České vysoké učení technické v Praze
Fakulta elektrotechnická

katedra počítačů

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Student: **Tomáš Nosek**

Studijní program: Softwarové technologie a management
Obor: Softwarové inženýrství

Název tématu: **SocioCars - aplikace pro Android**

Pokyny pro vypracování:

Seznamte se s projektem Sociocars, jehož cílem je vytvořit síť pro řidiče aut. Analyzujte požadavky na mobilní aplikaci pro tento projekt, dále podle analýzy proveďte návrh, implementaci a testy mobilní aplikace na platformě Android.

Předpokládané výstupy práce:
1. Analýza požadavků na obecnou mobilní aplikaci pro Sociocars.
2. Návrh a implementace mobilní aplikace na platformě Android schopné komunikovat se serverem Sociocars.
3. Uživatelské testy aplikace.

Seznam odborné literatury:

1. Skalický, T. Salich, L.: Sociocars, ČVUT FEL projekt A7B36SI2, 2013
2. Skalický, T.: Sociocars, ČVUT FEL semestrální projekt, 2013
3. Android Developer Documentation, http://developer.android.com
4. Google Maps API https://developers.google.com/maps/
5. Kubů, R.: Zpracování dat o provozu automobilů pro projekt Metrocar. ČVUT FEL, 2013

Vedoucí: Ing. Ondřej Macek

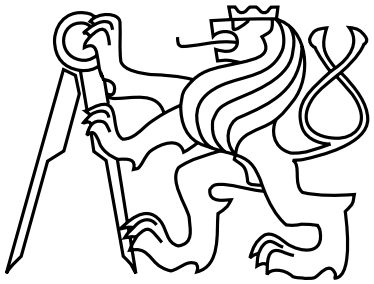
Platnost zadání: do konce letního semestru 2014/2015

doc. Ing. Filip Železný, Ph.D.
vedoucí katedry

prof. Ing. Pavel Ripka, CSc.
děkan

V Praze dne 28. 2. 2014

České vysoké učení technické v Praze
Fakulta elektrotechnická
Katedra počítačů



Bakalářská práce

Sociocars pro Android

Tomáš Nosek

Vedoucí práce: Ing .Ondřej Macek

Studijní program: Softwarové technologie a management, Bakalářský

Obor: Softwarové inženýrství

19. května 2014

Poděkování

Rád bych poděkoval vedoucímu své práce, Ing. Ondřeji Mackovi, za jeho ochotu a cenné rady.
Dále bych chtěl poděkovat své rodině, která mě po celou dobu studia podporovala.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze dne 15. 5. 2014 .....

Abstract

The objective of this thesis is the analysis, design, implementation and testing of a mobile application which will be part of a social network for motorists, the Sociocars.
The goal is to create a mobile application for the Android operating system, which will allow recording routes and will be capable of communicating with the engine control unit and communicate with the Sociocars server.

Abstrakt

Předmětem této práce je analýza, návrh, implementace a testování mobilní aplikace, která bude součástí projektu sociální sítě pro motoristy Sociocars.
Cílem je vytvoření mobilní aplikace pro operační systém Android, která bude umožňovat zaznamenávání tras, bude schopná spojení se s řídící jednotkou automobilu a komunikace se serverem Sociocars.

Obsah

1 Úvod **1**
- 1.1 O projektu Sociocars . . . 1
- 1.2 Cíl práce . . . 1
- 1.3 Motivace . . . 1
- 1.4 Co je obsahem této práce . . . 2

2 Analýza projektu **3**
- 2.1 Analýza požadavků . . . 3
  - 2.1.1 Funkční požadavky . . . 3
  - 2.1.2 Nefunkční požadavky . . . 4
- 2.2 Klíčové problémy . . . 4
  - 2.2.1 Získávání polohy . . . 5
  - 2.2.2 Komunikace s řídící jednotkou . . . 5
  - 2.2.3 Ukládání zaznamenaných tras . . . 6
  - 2.2.4 Spojení se serverem Sociocars . . . 6

3 Podobné systémy **7**
- 3.1 Zubie . . . 7
- 3.2 Mavia . . . 8
- 3.3 Torque . . . 8
- 3.4 Shrnutí . . . 9

4 Analýza tvorby aplikací pro platformu Android **11**
- 4.1 Základní popis OS Android . . . 11
  - 4.1.1 Historie operačního systému Android . . . 11
- 4.2 Architektura operačního systému Android . . . 12
- 4.3 Zásady tvorby aplikací pro Android . . . 13
  - 4.3.1 Doporučené rozložení obrazovky . . . 13
    - 4.3.1.1 Action bar . . . 13
    - 4.3.1.2 Vlastní obsah . . . 13
    - 4.3.1.3 Spodní lišta . . . 14
- 4.4 Postup vývoje aplikace Sociocars . . . 14

5 Návrh uživatelského rozhraní **15**
5.1 Návrh posloupnosti akcí v aplikaci . . . 15
5.2 Návrh jednotlivých obrazovek aplikace . . . 16
5.2.1 Úvodní obrazovka . . . 16
5.2.1.1 Případy užití pro úvodní obrazovku . . . 17
5.2.2 Záznam trasy . . . 18
5.2.2.1 Případy užití pro obrazovku se záznamem trasy . . . 18
5.2.3 Seznam uložených tras . . . 19
5.2.3.1 Případy užití pro obrazovku se seznamem tras . . . 19
5.2.4 Zobrazení uložené trasy . . . 20
5.2.4.1 Případy užití pro obrazovku se zobrazenou trasou . . . 20

6 Realizace **21**
6.1 Struktura . . . 21
6.2 Problémy a jejich řešení . . . 22
6.2.1 Získávání polohy . . . 22
6.2.2 Získávání dat z OBD2 jednotky . . . 23
6.2.3 Ukládání zaznamenaných tras . . . 23
6.2.3.1 Ukládání do databáze v průběhu záznamu . . . 23
6.2.3.2 Ukládání do KML souboru po skončení záznamu . . . 23
6.2.4 Zobrazení trasy . . . 24
6.2.5 Komunikace se serverem . . . 25
6.2.6 Ověření uživatele na serveru . . . 26
6.2.6.1 Získání seznamu vozidel uživatele . . . 26
6.2.6.2 Nahrání ujeté trasy na server . . . 27
6.2.6.3 Newsfeed . . . 28

7 Testování **29**
7.1 Testování pomocí nástroje Robotium . . . 29
7.2 Testování pomocí nástroje AppThwack . . . 30
7.2.1 Výsledek testování pomocí AppThwack . . . 30
7.3 Testování v reálném prostředí . . . 31

8 Závěr **33**
8.1 Možné pokračování práce . . . 33
8.2 Závěrečné zhodnocení . . . 33

A Seznam použitých zkratek **39**

B Obsah přiloženého CD **41**

Seznam obrázků

3.1 Snímky obrazovky aplikace Zubie 8
3.2 Snímky obrazovky aplikace Torque 9

4.1 Architektura operačního systému Android [1] 12
4.2 Action bar 13
4.3 Vlastní obsah 14
4.4 Spodní lišta 14

5.1 Návrh posloupnosti akci v aplikaci 15
5.2 Návrh úvodní obrazovky 16
5.3 Obrazovka se záznamem trasy 18
5.4 Obrazovka se seznamem tras 19
5.5 Návrh obrazovky se zobrazením trasy 20

6.1 Diagram balíčků 21
6.2 Diagram nasazení 25

7.1 Snímek obrazovky se zaznamenanou trasou 31
7.2 Zobrazení trasy na serveru 32

Kapitola 1

Úvod

1.1 O projektu Sociocars

Tato bakalářská práce je součástí projektu Sociocars, jehož cílem je vytvoření nové sociální sítě určené primárně pro motoristy, ve které si uživatelé mohou mezi sebou sdílet vozidla, příspěvky, sledovat stav svých vozidel a hlavně přidávat a sdílet ujeté trasy.

V současné době existuje sociální síť Sociocars pouze v podobě webové aplikace vyvíjené Tomášem Skalickým [2].

1.2 Cíl práce

Cílem této práce je vytvoření mobilní aplikace pro platformu Android, která bude součástí projektu Sociocars.

Vytvářená mobilní aplikace by měla sloužit jako podpůrná aplikace pro webové rozhraní sociální sítě Sociocars. Měla by být schopná zaznamenávat trasu a získávat co nejvíce informací o automobilu, všechna nasbíraná data sdílet se serverem Sociocars a umožnit uživateli přístup k datům poskytovaným serverem Sociocars.
V rámci projektu budou využity znalosti o zjišťování stavu vozidla získané v projektu Metrocars [3].

1.3 Motivace

V současné době, kdy se stále více uživatelů připojuje k internetu pomocí mobilních zařízení, je důležité, aby měl systém jako je sociální síť Sociocars podpůrnou mobilní aplikaci, pomocí které získají uživatelé možnost být stále v centru dění a v kontaktu s ostatními uživateli sociální sítě.

U projektu Sociocars je navíc důležitý sběr dat, například polohy nebo informací o automobilu, z toho důvodu je vytvoření mobilní aplikace pro jeho plnou využitelnost v podstatě nezbytné.

Projekt se zároveň dotýká pro mě velice zajímavého problému, a to spojením mobilního zařízení s řídící jednotkou automobilu. To je prozatím relativně nerozšířená možnost využití mobilních zařízení, která si bezesporu zaslouží pozornost.

1.4 Co je obsahem této práce

Tato práce se zabývá analýzou, návrhem, realizací, testováním a nasazením mobilní aplikace Sociocars pro operační systém Android.

Kapitola 2

Analýza projektu

V této části se budu zabývat analýzou požadavků, možných problémů, možnostmi jejich řešení a analýzou podobných systémů.

2.1 Analýza požadavků

Pro získání představy o tom, jakou funkčnost by měla výsledná aplikace poskytovat, bylo nutné vytvořit analýzu funkčních a nefunkčních požadavků.

2.1.1 Funkční požadavky

- FP00: Aplikace bude umožňovat zaznamenávání trasy.
- FP01: Aplikace bude umožňovat přerušení, znovuspuštění nebo ukončení a uložení zaznamenávané trasy.
- FP02: Aplikace bude umožňovat zobrazení zaznamenaných tras na mapě.
- FP03: Aplikace bude umožňovat zobrazení statistik jednotlivých tras.
- FP04: Aplikace bude umožňovat zobrazení průměrných rychlostí tras.
- FP05: Aplikace bude umožňovat zobrazení ujetých vzdáleností u zaznamenaných tras.
- FP06: Aplikace bude umožňovat zobrazení celkových překonaných převýšení u zaznamenaných tras.
- FP07: Aplikace bude umožňovat uložení souboru se zaznamenanou trasou do lokálního úložiště mobilního zařízeni.
- FP08: Aplikace bude při záznamu umožňovat zobrazení aktuálního stavu vozidla z dat získaných z OBD2 jednotky.
- FP09: Aplikace bude umožňovat přístup k rychlosti vozidla z OBD2 jednotky.

- FP10: Aplikace bude umožňovat přístup k teplotě chladící kapaliny motoru získané z OBD2 jednotky.
- FP11: Aplikace bude umožňovat přístup k otáčkám motoru získaných z OBD2 jednotky.
- FP12: Aplikace bude umožňovat přístup k aktuální spotřebě paliva získané z OBD2 jednotky.
- FP13: Aplikace bude umožňovat sdílení uložených tras se serverem Sociocars.
- FP14: Aplikace bude umožňovat sdílení aktuální polohy se serverem Sociocars.
- FP15: Aplikace bude umožňovat sdílení aktuálního stavu vozidla se serverem Sociocars.
- FP16: Aplikace bude umožňovat zaznamenávání informací získaných z OBD2 jednotky.
- FP17: Aplikace bude umožňovat přihlášení k serveru Sociocars.
- FP18: Aplikace bude umožňovat přístup k uloženým vozidlům ze serveru Sociocars.
- FP19: Aplikace bude umožňovat přidání nového vozidla na server Sociocars.
- FP20: Aplikace bude umožňovat zobrazení novinek ze serveru Sociocars.

2.1.2 Nefunkční požadavky

- Aplikace bude fungovat na všech zařízeních používající Android verze 4.0 nebo vyšší.
- Aplikace bude pro plnou funkčnost vyžadovat připojení k OBD2 [4] jednotce.
- Aplikace bude pro plnou funkčnost dále vyžadovat:
  - připojení k internetu,
  - připojení bluetooth,
  - přístup k poloze,
  - přístup k úložišti dat.

2.2 Klíčové problémy

V této sekci jsou shrnuty klíčové a potenciálně rizikové problémy, se kterými se bude nutné při vývoji vypořádat a jejich možné řešení.

2.2.1 Získávání polohy

Pro záznam trasy je potřebné mít co nejpřesněji určenou polohu. Android poskytuje několik možností jak získat aktuální polohu:

- Získávání polohy pomocí mobilní sítě.
- Získávání polohy pomocí dostupných sítí Wi-Fi.
- Získávání polohy pomocí GPS.

Poněvadž je k záznamu trasy potřeba co nejpřesnější poloha, v úvahu připadá pouze zjišťování polohy pomocí GPS. Používání GPS v mobilním zařízení sice zvyšuje spotřebu baterie, ale dá se předpokládat, že aplikace bude používána primárně při jízdě automobilem, ve kterém se většinou nachází zdroj energie. K získávání polohy z GPS bude použito *Android framework location API [5].*

2.2.2 Komunikace s řídící jednotkou

Pro získávání co nejvíce informací o vozidle bude aplikace schopná komunikovat s jeho řídící jednotkou. Jedinou možností je komunikace s řídící jednotkou pomocí OBD2 protokolu přes rozhraní bluetooth.

Zkratka OBD znamená *On-Board Diagnostics* a jedná se o protokol určený k diagnostice emisních systémů osobních automobilů. Přítomnost OBD2 ve vyráběných automobilech je povinná v USA od roku 1996, v Evropě od roku 2001 pro všechny automobily s benzínovým motorem a od roku 2004 pro všechny automobily s dieselovým motorem [4].

Mezi hlavní výhody použití OBD2 tedy patří skoro jistá přítomnost ve vozidle a také nízká cena OBD2 adaptéru, která se u nejlevnějších bluetooth adaptérů pohybuje kolem 300,- Kč.

Pro použití OBD2 jednotky s vyvíjenou aplikací se nabízí několik možností řešení:

- Implementace vlastního řešení pro komunikaci s OBD2 jednotkou.
- Najití a použití vhodné open-source knihovny určené pro komunikaci s OBD2 jednotkou.
- Použití existujícího řešení z aplikace Metrocars [6] vyvíjené Romanem Kubů v rámci FEL ČVUT.

Po nastudování problematiky bylo zhodnoceno, že nejlepší variantou bude použití a úprava existujícího řešení z projektu Metrocars.

2.2.3 Ukládání zaznamenaných tras

Aplikace má umožňovat ukládání zaznamenaných tras do datového úložiště mobilního zařízení a dále s nimi pracovat. Je tedy důležité zvolit vhodný formát dat, který bude zároveň možné dostatečně customizovat pro uložení informací z řídící jednotky.

V úvahu připadají dva formáty:

- GPX (Gps Exchange Format) [7]: Jednoduchý XML formát pro výměnu GPS dat (bodů, tras, atd.) .
- KML (Keyhole Markup Language) [8]: XML formát, který je standardem OGC (Open Geospatial Consortium [9]) a je vyvíjen společností Google. Slouží k vyjadřovaní geografických dat pomocí dvou- i tří-dimenzionálních prohlížečů. K formátu existuje rozsáhlá dokumentace. Do KML se při dodržení jeho formátu dají přidat libovolná data.

Pro ukládání dat byl vybrán formát KML. Zejména pro jeho rozšířenost, rozsáhlou dokumentaci a jednoduchou rozšiřitelnost vlastními daty.

2.2.4 Spojení se serverem Sociocars

Server Sociocars by měl podporovat architekturu rozhraní REST (Representional State Transfer). Rozhraní umožňuje snadný přístup ke zdrojům pomocí jasně definovaného URI [10].

REST rozhraní může obsahovat čtyři základní metody:

- GET: základní metoda pro získání zdroje.
- POST: Metoda pro vytvoření dat na serveru.
- DELETE: Metoda pro smazání zdroje na serveru.
- PUT: Metoda pro úpravu zdroje. Na rozdíl od metody POST se zde volá již existující zdroj

Pro přenos dat budou kvůli požadavkům serveru použity JSON [11] zprávy.

Kapitola 3

Podobné systémy

Tato kapitola se zabývá analýzou podobných systémů. Budou zde shrnuty jejich základní funkce a ty, které mají spojitost se zadáním projektu Sociocars.

Pomocí výsledné rešerše bude možné zanalyzovat, jak vypadá a funguje potenciální konkurence mobilní aplikace Sociocars. Bude se možné inspirovat, nebo naopak poučit z chyb.

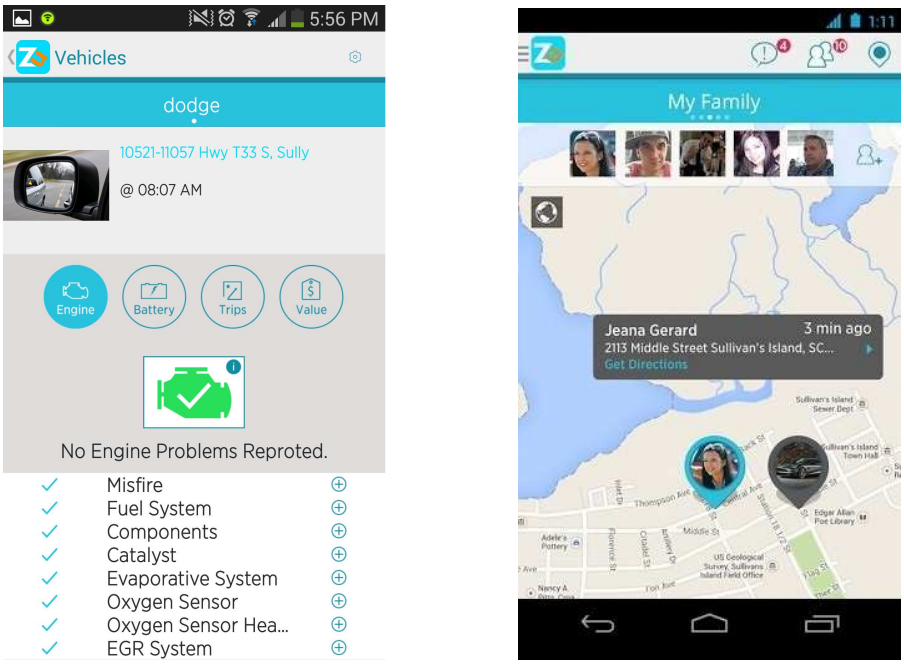
3.1 Zubie

Dle popisu na webových stránkách projektu [12] se jedná o aplikaci poskytující spojení webové aplikace a mobilního zařízení připojeného k řídící jednotce automobilu. Systém umožňuje sledování polohy, získávání informací o motoru, zobrazení chybových hlášení z řídící jednotky automobilu či propojení s rodinou nebo přáteli a vzájemné sdílení dat. Mezi hlavní funkce patří:

- Zjišťování polohy vozidla v reálném čase.
- Upozornění na problémy zjištěné z řídící jednotky.
- Měření kvality řízení.
- Vytváření skupin vzájemně sdílejících polohu pro car-sharing [1] nebo společné cestování.
- Zaznamenávání trasy společně s daty získanými z řídící jednotky a následné sdílení na sociální sítě.

[1] Car-sharing: komerční sdílení automobilů více řidiči

Na obrázcích 3.1a a 3.1b se nachází ukázka z aplikace Zubie.



(a) Zubie 1 (b) Zubie 2

Obrázek 3.1: Snímky obrazovky aplikace Zubie

3.2 Mavia

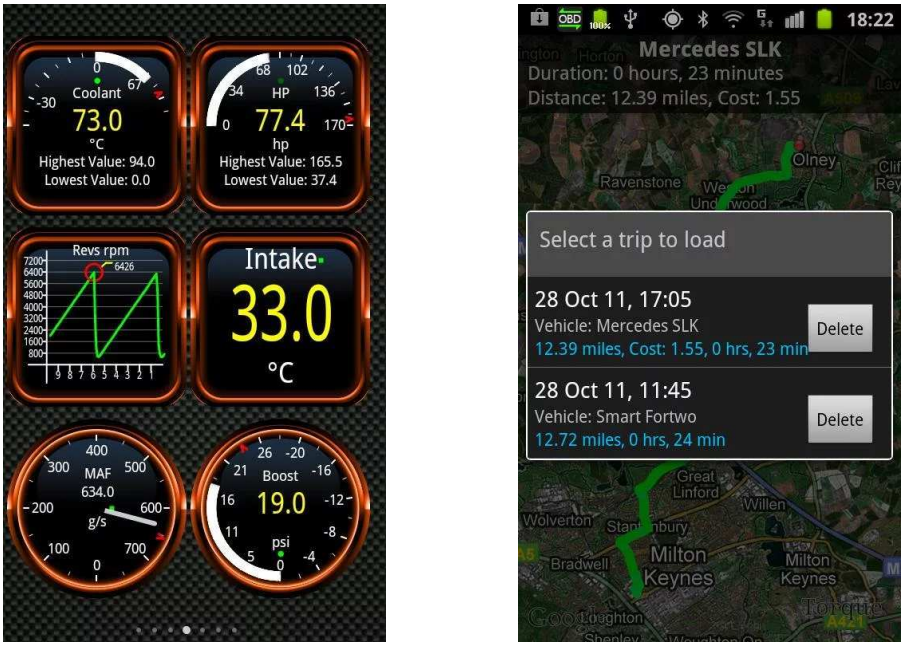
Mavia [13] podle webových stránek projektu nabízí podobné funkce jako aplikace Zubie. Opět se jedná o sociální síť pro řidiče automobilů. Je zde spojení s OBD2 jednotkou, sběr dat, lokace a propojení s ostatními uživateli. Mezi další funkce patří diagnostika vozu, podobně jako u aplikace Zubie.

3.3 Torque

Velmi rozšířená a oblíbená diagnostická aplikace používající OBD2 rozhraní [14]. Je schopná získat data z velkého množství senzorů, následně je zpracovat, zobrazit nebo zaznamenat. Dále aplikace nabízí rozsáhlé možnosti kastomizace uživatelského rozhraní, například úpravu zobrazovaných informací. Mezi hlavní funkce patří:

- Zobrazování informací o jízdě v reálném čase (rychlost, otáčky, teplota motoru, atd.).
- Záznam trasy a informací z řídící jednotky a následné sdílení.
- Zjišťování závad automobilu.

Na obrázcích 3.2a a 3.2b se nachází ukázka z aplikace Torque.



(a) Torque 1 (b) Torque 2

Obrázek 3.2: Snímky obrazovky aplikace Torque

3.4 Shrnutí

Z uvedeného průzkumu vyplývá, že aplikací, které se funkčností podobají aplikaci Sociocars, je několik. Kromě aplikace Torque, která se konceptu Sociocars podobá nejméně, nejsou tyto aplikace na trhu dlouho a nejsou moc rozšířené.

Mobilní aplikace Sociocars bude oproti uvedeným aplikacím blíže spolupracovat se serverem Sociocars, což by mohlo být uživatelsky zajímavé a chytlavé.

Funkcí, která mě na zkoumaných aplikacích nejvíce zaujala, a bylo by zajímavé ji přidat i do aplikace Sociocars, bylo hodnocení kvality řízení u aplikace Mavia. Aplikace podle všeho měří kvalitu řízení na základě rychlosti, zrychlování, přidávání plynu nebo frekvence řazení a na základě toho hodnotí řidiče.

Kapitola 4

Analýza tvorby aplikací pro platformu Android

4.1 Základní popis OS Android

Android je operační systém založený na linuxovém jádře určený především pro zařízení s dotykovým displejem, jako jsou mobilní telefony nebo tablety. Je vyvíjen konsorciem OHA (Open Handset Alliance) [15], do kterého spadá 84 technologických a mobilních společností, včetně společnosti Google, která do systému integruje velké množství svých produktů a je se systémem neodmyslitelně propojena [16].

Pro vývoj aplikací pro OS Android se používá programovací jazyk Java využívající Android SDK (Software Development Kit) [17].

Operační systém Android je vyvíjen pod Apache 2.0 a GPLv2 licencemi, což znamená, že je tzv. otevřeným softwarem. Díky tomu může být volně distribuován a upravovován společnostmi vyrábějícími mobilní zařízení [18].

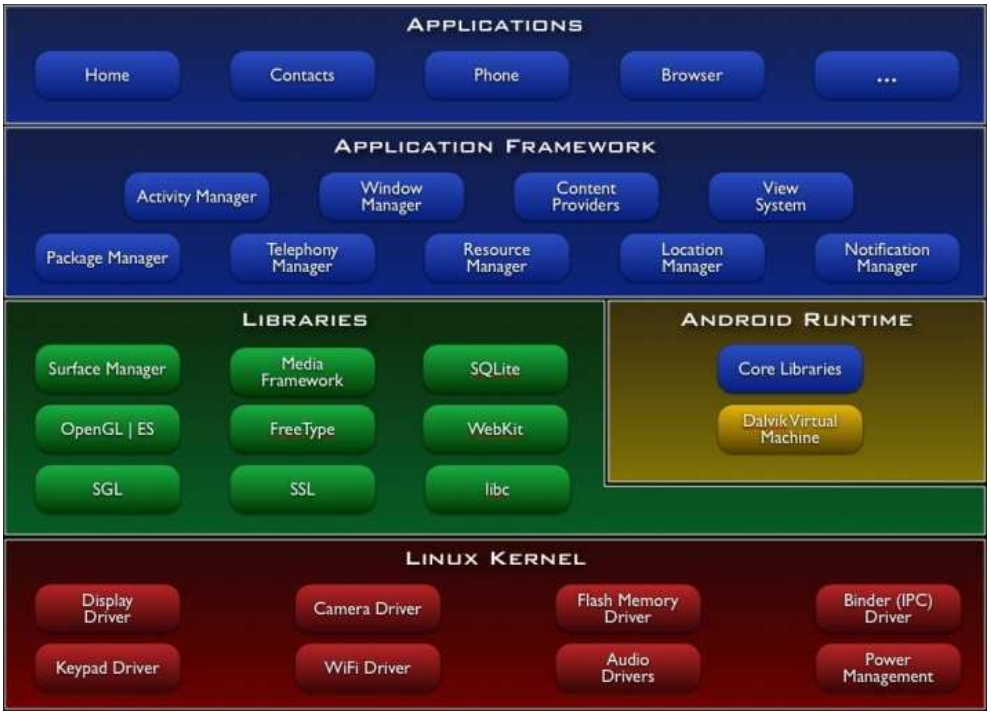
4.1.1 Historie operačního systému Android

Společnost Android Inc. byla založena v říjnu 2003 v Kalifornii, v roce 2005 byla koupena společností Google Inc.

Prvním oficiálním mobilním telefonem s operačním systémem Android se stal v říjnu 2008 mobilní telefon HTC Dream, s OS Android verze 1.0 [19].

4.2 Architektura operačního systému Android

Operační systém Android je rozdělen do pěti hlavních částí ve čtyřech vrstvách [20] 4.1:



Obrázek 4.1: Architektura operačního systému Android [1]

- **Linuxové jádro:** Jádro, na kterém je OS Android založen. Tato vrstva obsahuje všechny nízkoúrovňové ovladače, pro různorodé komponenty zařízení s OS Android.
- **Knihovny:** Tato vrstva obsahuje všechny knihovny, které poskytují hlavní funkce OS Android. Jedná se například o *SQLite* knihovnu, která zajišťuje podporu databáze, *WebKit* knihovnu, která poskytuje funkcionalitu pro prohlížení webu nebo *Media framework* knihovnu, která se stará o práci s médii.
- **Android runtime:** Na stejné vrstvě jako knihovny poskytuje Android runtime soubor základních Java knihoven, které vývojářům umožňují vyvíjet aplikace pro Android v programovacím jazyku Java. Android runtime kromě toho obsahuje Dalvik virtual machine (DVM), což je virtuální stroj vyvíjený speciálně pro OS Android. DVM se stará hlavně o spouštění aplikací a běh procesů a je optimalizovaný pro mobilní zařízení. Dále zajišťuje to, aby byla každá aplikace spuštěna ve svém vlastním vlákně s vlastní instancí DVM.
- **Aplikační framework:** Poskytuje možnosti OS Android vývojářům, díky čemuž je mohou využít ve svých aplikacích. Obsahuje například:
  - Activity manager: Stará se o základní stavební prvky Android aplikace, aktivity. [21]

  - Location manager: Poskytuje přístup k systémovým lokačním službám. [22]
  - Content provider: Umožňuje ukládat data a sdílet je mezi aplikacemi. [23]

- **Aplikace:** Nejvyšší vrstva, ve které se nacházejí samotné aplikace, jako například aplikace pro telefonní hovory, zprávy nebo internetový prohlížeč.

4.3 Zásady tvorby aplikací pro Android

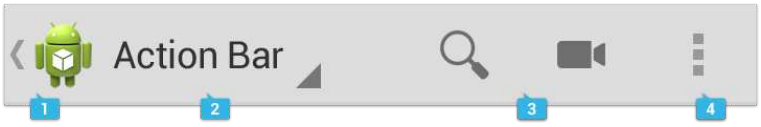
Aplikace je od začátku tvořena pro platformu Android a proto je nutné ji navrhovat v souladu s touto platformou, jejími pravidly a doporučeními z oficiálních stránek [16]. V této části bude tedy popsáno, jak má aplikace pro platformu Android vypadat a jaké v ní mají být použité grafické a ovládací prvky.

4.3.1 Doporučené rozložení obrazovky

- Horní panel, neboli action bar s navigačními a ovládacími prvky.
- Vlastní obsah aplikace.
- Spodní lišta, obsahující tlačítka pro ovládání systému.

4.3.1.1 Action bar

Action bar, viz. obrázek 4.2 většinou obsahuje ikonu s tlačítkem pro návrat o úroveň výš(1), rozklikávací menu(2), ovládací prvky aplikace(3) a nabídku dalších ovládacích prvků(4).



Obrázek 4.2: Action bar

4.3.1.2 Vlastní obsah

Zde se nachází hlavní část aplikace, například mapa s vyhledaným cílem jako v tomto případě. Viz obrázek 4.3.

Obrázek 4.3: Vlastní obsah

4.3.1.3 Spodní lišta

Ve spodní části se nachází systémová tlačítka, viz. obrázek 4.4.

Obrázek 4.4: Spodní lišta

4.4 Postup vývoje aplikace Sociocars

Na základě informací popsaných v této kapitole, probíhal vývoj aplikace Sociocars v následujících krocích:

1. Navržení základní struktury aplikace a obrazovek s odpovídajícími případy užití
2. Implementace aplikace
3. Testování
4. Oprava chyb nalezených při testování

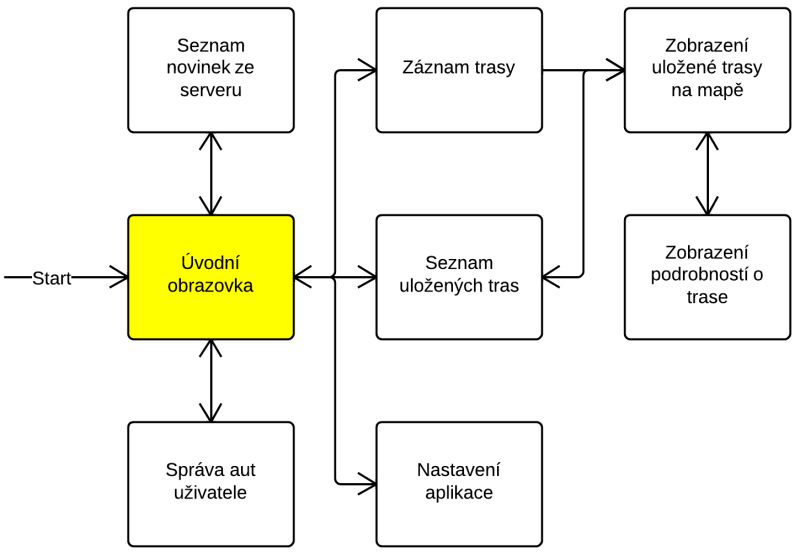
Kapitola 5

Návrh uživatelského rozhraní

Tato kapitola se věnuje návrhu uživatelského rozhraní, které je navrženo v souladu s analýzou projektu. Jsou zde zobrazeny jednotlivé části aplikace, u kterých jsou pro co největší srozumitelnost popsány odpovídající případy užití.

5.1 Návrh posloupnosti akcí v aplikaci

Nejprve byl vytvořen návrh posloupnosti akcí v aplikaci a zároveň definice základních obrazovek aplikace, viz. obr. 5.1.



Obrázek 5.1: Návrh posloupnosti akci v aplikaci

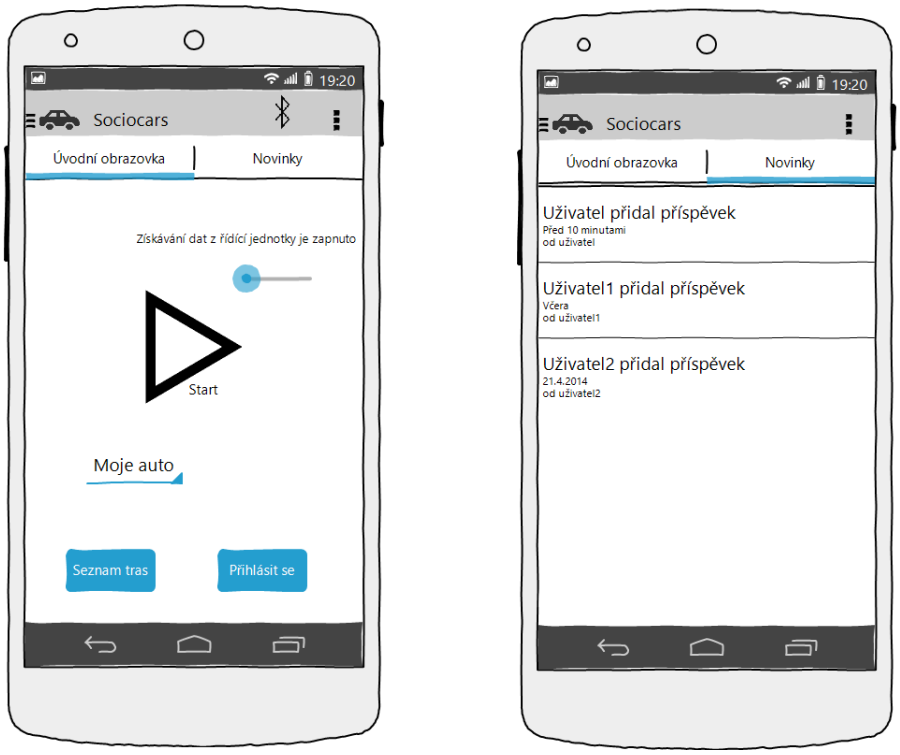
5.2 Návrh jednotlivých obrazovek aplikace

V této části jsou představeny návrhy obrazovek aplikace, u kterých jsou vypsány odpovídající případy užití, které mohou být vykonávány dvěma uživatelskými rolemi, nepřihlášeným a přihlášeným uživatelem.

5.2.1 Úvodní obrazovka

Obrazovka, která se uživateli zobrazí po spuštění aplikace, by měla být přehledná, uživatelsky přívětivá a měla by umožnit rychlé přepínaní mezi zobrazení základních ovládacích prvků a zobrazení novinek ze serveru. Uživatel by se z ní měl možnost navigovat ke správě svých vozidel, spuštění samotného záznamu trasy, k seznamu již uložených tras a případně k dalším funkcím nebo nastavením aplikace.

Jak je vidět na obrázcích 5.2a a 5.2b, obsahuje návrh úvodní obrazovky dva panely, přičemž se na prvním nachází hlavní ovládací prvky aplikace a na druhém novinky ze serveru.



(a) Návrh panelu s ovládacími prvky

(b) Návrh panelu s novinkami ze serveru

Obrázek 5.2: Návrh úvodní obrazovky

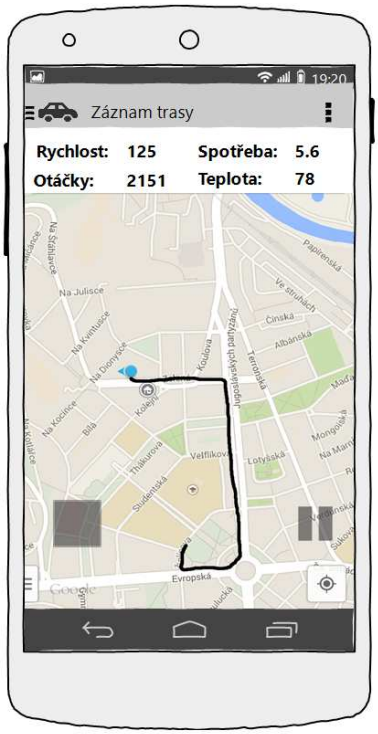
5.2.1.1 Případy užití pro úvodní obrazovku

- **Nepřihlášený uživatel**
  - Přihásit se
  - Spustit záznam trasy
  - Vypnout/ zapnout získávání dat z řídící jednotky
  - Zvolit bluetooth zařízení pro připojení k řídící jednotce
  - Zvolit zobrazení seznamu uložených tras
- **Přihlášený uživatel**
  - Odhlásit se
  - Zvolit zobrazení panelu s novinkami ze serveru
  - Zvolit zobrazení panelu s ovládacími prvky
  - Zobrazit komentáře příspěvku
  - Přidat komentář k příspěvku
  - Aktualizovat novinky ze serveru
  - Zvolit vozidlo ze seznamu vozidel získaného ze serveru

5.2.2 Záznam trasy

Obrazovka, která se spustí po zahájení záznamu trasy, by měla obsahovat aktuální statistiku probíhající jízdy (údaje z řídící jednotky vozidla, údaje z GPS, atd.) a mapu s aktuální polohou a s již zaznamenanou trasou. Dále by měla obsahovat ovládací prvky záznamu. Po ukončení záznamu trasy by měl být uživatel přesměrován na obrazovku se zobrazením zaznamenané trasy.

Jak je vidět na obrázku 5.3, obsahuje návrh této obrazovky mapu s aktuálně zaznamenanou trasou a data přijímaná z řídící jednotky.



Obrázek 5.3: Obrazovka se záznamem trasy

5.2.2.1 Případy užití pro obrazovku se záznamem trasy

- **Nepřihlášený uživatel**
  - Pozastavit záznam
  - Znovuspustit pozastavený záznam
  - Ukončnit a uložit záznam
  - Zobrazit aktuální polohu
  - Přiblížit mapu
  - Oddálit mapu

5.2.3 Seznam uložených tras

Obrazovka se seznamem uložených tras by měla zobrazovat seznam zaznamenaných tras, s možností zvolení vybrané trasy a následným zobrazením detailu zvolené trasy.

V návrhu této obrazovky, jak viz. obrázek 5.4, se nachází seznam tras, které jsou uložené v zařízení.

Obrázek 5.4: Obrazovka se seznamem tras

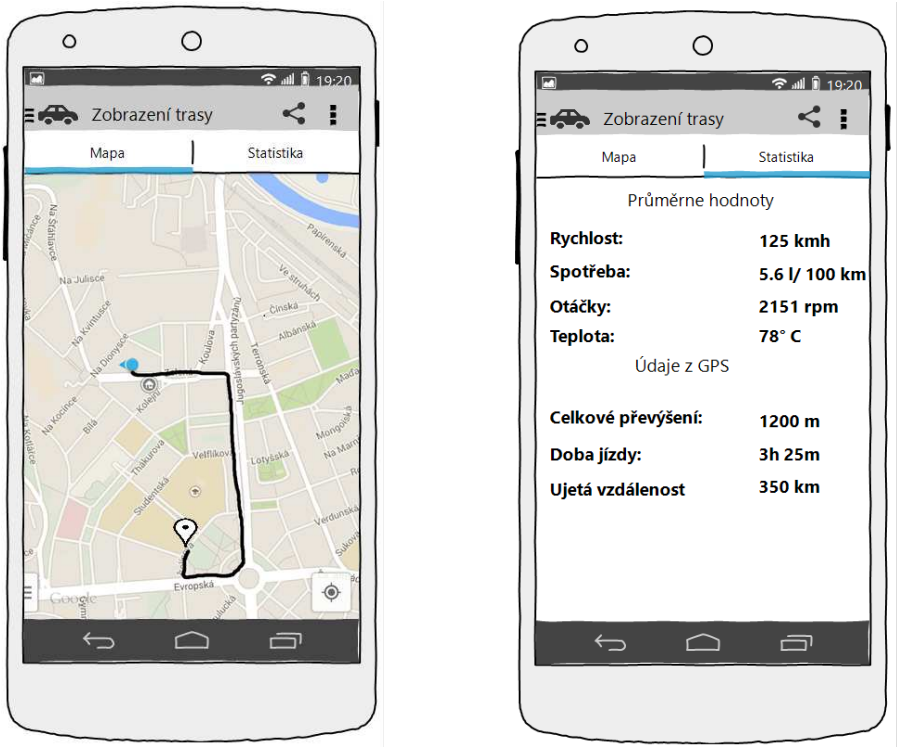
5.2.3.1 Případy užití pro obrazovku se seznamem tras

- **Nepřihlášený uživatel**
  - Zvolit trasu ze seznamu
  - Vrátit se na úvodní obrazovku

5.2.4 Zobrazení uložené trasy

Obrazovka se zobrazením detailu uložené trasy by měla obsahovat mapu se zobrazenou trasou a možnost rychlého přepnutí na zobrazení statistiky trasy (zpracovaná data z řídící jednotky, data z GPS, atd.).

Jak je vidět na obrázcích 5.5a a 5.5b, návrh této obrazovky obsahuje dva panely, kde se na prvním nachází mapa se zaznamenanou trasou a na druhém statistika trasy získaná z dat z řídící jednotky a z GPS.



(a) Návrh panelu s mapou trasy (b) Návrh panelu se statistikami

Obrázek 5.5: Návrh obrazovky se zobrazením trasy

5.2.4.1 Případy užití pro obrazovku se zobrazenou trasou

- **Nepřihlášený uživatel**
  - Zvolit zobrazení panelu se statistikou trasy
  - Zvolit zobrazení panelu s mapou trasy
  - Vrátit se na seznam uložených tras
  - Vrátit se na úvodní obrazovku
  - Smazat trasu
- **Přihlášený uživatel**
  - Nahrát trasu na server Sociocars

Kapitola 6

Realizace

Tato kapitola se věnuje samotné realizaci aplikace, nastiňuje její fungování a popisuje řešení problémů, které musely být v průběhu vývoje řešeny.

Aplikace byla vyvíjena ve vývojovém prostředí Eclipse s nainstalovaným rozšířením pro Android. Aplikace je napsána v programovacím jazyce Java.

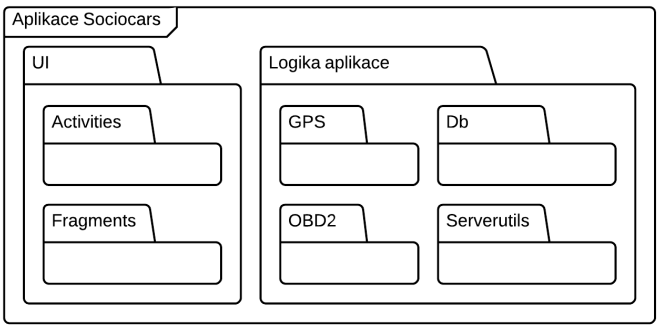
6.1 Struktura

Při vývoji aplikace jsem usiloval o rozdělení částí, které souvisejí s GUI, neboli uživatelským rozhraním, a částí, které obstarávají logiku a funkčnost aplikace.

Interakci s uživatelem mají v případě Android aplikací na starost tzv. aktivity [24]. Každá obrazovka aplikace je definována jednou aktivitou (ve formě instance třídy rozšiřující třídu Activity) a může se dělit na tzv. fragmenty. Fragment [25] je samostatně funkční celek, který reprezentuje určitou část uživatelského rozhraní a musí být součástí aktivity. Aktivita může obsahovat více fragmentů. Samotný vzhled každé aktivity/ fragmentu je definován v odděleném XML souboru.

V aktivitě se mohou odchytávat vstupy uživatele nebo mu naopak poskytovat obsah a řešit základní logiku aplikace.

Na základě této struktury je implementace rozdělena do balíčků, viz. obr. 6.1.



Obrázek 6.1: Diagram balíčků

6.2 Problémy a jejich řešení

V této části je přehled hlavních problémů, se kterými jsem se musel při vývoji vypořádat a stručný popis jejich řešení.

6.2.1 Získávání polohy

Jak již bylo zmíněno v analýze, pro přístup k poloze je použita třída LocationManager z balíčku android.location [5]. Vytvořené instanci této třídy je přiřazen listener implementující rozhraní LocationListener ze stejného balíčku [26]. Nejdůležitější metodou z tohoto rozhraní je metoda *onLocationChanged* s parametrem typu Location, která se zavolá pokaždé když se změní poloha. Při každé změně polohy je možné pracovat s objektem typu Location, který obsahuje všechny potřebné informace, jako aktuální poloha, rychlost nebo nadmořská výška. Poté je objekt uložen do databáze a zaslán zpátky do aktivity, ve které které je záznam spuštěn (TrackRouteActivity).

Ukázka metody *onLocationChanged*:

```
public void onLocationChanged(Location loc) {
        /*
         * K ulozeni souradnic doje pouze tehdy, pokud ma lokace dostatecnou
         * presnost a pokud ubehla vice jak sekunda od posledniho ulozeni
         * souradnic.
         */
        if (loc.getAccuracy() != 0
                        && System.currentTimeMillis() - locTime > 1000
                        && loc.getAccuracy() < 50) {

                        locTime = System.currentTimeMillis();
                /*
                 * Ulozeni souradnic do databaze.
                 */
                insertToDatabase(loc);
                /*
                 * Poslani lokace zpatky do aktivity, ve ktere je zaznam spuseten.
                 */
                setDelegateLocs(loc);

                }

        }
```

Listing 6.1: Metoda onLocationChanged

6.2.2 Získávání dat z OBD2 jednotky

Pro komunikaci s OBD2 jednotkou byla použita část kódu z android aplikace projektu Metrocars [3] vyvíjené Romanem Kubů v rámci bakalářské práce [6] a je k projektu připojena jako externí knihovna.

Obsluha jednotky běží v jednom vlákně spouštěném ve třídě metrocar.engine.UnitEngine. Instanci této třídy stačí přiřadit odpovídající Handler, aplikační kontext a zařízení bluetooth. Poté se sama připojí k OBD2 jednotce, zajistí přípravu jednotky a přijímá poskytované informace, které posílá dále do Handleru v aplikaci Sociocars. Získaná data se poté ukládají do databáze a zobrazují uživateli ve stejném intervalu jako data z GPS.

6.2.3 Ukládání zaznamenaných tras

Zaznamenané trasy je potřeba nějak ukládat. V analýze se již rozhodlo o ukládání tras do formátu KML, ale ukládání dlouhodobě získávaných dat do souboru v úložišti není nejlepší řešení. Ukládání se tedy rozdělilo na dvě fáze:

6.2.3.1 Ukládání do databáze v průběhu záznamu

Android poskytuje plnou podporu SQLite databáze [27] a díky poskytovanému API je velice jednoduché s ní pracovat.

Poněvadž jsou údaje z databáze potřebné pouze po dobu záznamu (po ukončení záznamu se uloží do KML souboru), obsahuje databáze pouze jednu tabulku se sloupci odpovídajícími získávaným datům (lokace, data z OBD2 jednotky). Data se z databáze po ukončení záznamu a uložení trasy vymažou.

6.2.3.2 Ukládání do KML souboru po skončení záznamu

Ve chvíli, kdy uživatel zvolí ukončení záznamu, dojde k překopírování dat z databáze do KML souboru. Soubory se ukládají do složky sociocars v úložišti, aby k nim měl uživatel snadný přístup, a mohl je lehce vyexportovat a zobrazit například v aplikaci Google Earth [28].

O ukládání do KML souboru se starají třídy z balíčku javax.xml.parsers [29] a org.w3c.dom [30], pomocí kterých se objekt typu Document postupně naplní daty z databáze a poté uloží do úložiště telefonu.

Ukázka KML formátu pro uložení dat z řídící jednotky:

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://www.opengis.net/kml/2.2">
   <Document>
       <LineString>
               <coordinates>
                      14.329588,50.011978,0.0
                      14.329588,50.011978,0.0
               </coordinates>
       </LineString>
   </Document>
   <unitinfo>
               <values>
                      107,2840,19,46
                      108,2850,19,46
               </values>
      <averagespeed>107.5</averagespeed>
      <averagerpm>2845.0</averagerpm>
      <averagethrottle>19.0</averagethrottle>
      <averagetemp>46.0</averagetemp>
   </unitinfo>
</kml>
```

Listing 6.2: Ukázka KML

První část dokumentu, v elementu <Document>, je originální KML. Druhá část, v elementu <unitinfo>, je doplňující část vytvořená pro uložení dat z řídící jednotky.

6.2.4 Zobrazení trasy

K zobrazení trasy a jejích statistik se může uživatel navigovat dvěma způsoby:

- Zvolí seznam tras, což spustí aktivitu TrackListActivity, ve které se do seznamu načtou všechny KML soubory ze složky sociocars umístěné v úložišti telefonu. K zobrazení trasy dojde po kliknutí na vybranou trasu.
- Trasa se zobrazí ihned po ukončení záznamu.

V obou případech dojde k načtení KML soubouru z úložiště telefonu a následnému zpracování dat pomocí tříd z balíčku org.w3c.dom [30].

Aktivita se zobrazením trasy je rozdělena na dva fragmenty, jeden se zobrazením mapy a jeden se zobrazením statistiky. Na mapě je trasa zobrazena pomocí objektu PolylineOptions [31], do kterého se postupně nahrají souřadnice z KML souboru. Ve fragmentu se statistikami se zobrazí souhrnné informace získané z KML souboru, viz. ukázka KML souboru.

6.2.5 Komunikace se serverem

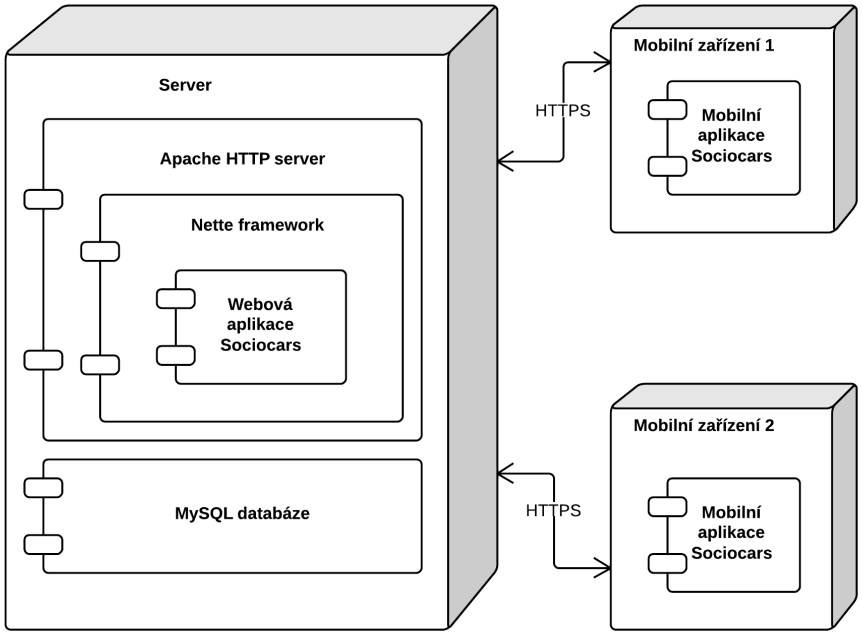
Komunikace se serverem Sociocars, poskytujícím REST rozhraní, probíhá pomocí asynchronní úlohy. Konkrétně se jedná o použití třídy WebServiceTask, rozšiřující třídu AsyncTask [32].

Asynchronní úloha poskytuje možnost vykonávat procesy na pozadí bez narušení vlákna uživatelského prostředí. Úloha se spustí, běží na pozadí a když je vykonána, vrátí výsledky do vlákna uživatelského rozhraní. V aplikaci Sociocars je toto použito na posílání POST a GET požadavků na REST rozhraní serveru a následné přijímaní odpovědí.

Komunikace se serverem probíhá pomocí zabezpečeného HTTPS spojení, díky tomu je možné vyměňovat si se serverem i citlivá data, jako například heslo pro ověření přihlášení uživatele.

Pro přenos se me mezi serverem a aplikací využívají JSON zprávy, jak bylo uvedeno v analýze. Pro jednoduché použiti formátu JSON v jazyce Java byla použita knihovna java-json [33].

Pro lepší představu o komunikaci aplikace se serverem byl vytvořen diagram nasazení, viz. obr. 6.2.



Obrázek 6.2: Diagram nasazení

Aplikace využívá spojení se serverem v mnoha případech, například při přihlášení, odesílání trasy na server nebo zobrazení příspěvků. V následujících podkapitolách je uveden přehled a popis hlavních případů komunikace.

6.2.6 Ověření uživatele na serveru

Poté, co se uživatel pokusí v aplikaci přihlásit, je potřeba ověřit jeho přihlašovací údaje vůči serveru. Díky zabezpečenému HTTPS spojení není nutné data nijak šifrovat. Aplikace pošle uživatelem zadané přihlašovací údaje na server ve formátu:

```
{
   "email":"test@test.com",
   "password":"password"
}
```

Listing 6.3: Příklad JSON zprávy s přihlašovacími údaji

Server může odpovědět chybovou hláškou v případě špatných přihlašovacích údajů, nebo zprávou s detaily uživatele, jak je uvedeno v následujícím příkladě:

```
 {
    "status":"ok",
    "user":{
       "id":1,
       "email":"test@test.com",
       "name":"Test",
       "surname":"Test",
       "sex":"male",
       "city":"Prague"
    }
 }
```

Listing 6.4: Příklad JSON zprávy o úspěšném přihlášení

6.2.6.1 Získání seznamu vozidel uživatele

Aby si mohl uživatel vybrat, k jakému vozidlu ze serveru se bude vztahovat jeho zaznamenaná trasa, je nutné získat ze serveru seznam aut daného uživatele. Aplikace pošle na server požadavek obsahující id uživatele o získání seznamu vozidel. Server odpoví zprávou se seznamem aut daného uživatele.

```
{
   "status":"ok",
   "cars":[
      {
         "id":1,
         "users_id":1,
         "name":"Auto",
         "desc":"Moje auto",
         "registration_number":"7A51443",
         ...
      }
```

```
   ]
}
```

Listing 6.5: Příklad JSON zprávy se seznamem aut uživatele

6.2.6.2 Nahrání ujeté trasy na server

Při zvolení možnosti nahrání ujeté trasy na server se nejdříve trasa uložená v úložišti telefonu ve formátu KML převede do formátu JSON. S tím jsou do ní přidány informace o přihlášeném uživateli a vozidle, ke kterému chce trasu na serveru přiřadit. Poté je JSON zpráva odeslána na server.

```
{
   "car_id":1,
   "password":"password",
   "user_id":1,
   "entries":[
      {
         "timestamp":"2012-10-24T01:00:27.372Z",
         "velocity":95,
         "engine_temp":81,
         "throttle":10,
         "engine_rpm":2505,
         "altitude":230.8000030517578,
          "location":[
                    14.39404003,
                    50.11267845
                ]
      }
   ]
}
```

Listing 6.6: Příklad JSON zprávy s trasou

6.2.6.3 Newsfeed

Další funkcí aplikace, ke které je potřeba komunikace se serverem, je získání příspěvků ze serveru, jejich komentářů a možnost je okomentovat. Aplikace odešle serveru požadavek na seznam příspěvků a jejich komentářů pro konkrétního uživatele a ten odpoví JSON zprávou s požadovanými příspěvky.

```
{
   "user_id":1,
   "posts":[
      {
         "id":1,
         "author":"John",
         "date":"2012-10-24T01:00:27.372Z",
         "text":"Uzivatel neco udelal",
         "comments":[
            {
               "id":11,
               "author":"Petr",
               "date":"2012-10-24T01:00:27.372Z",
               "text":"Uzivatel okomentoval prispevek"
            }
         ]
      }
   ]
}
```

Listing 6.7: Příklad JSON zprávy s příspěvky pro konkrétního uživatele

Uživatel má poté možnost zobrazit příspěvky, včetně jejich komentářů a má možnost okomentovat konkrétní příspěvek. Po okomentování uživatelem aplikace odešle na server zprávu s id uživatele, id zprávy kterou komentuje a samotným komentářem.

```
{
   "user_id":1,
   "post_id":12,
   "text":"Komentuju prispevek"
}
```

Listing 6.8: Příklad JSON zprávy s komentářem příspěvku

Kapitola 7

Testování

Po implementaci aplikace bylo nutné ověřit její funkčnost, spolehlivost a kompatibilitu. Tato kapitola se věnuje testování aplikace. Jsou zde popsány metody, kterými byla aplikace testována a jejich výsledky.

7.1 Testování pomocí nástroje Robotium

Robotium [34] je framework pro automatizované testování Android aplikací. Je určený k testování scénářů případů užití. S jeho pomocí lze tedy nasimulovat uživatelské chování a zhodnotit výsledky. Testování pomocí nástroje Robotium se řadí do skupiny tzv. black box testování (testování bez znalosti vnitřní struktury testovaného systému). Testování je podobné testování populárním nástrojem Selenium [35], který je určen pro automatizované testování webu.

Samotné testy vycházejí z JUnit testů. Aplikace je ovládána pomocí objektu Solo, který poskytuje funkce pro navigaci v aplikaci nebo pro ověřování jejích výstupů.

Pomocí nástroje Robotium byly vytvořeny testy hlavních scénářů užití, které v aplikaci mohou nastat.

- Přihlášení a odhlášení.
- Záznam trasy, uložení trasy do úložiště zařízení, zobrazení trasy.
- Vybrání trasy ze seznamu tras uložených v úložišti zařízení, zobrazení trasy, nahrání trasy na server.
- Získání příspěvků ze serveru, zobrazení konkrétního příspěvku a komentářů, přidání komentáře.

Pomocí těchto testů šlo jednoduše a automaticky testovat funkčnost uživatelského rozhraní a celé aplikace po každé změně v implementaci.

Další výhodou, kterou poskytlo vytvoření těchto testů, bylo rychlé testování aplikace na různých zařízeních. Ideální je aplikaci otestovat na co největším počtu zařízení, v domácích podmínkách jich byl ale pouze omezený počet. Tento problém pomohl vyřešit nástroj AppThwack [36], který je popsán v následující části.

7.2 Testování pomocí nástroje AppThwack

AppThwack je nástroj poskytující možnost vzdáleného testování mobilních aplikací na reálných zařízeních. Podle popisu je možné využít až několik set zařízení s OS Android od mnoha různých výrobců. Použití nástroje je zpoplatněno, platí se za čas který na zařízeních zabere testování. Při registraci dostane uživatel 100 minut testování zdarma a navíc jsou k dispozici zařízení, jejichž využití není zpoplatněno.

Použití nástroje je velice jednoduché, stačí do něj nahrát testovanou aplikaci a zvolit způsob testování. V tomto případě testování pomocí Robotium testů. Poté už stačí pouze nahrát vytvořené Robotium testy a spustit.

7.2.1 Výsledek testování pomocí AppThwack

Testování pomocí nástroje AppThwack dopadlo nad očekávání dobře. Aplikace byla několikrát otestována na 20 zařízeních různých výrobců. Při každém testování došlo na každém zařízení k:

- nainstalování aplikace,
- spuštění všech testovacích uvedených v předchozí části,
- odinstalování aplikace.

To znamená, že při každém testu proběhlo 80 testovacích scénářů. Úspěšnost se pohybovala kolem 95%, přičemž k chybám docházelo pokaždé na jiných zařízeních a při různých testech. V žádném z logů zařízení, na kterých selhal nějaký test, nebyla nalezena chyba aplikace Sociocars. Nejspíš ani nedošlo k jejímu spuštění před testem, tím pádem ji testovací nástroj nemohl otestovat a hlásil test jako chybný. Z toho vyplývá, že chyby nejspíš nebyly na straně aplikace Sociocars.

Dalším ukazatelem, který nástroj nabízí, je sledování zátěže zařízení při testování. K dispozici je přehled využití procesoru, paměti RAM nebo rychlost vykreslování obrazovek v čase. Dále je k dispozici průměr těchto hodnot u všech zařízení za celou dobu běhu aplikace. Průměrné hodnoty jsou následující:

- **Průměrné využití procesoru**: 13.36 %
- **Průměrné využití paměti RAM**: 22.12 MB
- **Průměrný čas vykreslování obrazovek**: 22.83 ms

Celkově dopadlo testování nástrojem AppThwack velice dobře. Bylo ověřeno, že je aplikace kompatibilní s velkým množstvím zařízení a svými hardwarovými požadavky nijak nevybočuje z nabídky ostatních aplikací.

7.3 Testování v reálném prostředí

Testování probíhalo po celou dobu vývoje hlavně v domácích podmínkách, s umělými GPS daty a s řídící jednotkou v podobě simulátoru [37]. Pomocí tohoto testování bylo odhaleno a opraveno velké množství chyb. I přes to bylo potřeba otestovat aplikaci v reálném prostředí.

Pro účely tohoto testování byl zakoupen OBD2 bluetooth adaptér *ELM 327 Super Mini Bluetooth OBD II V1.5* [38], který má v technické specifikaci uvedenou podporu Android zařízení.

Samotné testování probíhalo s následující konfigurací:

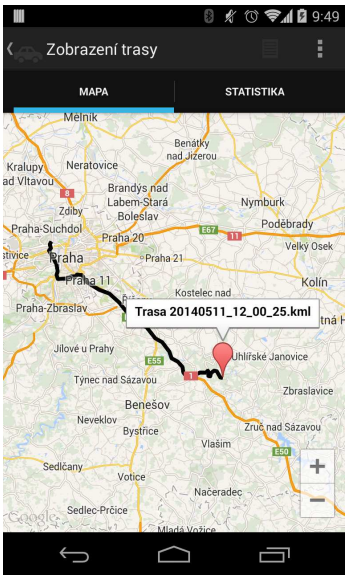
- **Automobil:** Škoda Octavia II
- **OBD2 adaptér:** ELM 327 Super Mini Bluetooth OBD II V1.5
- **Mobilní telefon:** LG Nexus 4
- **Verze OS:** Android 4.4.2 KitKat

Po zapojení blueotooth adaptéru do OBD2 konektoru automobilu bylo nutné ho spárovat mobilním telefonem a poté spustit aplikaci Sociocars. Aplikace se na první pokus spojila s řídící jednotkou a začala ukazovat správné hodnoty.

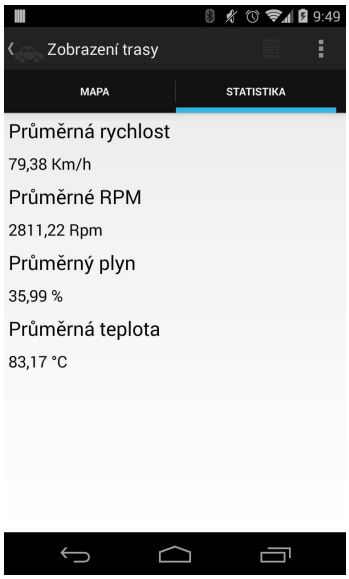
Testovací trasa byla dlouhá 60 kilometrů a trvala asi 50 minut. Trasa vedla po běžných silnicích, dálnici i městem. Na trase se nacházely 3 tunely. Během záznamu nedošlo k žádnému problému, po celou dobu aplikace získávala a ukládala data z řídící jednotky i z GPS.

Během záznamu byla po celou dobu zaplá obrazovka mobilního zařízení a baterie se vybila o 25%. Což je, v porovnání například s navigací Google, přijatelná hodnota.

Na konci trasy byl záznam ukončen a trasa se úspěšně uložila do úložiště mobilního zařízení. Zaznamenaná trasa je vidět na obrázcích 7.1a a 7.1b.



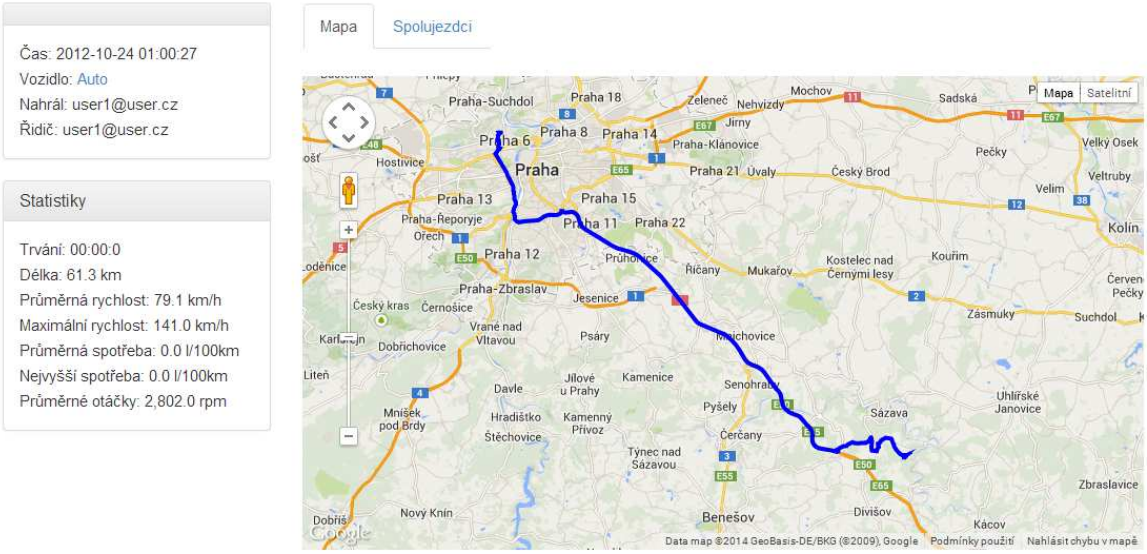
(a) Panel s mapou



(b) Panel se statistikou

Obrázek 7.1: Snímek obrazovky se zaznamenanou trasou

Po ukončení záznamu trasy bylo provedeno přihlášení na server Sociocars a zvolení automobilu. Poté bylo zvoleno nahrání trasy na server Sociocars, což úspěšně proběhlo, jak je vidět na obrázku 7.2.



Obrázek 7.2: Zobrazení trasy na serveru

Testování v reálném prostředí tedy proběhlo velmi dobře. Aplikace se bez problémů spojila s náhodně koupeným OBD2 bluetooth adaptérem a na první pokus zaznamenala relativně dlouhou trasu.

Toto testování nicméně ukázalo i nedostatky aplikace, které se musí v budoucnosti vyřešit. Patří mezi ně:

- Nepřehledné uživatelské rozhraní při záznamu trasy,
- nutnost nechat aplikaci při záznamu v popředí. Záznam trasy a komunikace s řídící jednotkou by měly být v aplikaci implementovány jako služba [39].

Kapitola 8

Závěr

Tato kapitola se věnuje shrnutí hlavních cílů a možnému pokračování práce do budoucna.

8.1 Možné pokračování práce

Zejména kvůli časovým problémům se nedostalo na realizaci spousty nápadů, které byly objeveny při implementaci a testování aplikace.

Jedná se například o rozšíření funkčnosti související s řídící jednotkou automobilu, jako přijímání více informací o jízdě nebo chybových hlášení. Dále se jedná o již zmiňované předělání zaznamenávání trasy a komunikace s řídící jednotkou na Android služby, což by umožňovalo pohodlnější používání aplikace.

Asi nejdůležitější oblastí, ve které se aplikace Sociocars zatím nemůže rovnat konkurenci, je uživatelské rozhraní. Poněvadž byla většina času věnována hlavně vývoji a testování funkčnosti, nezbylo moc času na vývoj a testování uživatelského rozhraní. Pokračování práce by se proto mělo primárně věnovat uživatelskému rozhraní.

8.2 Závěrečné zhodnocení

V rámci práce byla zanalyzována, navržena, naimplementována a otestována mobilní aplikace Sociocars, která je součásti projektu sociální sítě Sociocars.

Výsledkem je funkční aplikace pro operační systém Android, která je schopná zaznamenávat trasu, komunikovat s řídící jednotkou, zobrazovat trasy a komunikovat se serverem Sociocars. Byly splněny požadavky určené v zadání a já jsem s výsledkem své práce spokojen.

Literatura

[1] "Architektura OS Android."
<http://developer.android.com/images/system-architecture.jpg>,
stav ze 1. 5. 2014.

[2] "Sociocars."
<https://gitlab.fel.cvut.cz/b131_bp_macekond/skalito1_socio_cars>,
stav ze 5. 5. 2014.

[3] "Metrocars."
<https://www.assembla.com/spaces/wagnejan_metrocar/wiki>,
stav ze 1. 5. 2014.

[4] "OBD 2."
<http://www.obdii.com/background.html>,
stav ze 1. 5. 2014.

[5] "Android developers Location."
<http://developer.android.com/reference/android/location/
package-summary.html>,
stav ze 26. 4. 2014.

[6] R. Kubů, "Android aplikace Metrocars."
<https://dip.felk.cvut.cz/browse/pdfcache/kuburoma_2013bach.pdf>,
stav ze 1. 5. 2014.

[7] "Gpx."
<http://www.topografix.com/gpx.asp>,
stav ze 1. 5. 2014.

[8] "Kml Google."
<https://developers.google.com/kml/>,
stav ze 1. 5. 2014.

[9] "Open Geospatial Consortium."
<http://www.opengeospatial.org/standards/kml/>,
stav ze 1. 5. 2014.

[10] "Rozhraní REST."
<https://www.ibm.com/developerworks/webservices/library/ws-restful/>,
stav ze 1. 5. 2014.

[11] "Oficiální web JSON."
<http://www.json.org/>,
stav ze 7. 5. 2014.

[12] "Zubie."
<http://zubie.co/>,
stav ze 1. 5. 2014.

[13] "Mavia."
<http://mavizon.com/>,
stav ze 1. 5. 2014.

[14] "Torque."
<https://play.google.com/store/apps/details?id=org.prowl.torque>,
stav ze 1. 5. 2014.

[15] "Open Handset Alliance."
<http://www.openhandsetalliance.com/>,
stav ze 1. 5. 2014.

[16] "Android developer."
<http://developer.android.com>,
stav ze 1. 5. 2014.

[17] "Android SDK."
<http://developer.android.com/sdk/index.html>,
stav ze 1. 5. 2014.

[18] "Android licence."
<http://source.android.com/source/licenses.html>,
stav ze 1. 5. 2014.

[19] "Historie OS Android."
<http://www.cnet.com/news/a-brief-history-of-android-phones/>,
stav ze 1. 5. 2014.

[20] Lee Wei-meng, *Android 4 Application Development*. 10475 Crosspoint Boulevard, Indianapolis: John Wiley & sons, 1st ed., 2012.

[21] "Activity manager, Android developer."
<http://developer.android.com/reference/android/app/ActivityManager.html>,
stav ze 1. 5. 2014.

[22] "Location manager, Android developer."
<http://developer.android.com/reference/android/location/LocationManager.html>,
stav ze 1. 5. 2014.

[23] "Content provider, Android developer."
<http://developer.android.com/reference/android/content/ContentProvider.html>,
stav ze 1. 5. 2014.

[24] "Activity, Android developer."
<http://developer.android.com/reference/android/app/Activity.html>,
stav ze 1. 5. 2014.

[25] "Fragment, Android developer."
<http://developer.android.com/guide/components/fragments.html>,
stav ze 1. 5. 2014.

[26] "Location listener, Android developer."
<http://developer.android.com/reference/android/location/LocationListener.html>,
stav ze 1. 5. 2014.

[27] "SQLite, Android developer."
<http://developer.android.com/guide/topics/data/data-storage.html>,
stav ze 1. 5. 2014.

[28] "Google Earth."
<http://www.google.cz/intl/cs/earth/>,
stav ze 1. 5. 2014.

[29] "Knihovna javax.xml.parsers."
<http://docs.oracle.com/javase/7/docs/api/javax/xml/parsers/package-summary.html>,
stav ze 1. 5. 2014.

[30] "Knihovna org.w3c.dom."
<http://docs.oracle.com/javase/7/docs/api/org/w3c/dom/package-summary.html>,
stav ze 1. 5. 2014.

[31] "Polyline options, Android developer."
<http://developer.android.com/reference/com/google/android/gms/maps/model/PolylineOptions.html>,
stav ze 1. 5. 2014.

[32] "Async task, Android developer."
<http://developer.android.com/reference/android/os/AsyncTask.html>,
stav ze 1. 5. 2014.

[33] "Knihovna java-json."
<http://www.json.org/java/>,
stav ze 1. 5. 2014.

[34] "Framework Robotium."
<https://code.google.com/p/robotium/>,
stav ze 1. 5. 2014.

[35] "Nástroj Selenium."
<http://docs.seleniumhq.org/>,
stav ze 1. 5. 2014.

[36] "Nástroj AppThwack."
<https://appthwack.com/overview>,
stav ze 1. 5. 2014.

[37] "OBD2 simulátor."
<http://icculus.org/obdgpslogger/obdsim.html>,
stav ze 7. 5. 2014.

[38] "ELM327 super mini Bluetooth v1.5."
<http://www.amazon.com/ELM327-Bluetooth-CAN-BUS-Diagnostic-Windows/dp/B008U1MOM8>,
stav ze 7. 5. 2014.

[39] "Service, Android developer."
<http://developer.android.com/guide/components/services.html>,
stav ze 7. 5. 2014.

Příloha A

Seznam použitých zkratek

OS Operační systém

OBD On-board diagnostics

XML Extensible Markup Language

KML Keyhole Markup Language

REST Representational state transfer

JSON JavaScript Object Notation

SDK Software development kit

GPL General Public License

DVM Dalvik virtual machine

GPS Global Positioning System

GUI Graphical user interface

HTTPS Hypertext Transfer Protocol Secure

Příloha B

Obsah přiloženého CD

- `readme.txt` ..... Stručný popis obsahu CD
- `Source`
  - `MetrocarEngine` ..... Eclipse projekt knihovny pro spojení s OBD2 jednotkou
  - `Sociocars_android` ..... Eclipse projekt samotné aplikace
  - `Sociocars_android_test` ..... Eclipse projekt robotium testů aplikace
- `text` ..... Adresář s textem bakalářské práce
  - `Sociocars.pdf` ..... Text bakalářské práce ve formátu PDF
- `Docs`
  - `Work` ..... Pomocné soubory k textu bakalářské práce
  - `Zprava` ..... Zdrojový kód textu bakalářské práce ve formátu LaTeX
    - `GUIprototype` ..... Adresář s obrázky návrhu GUI
    - `pics` ..... Adresář s obrázky použitými v bakalářské práci
- `apk` ..... Adresář s instalačním balíčkem aplikace
  - `Sociocars.apk` ..... Instalační balíček aplikace Sociocars pro Android
- `GUI_prototype` ..... WireframeSketcher projekt prototypu GUI