktivita převezme fokus. Obvykle ukládá provedené změny a zastavuje animace, pro rychlejší běh.
- onStop() - Volána ,pokud jiná aktivita zcela překryje stávající. Může být volána vícekrát za život aktivity.
- onRestart() - Volána při opětovném spuštění aktivity, jestliže byla v zastaveném stavu.
- onDestroy – Volána před ukončením aktivity. Ruší systémové prostředky alokované touto aktivitou.

Uvedené metody definují celý životní cyklus, který je zobrazen na obrázku 2.4.



Obrázek 2.4: Životní cyklus Aktivity [7]

Životní cyklus služby

Životní cyklus služby může mít dvě různé podoby:

- Spouštěná služba – Služba je spouštěna z jiné komponenty metodou *startService()*. Po dokončení činnosti se služba sama zastaví metodou *stopSelf()* nebo je zastavena nějakou komponentou metodou *stopService()*. Jakmile je služba zastavena, systém ji ukončí.
- Spojovaná služba – Služba je vytvořena metodou *bindService()* ve chvíli, kdy se k ní připojí první klient. Klient poté se službou komunikuje pomocí rozhraní *IBinder*. Metodou *unbindService()* klient uzavírá spojení se službou. Služba je ukončena po odpojení všech klientů.

Oba způsoby mohou být kombinovány, pokud jsou dodrženy dané zásady.

Stejně jako aktivita, i služba implementuje metody reagující na změnu stavu životního cyklu.

- onCreate() - Metoda je volána při počátečním vytváření služby. Provádí počáteční inicializaci.
- onStartCommand() - Volána jako reakce na spuštění služby metodou *startService()*.
- onBind() - Volána při připojení klienta metodou *bindService()*.
- onUnbind() - Volána po odpojení všech klientů.

- onDestroy() - Volána při ukončování služby.

Životní cyklus Broadcast Receiveru

Životní cyklus broadcast receiveru má pouze dva stavy: aktivní a neaktivní. Aktivní je pouze při obsluze metody *onReceive()*. Tato metoda je systémem volána ve chvíli, kdy přijde zpráva. Zpráva je metodě předána ve formě Intentu. Po zbytek doby je broadcast receiver neaktivní.

2.2 GPS

Global Positioning System, zkráceně GPS [13], je původně vojenský globální družicový polohový systém provozovaný ministerstvem obrany Spojených států amerických. Tento systém umožňuje určit přesný čas a polohu kdekoliv na Zemi s přesností do deseti metrů. Část tohoto systému je volně k dispozici civilnímu sektoru.

Systém se skládá ze tří sektorů:

- Kosmický – Tento segment projektovaný na 24 družic (dnes 32 družic) obíhajících ve výšce 20200km nad povrchem Země na 6 drahách. Na každé dráze jsou pravidelně rozmístěny 4 družice (nyní nepravidelně 5-6 družic). Každá družice obsahuje atomové hodiny pro určení času vysílané polohy.
- Řídící – Segment je tvořen kontrolními a monitorovacími stanicemi, které udržují družice na správné dráze a nastavují hodiny družice.
- Uživatelský – Uživatelé pomocí GPS přijímače zjišťují svoji polohu podle signálů z družic, jež jsou zrovna nad obzorem. Na základě přijatých dat (časových značek a polohy družice) a předem definovaných parametrů přijímač vypočítá svoji polohu, nadmořskou výšku a přesné datum a čas.

2.2.1 Výpočet polohy a času

Družicové polohové systémy jsou navrženy k jednoduchému výpočtu polohy. Ve speciálních případech je možno uplatnit jiné způsoby měření: [14]

- Kódová – Měření jsou jednoduchá, přesná a nejčastěji používaná. Jejich popis je v dalším textu.
- Fázová – Měření se vykazují vysokou přesností, ale nejednoznačností. Je časově náročné, vyžaduje vhodné podmínky a speciální aparatury nezbytné k okamžité korekci z jiného přijímače. Používá se u vědeckých aplikací.
- Dopplerovská – Měření využívá principu zjišťování frekvence pro pohybující se zdroj signálu (Dopplerův efekt). Na základě údajů z jedné družice zle vypočítat relativní polohu vůči

družici ve dvojrozměrném prostoru a z toho dopočítat polohu na Zemi. Pro určení polohy v trojrozměrném prostoru je třeba měření z více družic.

- Úhloměrná – Měření zaměřuje družici jako zdroj signálu pomocí směrových antén a určuje úhly vzhledem k vodorovné rovině. Provádí se k více družicím současně. Tato metoda se kvůli komplikovanosti měření a malé přesnosti nepoužívá.

Kódová měření

Přijímač na základě signálu z družice dekóduje časové značky odeslání signálu z družice (t) a polohu družice v prostoru (x, y, z). Pozici přijímače musíme při výpočtu popsat třemi souřadnicemi v kartézském systému (X, Y, Z). Pro výpočet je označena jako proměnná i čas přijímače (T). Samotný výpočet probíhá pomocí 4 rovnic (podle vzorce 2.1 [14]), kde *c* je rychlost světla, o 4 neznámých (X, Y, Z, T), za předpokladu, že známe (x, y, z, t) pro 4 družice.

$$(X - x_n)^2 + (Y - y_n)^2 + (Z - z_n)^2 = [(T - t_n)c]^2 \qquad (2.1)$$

V případě příjmu signálu z více než 4 družic je poloha váženým průměrem, čímž může být výsledek mnohem stabilnější a přesnější. Pokud jsou k dispozici pouze 3 družice, je určena poloha pouze ve dvourozměrném prostoru.

2.3 Open Street Map

Open Street Map je projekt, jehož cílem je tvorba volně dostupných geografických dat a následně jejich vizualizace do podoby typografických map. Pro tvorbu geodat se používá záznamů z GPS přijímačů nebo z jiných digitalizovaných map, které jsou licenčně kompatibilní. Projekt je založen na komunitní spolupráci a na koncepci Open source. Data jsou poskytována pod licencí Creative Commons[3]. Projekt umožňuje jednoduchou editaci dat, uchovává kompletní historii změn a výsledky jsou dostupné veřejnosti.

2.4 Platforma Gamooga

Platforma Gamooga poskytuje síťovou infrastrukturu potřebnou pro zasílání realtime zpráv mezi aplikacemi nebo hrami pro více hráčů. Platforma podporuje více koncových bodů a umožňuje vzájemnou komunikaci mezi různými zařízeními a platformami. Těmi jsou HTTP/Javascript, Flash, Android, iOs, Unity. Platforma sestává ze strany Serveru a Klienta.

3 Dostupné z http://creativecommons.org/

2.4.1 Strana serveru

Na Gamooga clustery je možné nahrát *gamlet*, neboli serverový skript. Tyto skripty zajišťují přijímání, zpracování a posílání zpráv svým připojeným klientům. Každý gamlet je tvořen dvěma prvky, jimiž jsou Room a Session. *Room* (česky místnost) reprezentuje skupinu klientů, kteří mezi sebou komunikují. Místnost je tvořena množinou funkcí, jenž představují reakce na přijaté zprávy od klientů. Každý gamlet obsahuje jedinou instanci místnosti. *Session* (neboli relace) je funkčně stejná jako místnost. Rozdílem je, že na gamlet může být více relací, které představují skupinky hráčů (například skupina hráčů pokeru).

Pro gamlet skripty je vyžadován programovací jazyk *Lua* [18]. Jazyk Lua je určen jako rozšiřující nebo skriptovací jazyk. Podporuje pouze základní datové typy, jako jsou boolovské hodnoty, čísla a řetězce. Běžné datové struktury jako pole, množiny, seznamy mohou být reprezentovány použitím jediné datové struktury – tabulky. Sémantika jazyka může být dále rozšiřována a měněna předefinováním zabudovaných funkcí. Navíc Lua podporuje některé pokročilé vlastnosti, jako funkce vyššího řádu a garbage collector, což umožňuje psát i objektově orientované programy.

Gamooga API pro serverovou stranu poskytuje několik základních metod:

- onconnect – tato funkce slouží pro vytvoření reakce na připojení nového klienta. Parametrem této funkce je vygenerovaný identifikátor nového uživatele.
- onmessage – slouží pro zachycení zprávy odeslané uživatelem. Metoda přijímá dva parametry. Prvním je textový identifikátor zprávy a druhým parametrem je funkce reagující na danou zprávu. Tato funkce má dva parametry, jimiž jsou identifikátor uživatele a data zprávy.
- ondisconnect – je opakem onconnect. Reaguje na odpojení uživatele.
- send – je funkce sloužící pro odeslání zprávy vybranému uživateli. Funkce přijímá tři parametry. Prvním je identifikátor uživatele, jemuž bude zpráva zaslána, druhým je textový identifikátor zprávy a posledním parametrem je datový obsah zprávy.
- broadcast – taktéž slouží pro odeslání zprávy, ale zprávu odesílá všem aktuálně připojeným uživatelům. Oproti funkci *send* nemá parametr specifikující uživatele pro příjem zprávy.
- broadcastexcept – slouží pro odeslání zprávy všem uživatelům, kromě vybraných. Tento seznam vyloučených uživatelů je specifikován ve třetím parametru funkce. Prvními dvěma parametry jsou opět identifikátor zprávy a data.

2.4.2 Strana klienta

Pro interakci s gamlety na straně serveru poskytuje Gamooga pro klientskou stranu API, které je dostupné pro výše uvedené platformy. Toto API poskytuje sadu základních funkcí, jako vytvoření spojení, připojení do místnosti, připojení k relaci, odeslaní a příjem zpráv a odpojení. Připojení ke konkrétnímu gamletu je realizováno pomocí Gamlet ID (číslo gamletu) a UUID (univerzální unikátní identifikátor). Tyto jsou přiděleny každému gamletu při vytvoření. Zprávy, pomocí nichž mezi sebou obě strany komunikují, jsou identifikovány pomocí řetězce *msg_type* a mohou nést libovolný datový obsah.

3 Návrh Aplikace

V této kapitole je popsán návrh funkčnosti aplikace a herní systém. Dále je popsán návrh databáze pro uložení dat a na závěr návrh uživatelského rozhraní aplikace.

3.1 Funkčnost aplikace

Navrhovaná aplikace je lokalizační hra, při které je hráčovým úkolem projít na mapě zvýrazněnou množinu bodů. Tato množina bodů je nazvaná trať. Dále aplikace umožňuje pohodlnou správu tratí a zobrazení statistik na každou trať, ale i celkové. Aplikace je funkčně rozdělena na dvě části. První částí je manažer pro správu tratí a druhá část je určená pro hru.

3.1.1 Manažer tratí

Správa tratí je určena pro vytváření, editaci a mazání tratí. Při vytváření a editaci trati je možné změnit pojmenování trati a na zobrazené mapě body tratě. Přidání nového bodu je možné po stisku tlačítka a následného výběru z mapy. Editace nebo smazání bodu je možné při kliknutí na bod. Bod je možné také vygenerovat. Generování je prováděno na aktuálně zobrazeném výřezu mapy. Ke každému bodu jsou vázány informace, jenž doplňují jméno bodu a jeho souřadnice. Těmito informacemi jsou pořadí bodu v trati a volitelná doplňující otázka a odpověď vztahující se k místu.

3.1.2 Herní část

Herní část je rozdělena na dvě části a to hru pro jednoho hráče a hru pro více hráčů. Obě části obsahují společné nastavení hry. V tomto nastavení uživatel volí trať, typ hry a zda budou ve hře povoleny otázky u bodů. Hra obsahuje čtyři následující typy hry:

- Klasický - Úkolem hráče je projít všechny body trati. Vyhrává ten hráč, který má nejnižší čas (při hře pro více hráčů)
- Na čas - Uživatel pro hru nastaví časový limit. Hráč s nejvyšším dosaženým počtem bodů vyhrává.
- Podle pořadí - Cílem hry je dosáhnout nejkratšího času při průchodu bodů podle jejich pořadí.
- Na slepo - U tohoto typu hry nejsou body na mapě zobrazeny. Je zobrazen pouze směr k nejbližšímu bodu.

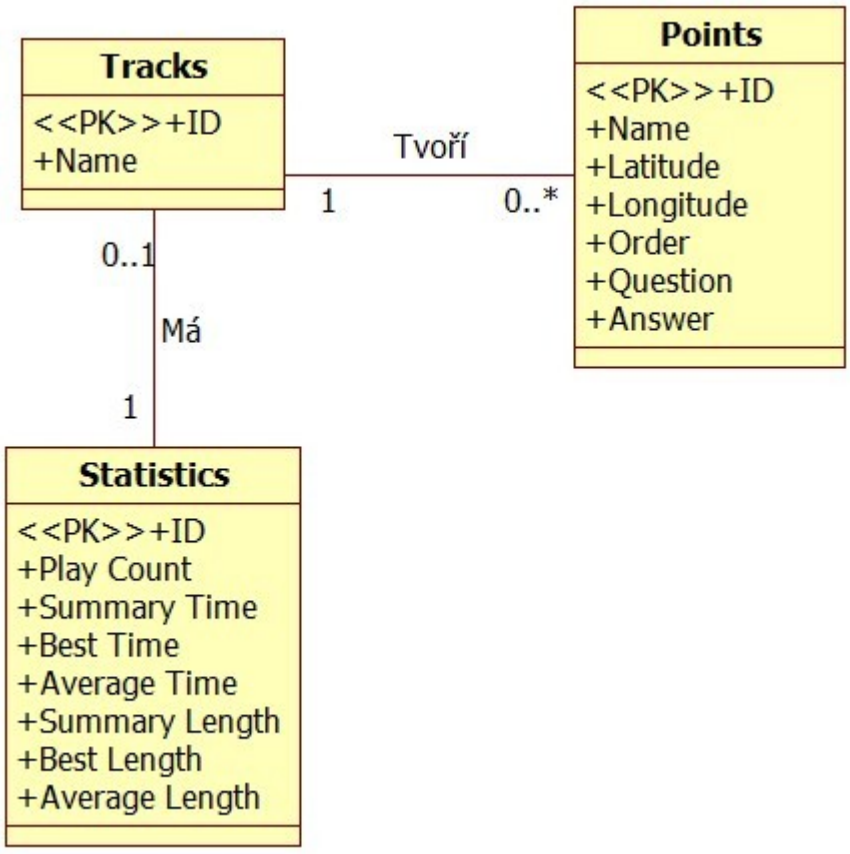
Ve hře pro více hráčů se má uživatel jednu ze dvou rolí. Vytvářející nastavuje parametry hry a vybírá trať. Po připojení je tomuto uživateli zobrazeno identifikační číslo hry. Ostatní uživatelé se

připojují ke hře právě pomocí tohoto identifikátoru. Všichni uživatelé zadávají svoje jméno, aby je bylo možné rozlišit. Po připojení hráče se přenesou nastavení hry a vybraná trať. Vytvářející spouští hru pro všechny připojené hráče.

3.2 Model databáze

Pro uložení dat byl zvolen databázový systém SQLite, jelikož Android poskytuje plný přístup k funkcím tohoto systému. Jako konceptuální model databáze byl zvolen Entity-Relationship diagram (zkráceně ER diagram), jehož hlavní předností je, že lze snadno převést na schéma relační databáze. Výsledný diagram je na obrázku 3.1.

Diagram obsahuje celkem 3 entitní množiny: *Tracks, Points, Statistics*. Entita Tracks obsahuje záznam tratě. Entita Points obsahuje detaily k danému bodu. Mezi entitami Tracks a Points je vztah 1:N, jelikož trať obsahuje více bodů, ale bod náleží právě jedné trati. Entita Statistics sestává z atributů pro uložení statistik k jedné trati, ale navíc obsahuje celkové statistiky. Proto je mezi těmito entitami vztah 1:0..1.



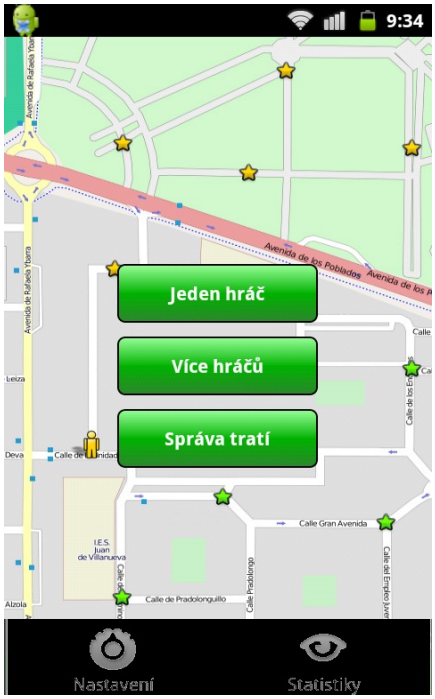
Obrázek 3.1: ER diagram databáze

3.3 Uživatelské rozhraní

Uživatelské rozhraní je nedílnou součástí aplikace, proto je nutné vhodně navrhnout jednotlivé Aktivity a jejich vzájemnou funkcionalitu. Návrh uživatelského rozhraní začal nákresy na papír, na jejich základě bylo rozhraní deklarováno v aplikaci. Při návrhu byl kladen důraz na přehlednost, jednoduchost a pohodlné dotykové ovládání aplikace. Využito je základních prvků widget z nabídky Android View.

3.3.1 Úvodní obrazovka

Po spuštění aplikace je zobrazena obrazovka umožňující uživateli pokračovat do dalších částí aplikace. Tato obrazovka je tedy tvořena třemi tlačítky. První pro přístup do herní části pro jednoho hráče. Druhé pro hru pro více hráčů a třetí tlačítko pro správu tratí. Návrh obrazovky je na obrázku 3.2. Je zde také zobrazeno menu, které je dostupné ze všech obrazovek uživatelského rozhraní. Toto menu obsahuje položky Nastavení a Statistiky. Po zvolení možnosti Nastavení je uživateli zobrazena obrazovka s možnostmi aplikace. Obrazovka se Statistikami je rozdělena do dvou záložek, které reprezentují statistiky celkové a pro vybranou trať. Dále tato obrazovka obsahuje výpis všech dostupných statistik.



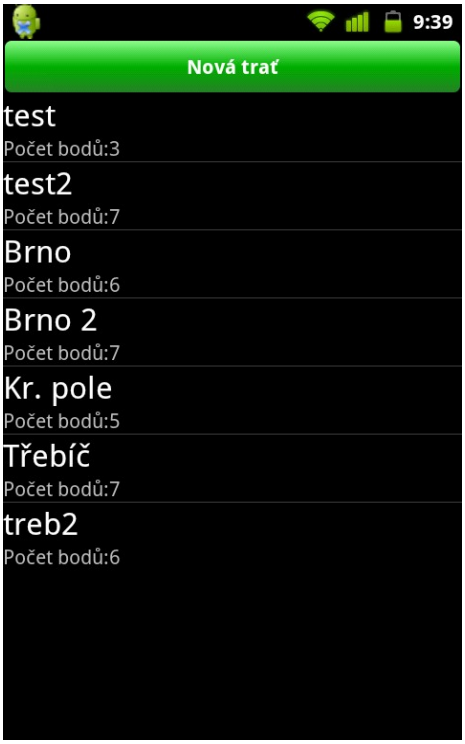
Obrázek 3.2: Úvodní obrazovka aplikace se zobrazeným menu

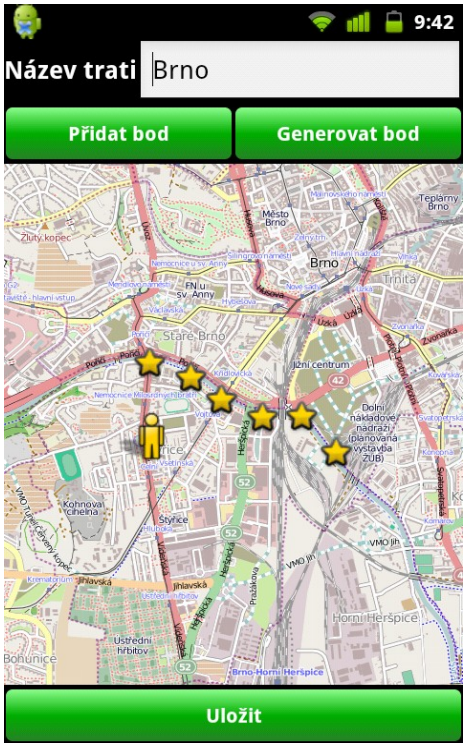
3.3.2 Správa tratí

Správa tratí je tvořena seznamem vytvořených tratí. V seznamu je zobrazováno jméno tratě. Při kliknutí na trať je zobrazena nabídka pro úpravu a smazání tratě. Posledním prvkem obrazovky je tlačítko pro vytvoření nové tratě. Obrazovka je zobrazena na obrázku 3.3.

Při vytváření nebo editaci tratě je zobrazena obrazovka z obrázku 3.4. Jejím obsahem je textové pole pro zadání názvu tratě. Následují dvě tlačítka při přidávání a generování nových bodů. Většinu plochy obrazovky zabírá mapa, na které jsou zobrazeny vytvořené body tratě. Na závěr obrazovka obsahuje tlačítko pro uložení provedených změn.

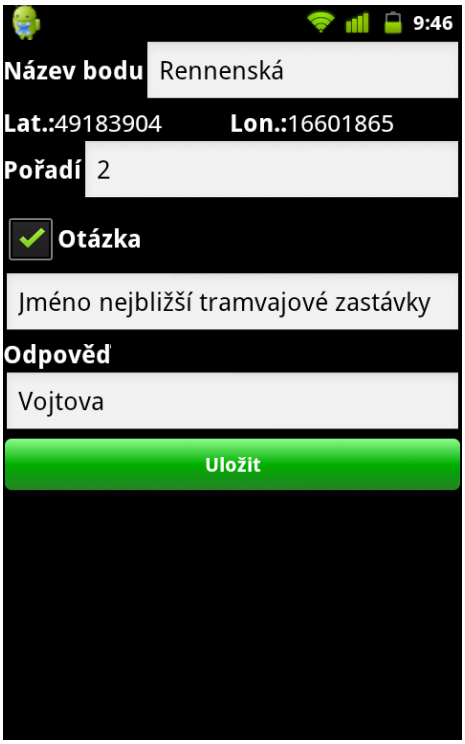
Obrazovka zobrazená při vytvoření nebo editaci bodu obsahuje prvky pro editaci informací o daném bodu. Těmito prvky jsou pole pro zadání jména bodu. Následují informace o zeměpisné výšce a šířce. Dále uživatel může změnit pořadí bodu na trati a povolit u bodu otázku a odpověď. Ty zadává do následujících textových polí. Zbývajícím prvkem je tlačítko pro uložení změn. Návrh obrazovky je na obrázku 3.5.





Obrázek 3.3: Obrazovka seznamu tratí (vlevo)

Obrázek 3.4: Obrazovka editace tratě (vpravo)

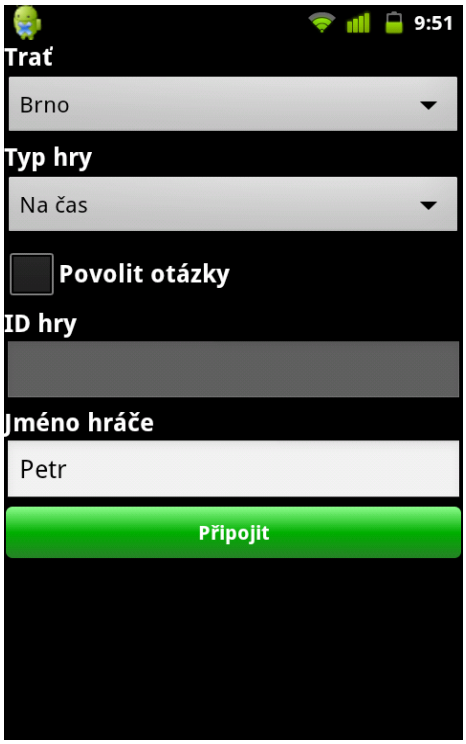


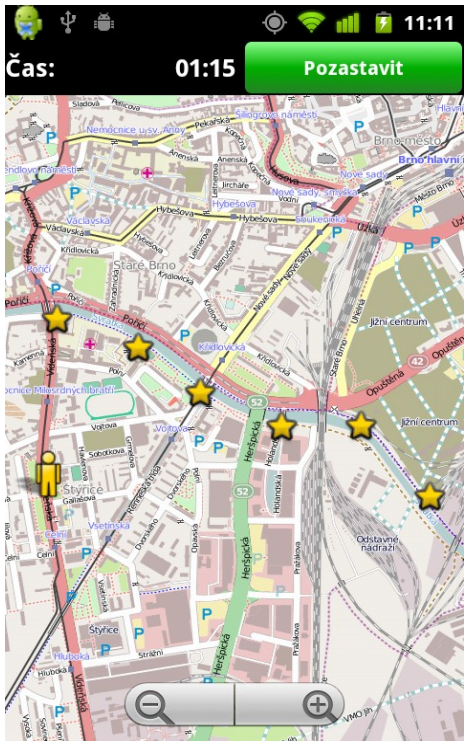
Obrázek 3.5: Obrazovka editace bodu

3.3.3 Herní část

Herní část je rozdělena do dvou obrazovek. První z nich slouží pro nastavení parametrů hry. Obsahuje dva rozevírací seznamy pro výběr tratě a pro výběr typu hry. Následuje zaškrtávací políčko pro povolení otázek ve hře. Tato část je společná pro hru jednoho i více hráčů. Ve druhém případě je obrazovka rozšířena o pole pro zadání identifikace hry a pole pro vyplnění jména hráče. Tato obrazovka je na obrázku 3.6.

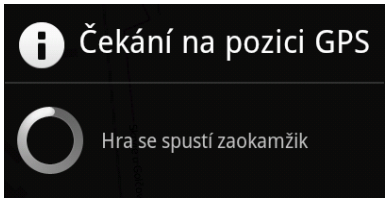
Většinu plochy herní obrazovky zabírá zobrazená mapa. Na mapě je zobrazena poloha uživatele, body které má projít nebo prošel a při hře více hráčů jsou zobrazeni oponenti. Obrazovku doplňuje ukazatel času ve hře a tlačítko pro pozastavení a opětovné spuštění hry. Obrazovka je na obrázku 3.7. Přes herní obrazovku se uživateli pomocí dialogů zobrazují informace o průběhu hry. Při hře pro jednoho hráče je po spuštění hry zobrazen dialog, který uživatele upozorňuje na čekání na GPS pozici. Uživatelům hry pro více hráčů je zobrazen dialog s číslem hry a seznamem připojených hráčů. Oba tyto dialogy jsou zobrazeny na obrázku 3.8. Na dokončení hry je uživatel upozorněn dialogem (viz obrázek 3.9), který zobrazuje finální dosažený čas hry a při hře více hráčů pořadí, v kterém uživatel hru dokončil.



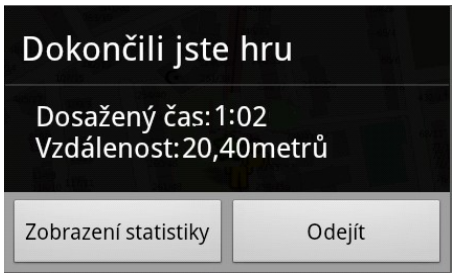


Obrázek 3.6: Obrazovka nastavení hry pro více hráčů (vlevo)

Obrázek 3.7: Herní obrazovka (vpravo)



Obrázek 3.8: Dialogy čekání na zahájení hry



Obrázek 3.9: Dialog dokončení hry

4 Implementace

Tato kapitola se zabývá popisem implementace aplikace. Aplikace je implementována v jazyce Java s použitím Android SDK. Jako minimální verze systému byl stanoven Android 2.1, protože nižší verze mají mají na trhu již malé zastoupení (viz Obrázek 2.1).

4.1 Databáze

Aplikace využívá pro uložení databáze systém SQLite. Práce s databází obstarává třída *TrackDBAdapter* a využívá třídy SQLiteOpenHelper. Tento adaptér se stará o počáteční vytvoření databáze se všemi tabulkami. Dále umožňuje verzování databáze, kdy při změně verze jsou smazány a znovu vytvořeny všechny tabulky. Třída obsahuje funkce pro vytváření, úpravu a mazání záznamů v tabulkách. Dále obsahuje funkce pro nalezení záznamů. Tyto funkce mají jako návratovou hodnotu objekt třídy *Cursor*, který umožňuje přístup k datům databáze.

4.2 Uživatelské rozhraní

Uživatelské rozhraní je vytvořeno deklarativně pomocí XML souborů. Takto definované uživatelské rozhraní je odděleno od funkční logiky aplikace. Pokud je potřeba dosáhnout dynamického chování (například změna popisku komponenty nebo reakce na stisk tlačítka), je komponenta v aplikačním kódu instanciována. Komponenta GUI je v XML souboru deklarována jako element, jehož název odpovídá typu komponenty. Všechny výchozí komponenty systému Android jsou odvozeny od třídy *View*. Atributy elementu slouží pro inicializaci vlastností komponenty.

Příklad deklarace tlačítka ze souboru *game.xml* je na následujícím výpisu 4.1. Tento soubor slouží pro definici obrazovky hry. Atribut *id* je určený k identifikaci tlačítka v kódu aplikace. *Text* slouží pro definování popisku tlačítka. Zde je hodnota atributu dynamicky nastavena z lokalizovaného textového řetězce. Atributy *layout_width* a *layout_height* určují šířku a výšku tlačítka. Jejich hodnota může být zadána počtem pixelů nebo identifikátorem dynamického nastavení. Těmi mohou být *fill_parent* a *wrap_content*. První vyplní v daném rozměru prostor nadřazeného elementu a druhy přizpůsobí rozměr podle obsahu daného elementu. Atribut *layout_weight* udává, jaký poměr obrazovky bude tlačítko zabírat. Dalším použitým atributem je *background*, který může nést hodnotu příslušné barvy pozadí, nebo takzvaný *selektor*. Selektor je XML soubor, který určuje pozadí prvku podle jeho stavu (například u tlačítka změní barvu, pokud je tlačítko zakázáno). Posledními atributy jsou *textStyle* určující styl písma a *textColor* definující barvu písma.

```
<Button
    android:id="@+id/gm_pausebtn"
    android:layout_width="wrap_content"
    android:layout_height="40dip"
    android:layout_weight="0.42"
    android:background="@drawable/button_bg_selector"
    android:textStyle="bold"
    android:textColor="#FFFFFF"
    android:text="@string/gm_pause" />
```

Výpis 4.1: Příklad deklarace tlačítka pomocí XML jazyka

4.3 Zobrazení OpenStreetMap

O zobrazení mapových podkladů a práci nad mapou se stará knihovna Osmdroid[4]. Tato knihovna je poskytována pod licencí GNU General Public License. Jako hlavní prvek pro zobrazení mapy slouží třída *MapView*, která je potomkem třídy View a lze ji použít i pro deklaraci v XML souborech. Tato třída se stará o zobrazení mapových podkladů, proto je u ní možné nastavit různý zdroj těchto podkladů (též render). V aplikace je využit výchozí render Mapnik. Jako další se u této třídy nastavuje ovládání zobrazení mapy. Je možné povolit zobrazení výchozích tlačítek pro zoom mapy nebo využít ovládání pomocí více prstů. Jako další využívaná třída z této knihovny je *MapController*. Tato třída slouží pro ovládání aktuálního pohledu mapy. Lze pomocí ní nastavit aktuální střed mapy a ovládat úroveň přiblížení.

Pro zobrazení jakéhokoliv prvku na mapě je využíváno tříd odvozených od abstraktní třídy *Overlay*. Mezi tyto třídy patří SimpleLocationOverlay, DirectedLocationOverlay, ItemizedOverlay, MinimapOverlay a další. Je možné taktéž vytvořit vlastní odvozenou třídu. Prvky definované v těchto třídách při vykreslování překryjí mapu a je možné použít i více překrytí najednou. Pro všechny definice prvků přidávaných do těchto tříd slouží třída *GeoPoint*. Objekt této třídy představuje bod na mapě. Ten je udávaný pomocí hodnot *latitude*, *longitude* a *altitude*. Dále tato třída poskytuje metody *distanceTo()*, pro výpočet vzdálenosti k jinému bodu, *bearingTo(),* pro určení úhlu směru k jinému bodu a metodu *destinationPoint()*, která naopak z těchto dvou hodnot určí zpětně bod.

Pro zobrazení aktuální hráčovy pozice je využívána třída *SimpleLocationOverlay*, která na mapě zobrazuje pouze jeden prvek. Pro zobrazení směru k nejbližšímu bodu (využíváno při hře Na slepo) je použita třída *DirectedLocationOverlay*. Tato třída stejně jako předchozí zobrazuje pozici uživatele, ale navíc přidává ukazatel směru k nejbližšímu bodu. Třída *ItemizedOverlay* vykresluje seznam prvků *OverlayItem* na mapu. Tyto prvky jsou zadávány pomocí hodnot objektu GeoPoint a je jim možno nastavit libovolný grafický vzhled. Dále tato třída poskytuje metody pro obsloužení událostí při kliknutí na některý z prvků seznamu.

4 Oficiální webové stránky projektu Osmdroid: http://code.google.com/p/osmdroid/

4.4 Řízení hry

O řízení hry se stará třída *Game_Act*, která má na starost zobrazení herní obrazovky. Jejím hlavním úkolem je zobrazení mapy a na ní nutných herních údajů. Mezi tyto údaje patří hráčova poloha, body tratě a při hře více hráčů navíc poloha oponentů. Zobrazení těchto údajů bylo popsáno v kapitole 4.3. Tato kapitola je zaměřena na popis získání potřebných zobrazovaných údajů a dalších komponent potřebných pro hru.

Zobrazování bodů má na starost metoda *refreshPointsOverlay*. Tato metoda získá z databáze body tratě, ze kterých následně vytváří zobrazovaný seznam. Tento seznam je tvořen objekty typu OverlayItem, které jsou definovány pomocí jména a souřadnic bodu. Dále metoda u každého bodu ověřuje, jestli jej už uživatel neprošel. Tento příznak je uložen v seznamu *mReachedPoints* jako identifikátor daného bodu.

Seznam hráčů je tvořen objekty třídy *user_state_list*. Atributy této třídy reprezentují hráčův identifikátor, jméno, pozici a počet dosažených bodů. Z tohoto seznamu je v metodě *refreshPlayerOverlay* vytvářen seznam hráčů pro zobrazení na mapě.

Pro komunikaci se službou poskytující hráčovu pozici je využíván *BroadcastReceiver*. Přepsáním jeho metody *onReceive* je zajištěna reakce na zprávy, které jsou přijímači doručeny. Tyto zprávy jsou ve formě Intentů, které jsou parametrem obslužné metody.

V průběhu hry je uživateli zobrazován uplynulý čas hry nebo v případě hry Na čas zbývající čas. O zobrazení času se stará třídy *myTimerTask*, jenž rozšiřuje třídu *TimerTask*. V metodě *Run* této třídy je čas vypisován do textového pole. Metoda dále ověřuje že nebyl překročen konečný čas pro hru Na čas a popřípadě ukončí hru.

4.5 Služba lokalizace GPS

V průběhu hry je vyžadováno přesné určení hráčovy polohy. O její získávání se stará třída *GPSService*, která rozšiřuje třídy *Service* a *LocationListener*. Tato třída je tedy službou a proto běží na pozadí a vykonává svoji činnost. Předefinované metody služby, mezi které patří *onCreate*, *onBind* a *onUnbind*, slouží pro počáteční nastavení třídy. Je zde prováděno načtení bodů tratě do seznamu a inicializace potřebných komponent.

Hlavní využívaná komponenta je odvozena od třídy *LocationManager*. Tato třída poskytuje přístup k systémové lokalizační službě. a umožňuje získávání periodických aktualizací polohy. Pomocí této komponenty je metodou requestLocationUpdate nastaveno periodické získávání geografické polohy. Je možné nastavit zdroj polohy na poskytovatele sítě nebo GPS satelitů, časový interval a vzdálenost mezi dotazováním nové polohy.

O získání nové polohy je služba informována metodou *onLocationChanged*, která má jako jediný parametr právě získanou polohu. Služba musí o této události informovat aktivitu se hrou. Zde je využit intent, který je odesílán funkcí *sendBroadcast*. Tento intent může nést další doplňující informace ke hře. Touto informací může být dosažení některého bodu tratě. Nová poloha je porovnávána s body tratě a pokud je k některému blíž než nastavený interval, označí se takový bod jako splněný. Další přenášenou informací může být vzdálenost k nejbližšímu bodu, pokud je zvolen typ hry Na slepo. Funkce dále zjišťuje vzdálenost, kterou hráč urazil od poslední změny pozice. K tomuto je využita funkce *distanceTo*, kterou poskytuje třída GeoPoint (viz kapitola 4.3). Poslední informací, kterou může odesílaný intent nést, je zpráva o splnění všech bodů tratě. Na závěr funkce přepíše starou polohu, čímž se z aktuální pozice stane minulá pro další změnu polohy. Celá funkce je zobrazena na výpise 4.2.

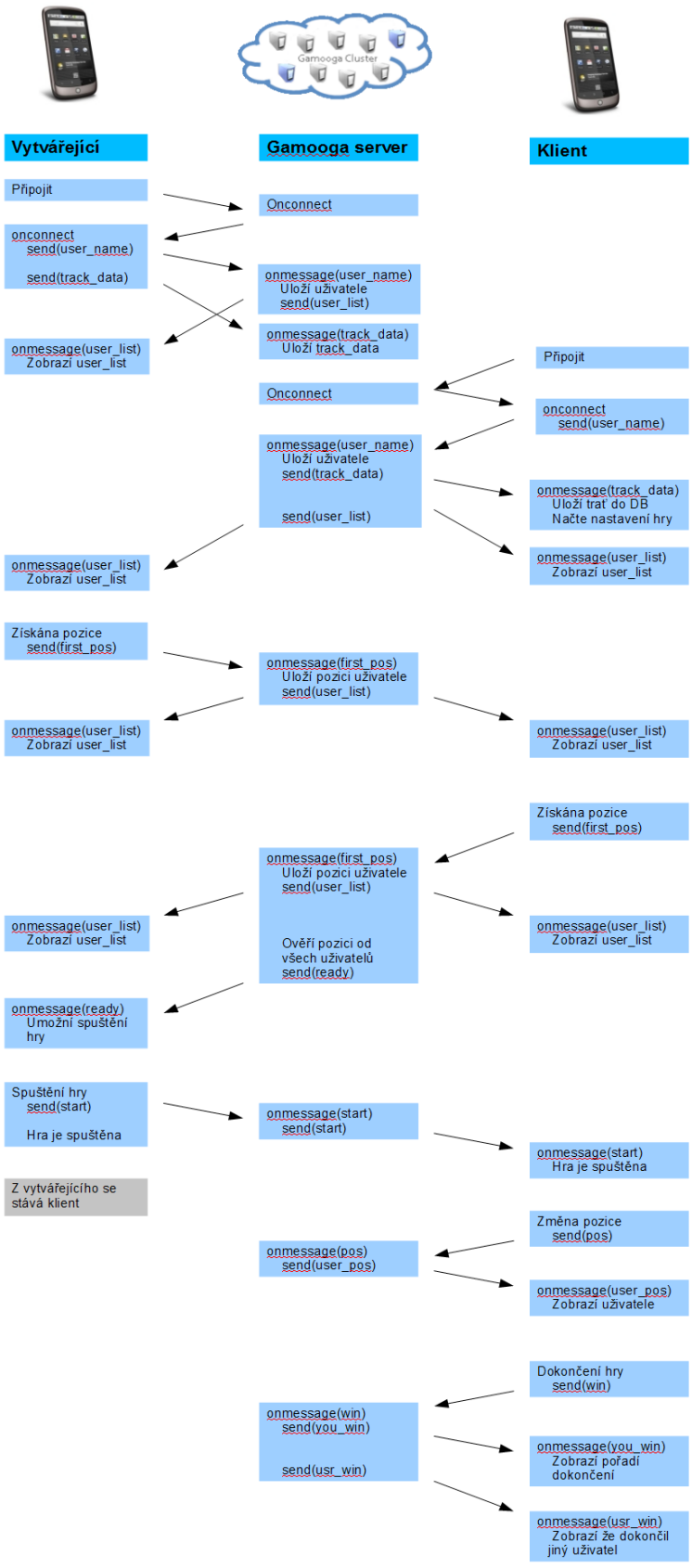
```
public void onLocationChanged(Location location) {
  Intent update = new Intent(getString(R.string.gps_service_intent));
  gPoint nearPoint = null;  //nearest point
  for (Iterator<gPoint> it = pointlist.iterator(); it.hasNext(); ) {
    gPoint poi = it.next();

    if (poi.check)  //overi jestli uz bod nebyl splnen
      continue;

    if (mGameType == Game_Act.GAME_TYPE_ORD)  //overit order
      if( poi.order != mLastOrd)
        continue;

    int distance = cur_loc.distanceTo(poi.getGeoPoint());
    if (distance <= MAXIMUM_LOC_DISTANCE) {  //overi maximalni vzdalenost k bodu
        poi.check = true;
        poi_check_count++;
        update.putExtra("ReachPoint", true);  //posle aktualne dosazeny bod
        update.putExtra("rp_Id", poi.dbId);
        if (mGameType == Game_Act.GAME_TYPE_ORD)
          mLastOrd++;
    }

    if (distance < mindist) {  //najde nejblizsi bod
      mindist = distance;
      nearPoint = poi;
    }
  }//end for

  if (mGameType == Game_Act.GAME_TYPE_BLI) {  //posle smer k nejblizsimu bodu
    update.putExtra("NearestPoint", true);
    update.putExtra("np_bearing", (float) cur_loc.bearingTo(nearPoint.getGeoPoint()));
  }

  track_len += location.distanceTo(mLastLoca);  //ulozit hodnoty pro statistiku
  update.putExtra("tr_length", track_len);

  update.putExtra("cur_loc", location);  //posle soucasnou pozici

  mLastLoca = location;  //ulozit soucasnou pozici

  if (poi_check_count == pointlist.size()) {  //overit pocet splnenych bodu
    update.putExtra("Win_Game", true);
  } else
    update.putExtra("Win_Game", false);

  sendBroadcast(update);
}
```

Výpis 4.2: Kód metody onLocationChanged

4.6 Síťová komunikace

Síťová komunikace u hry pro více hráčů byla navržena s ohledem na použitý herní systém Gamooga. Ten podporuje komunikaci pomocí zpráv. Tyto zprávy jsou rozlišovány pomocí textového identifikátoru a mohou nést datový obsah. Tento obsah je nejčastěji ve formě objektu třídy *JSONObject*. Takovýto objekt může nést libovolná data, která jsou v něm uvozena textovým jménem.

Komunikace ve hře probíhá následujícím způsobem. Uživatel po připojení odesílá zprávu typu „user_name“, která jako data obsahuje jméno uživatele. Server uloží uživatele do seznamu uživatelů a pošle tento seznam všem uživatelům. Dále právě připojenému uživateli pošle trať a nastavení hry (typ zprávy „track_data“). Po přijetí těchto dat uživatel přijatý seznam uživatelů zobrazí a přenesenou trať uloží do své databáze. Vytvářející tyto data nepřijímá. Po získání platné GPS pozice posílá uživatel zprávu „first_pos“, a tuto pozici posílá jako data zprávy. Server po obdržení této zprávy uloží pozici jako atribut do seznamu uživatelů a opět pošle tento seznam všem uživatelům. Dále server ověří, jestli již všichni připojení uživatelé poslali pozici. Pokud má pozici od všech uživatelů, odešle vytvářejícímu zprávu typu „ready“. Vytvářející po obdržení této zprávy může spustit hru, což znamená odeslání zprávy „start“, kterou server odešle všem ostatním uživatelům. Od tohoto okamžiku se vytvářející stává běžným uživatelem. Dalším typem zprávy je „pos“, kterou uživatel zasílá při změně pozice ve hře. Jako datový obsah této zprávy je opět aktuální GPS pozice uživatele. Server tuto zprávu odešle všem uživatelům a uloží pozici do seznamu uživatelů. Posledním typem zprávy je „win“, která je odeslána pokud uživatel dokončí hru. Server po přijetí této zprávy posílá dvě různé zprávy. Uživateli, který zprávu pošle zasílá ve zprávě „you_win“ pořadí v jakém hru dokončil a ostatní uživatele informuje zprávou „usr_win“ o dokončení daného uživatele. Pokud se uživatel kdykoliv v průběhu hry odhlásí, server reaguje speciální metodou ondisconnect. V této metodě odebere uživatele ze seznamu a takto upravený seznam odesílá všem uživatelům. Celou komunikaci lze vidět na diagramu z Obrázku 4.1.



Obrázek 4.1: Diagram síťové komunikace

5 Testování

Důležitým aspektem při vytváření aplikace je její důkladné otestování. Pro testování byl použit především emulátor, ale i několik reálných zařízení. Testování probíhalo dvěma způsoby. Pro testování vzhledu uživatelského rozhraní byl z větší míry použit emulátor, díky možnosti měnit parametry emulovaného zařízení. Naopak pro testování funkčnosti aplikace byla využívána reálná zařízení, protože aplikace ke svému běhu potřebuje signál GPS, který není možné na emulátoru zachytit.

Testování byla prováděna na různých verzí systému Android. Na emulátoru byly vyzkoušeny systémy od verze 2.1 po verzi 2.3. navíc byla použita různá rozlišení displeje a to QVGA (240x320 pixelů), HVGA (320x480 pixelů), WVGA (460x800 pixelů) a WXGA (800x1280 pixelů). Jako reálná zařízení byla použita HTC Desire s verzí systému 2.3.7 a rozlišením WVGQ, dále HTC Wildfire S se systémem 2.3.5 s rozlišením QVGA a Samsung Galaxy Ace s rozlišením HVGA a verzí systému 2.2.

Největší čas testovaní zabralo testování správné funkčnosti získávání GPS pozice. Pomocí tohoto testování bylo například zjištěno špatné zpracování aktuální pozice, kdy docházelo ke špatnému záznamu a pozice se tak neaktualizovala. Další chybou zjištěnou při testování bylo nesprávné porovnávání pořadí bodů, kdy docházelo ke špatnému určení následujícího bodu.

Obrázek 5.1: Testování aplikace na emulátoru

6 Závěr

Cílem práce bylo vytvořit lokalizační hru pro mobilní zařízení, která využívá GPS modul a data zobrazuje na mapových podkladech Open Street Maps. Tato kapitola se zabývá zhodnocením dosažených cílů práce a možnou budoucností vypracovaného projektu.

6.1 Přínos práce

Největším přínosem této práce pro mě bylo seznámení se s platformou Android a jejím SDK. Z hlediska uživatele pro mě bylo zajímavé pochopení funkcí a chování systému Android a jeho aplikací. Z programátorského hlediska pochopení návrhu a vytváření aplikace pro velmi rozšířenou mobilní platformu. Dále jsem si osvojil návrh a vytváření uživatelského rozhraní pomocí deklarací jazyka XML, pomocí kterého je oddělena aplikační logika od uživatelského rozhraní a od dalších zdrojů jako grafika aplikace a lokalizované textové řetězce. Dalším přínosem pro mě byla práce s databází SQLite a návrh síťové komunikace hry.

6.2 Budoucnost projektu

Jako nejbližší budoucnost implementované aplikace je její uvedení v obchodu s aplikacemi Google Play. S tím souvisí vyzkoušení aplikace reálnými uživateli, kteří mohou poskytnout užitečné rady a nápady ohledně vylepšení aplikace. V případě úspěchu aplikace je možné rozšířit ji na další mobilní platformy jako iOS společnost Apple nebo Windows Mobile od Microsoft. Další možností rozšíření aplikace je vytvoření webového rozhraní, které bude sloužit pro vytváření tratí pro hru a jejich sdílení mezi uživateli. Mezi rozšíření funkcí hry lze uvést přidání další herních typů nebo například generování celé trasy podle zadaného středu, rádiu a počtu bodů.

Literatura

[1] Burnette, E.: *Hello, Android*. Pragmatic Bookshelf, Červenec 2010, 300 str., iSBN 1-934356-56-5.

[2] GOOGLE: *What is Android?* [online]. Únor 2012 [cit. 2012-02-21]. URL: http://developer.android.com/guide/basics/what-is-android.html

[3] GOOGLE: *Platform Versions* [online]. Únor 2012 [cit. 2012-02-21]. URL: http://developer.android.com/resources/dashboard/platform-versions.html

[4] WIKIPEDIA: *Android_(operační_systém)* [online] 21. únor 2012 [cit. 2012-02-21]. URL: http://cs.wikipedia.org/wiki/Android_(operační_systém)

[5] GOOGLE: *Application Fundamentals* [online]. Únor 2012 [cit. 2012-02-21]. URL: http://developer.android.com/guide/topics/fundamentals.html

[6] GOOGLE: *Tasks and back Stack* [online]. Únor 2012 [cit. 2012-02-21]. URL: http://developer.android.com/guide/topics/fundamentals/tasks-and-back-stack.html

[7] GOOGLE: *Activities* [online]. Únor 2012 [cit. 2012-02-22]. URL: http://developer.android.com/guide/topics/fundamentals/activities.html

[8] GOOGLE: *Services* [online]. Únor 2012 [cit. 2012-02-22]. URL: http://developer.android.com/guide/topics/fundamentals/services.html

[9] Meier, R.: *Profesional Android 2 Application development*. Wiley Publishing, Inc., Březen 2010, 576 str., iSBN 987-0-470-56552-0.

[10] Haseman, C.: *Creating Android Application*. Peachpit Press, Listopad 2011, 272 str., iSBN 978-0321784094.

[11] Ostrander, J.: *Android UI Fundamentals*. Peachpit Press, Únor 2012, 320 str., iSBN 978-0-321-81458-6.

[12] KVAPIL, J.: Kosmický segment GPS a jeho budoucnost [online]. Leden 2005 [cit. 2012-02-24]. URL: http://www.aldebaran.cz/bulletin/2005_02_gps.php

[13] WIKIPEDIA: *Global Positioning System* [online]. Únor 2012 [cit. 2012-02-24]. URL: http://cs.wikipedia.org/wiki/GPS

[14] WIKIPEDIA: *Globální družicový polohový systém* [online]. Leden 2012 [cit. 2012-02-24]. URL: http://cs.wikipedia.org/wiki/Globální_družicový_polohový_systém

[15] WIKIPEDIA: *OpenStreetMap* [online]. Únor 2012 [cit. 2012-02-24]. URL: http://cs.wikipedia.org/wiki/OpenStreetMap

[16] Rethans, D.: What is OpenStreetMap? [online]. Květen 2011 [cit. 2012-02-24]
URL: http://derickrethans.nl/what-is-openstreetmap.html

[17] GAMOOGA: *Gamooga Documentation* [online]. Březen 2012 [cit. 2012-04-14]
URL: http://www.gamooga.com/dev/docs/index.html

[18] WIKIPEDIA: *Lua* [online]. Leden 2012 [cit. 2012-04-14]
URL: http://cs.wikipedia.org/wiki/Lua

Seznam příloh

Příloha A

Obsah CD

CD:/Aplikace

/apk – instalační balíček pro systém Android

/source – zdrojové kódy projektu pro prostředí Eclipse

/Technická zpráva

/pdf – tato zpráva ve formátu pdf

/zdroj – zdrojový text dokumentu ve formátu OpenOffice dokument odt

/Manuál

/pdf – uživatelská příručka instalace a ovládání aplikace

/Plakát

/pdf – plakát představující aplikaci