FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Název:	Aplikace pro vzdálené sledování polohy mobilního zařízení
Student:	Ondřej John
Vedoucí:	Ing. Miroslav Balík, Ph.D.
Studijní program:	Informatika
Studijní obor:	Webové a softwarové inženýrství
Katedra:	Katedra softwarového inženýrství
Platnost zadání:	Do konce letního semestru 2018/19

Pokyny pro vypracování

Cílem práce je návrh, implementace, otestování a nasazení mobilní aplikace pro OS Android sloužící ke vzdálenému sledování polohy mobilního zařízení. Aplikace bude fungovat v offline režimu, ovládání na dálku a hlášení polohy bude realizováno pomocí SMS zpráv. Budou podporovány telefony a tablety na platformě Google Android, které disponují GSM modulem.

1. analyzujte 5 používaných konkurenčních řešení dostupných v Google Play Store
2. popište možné způsoby lokalizace mobilního zařízení
3. navrhněte a popište architekturu mobilní aplikace
4. implementujte aplikaci a otestujte ji
5. zveřejněte aplikaci veřejně ke stažení včetně zdrojového kódu
6. zhodnoťte výslednou aplikaci, porovnejte s ostatními

Seznam odborné literatury

Dodá vedoucí práce.

Ing. Michal Valenta, Ph.D.
vedoucí katedry

doc. RNDr. Ing. Marcel Jiřina, Ph.D.
děkan

V Praze dne 29. ledna 2018

České vysoké učení technické v Praze
Fakulta informačních technologií
Katedra softwarového inženýrství

Bakalářská práce

Aplikace pro vzdálené sledování polohy mobilního zařízení

Ondřej John

Vedoucí práce: Ing. Miroslav Balík, Ph.D.

10. května 2018

Poděkování

Rád bych poděkoval panu Ing. Miroslavu Balíkovi, Ph.D. za vedení této práce a za čas, který mi během psaní práce věnoval. Děkuji také své rodině za podporu během celého studia.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval(a) samostatně a že jsem uvedl(a) veškeré použité informační zdroje v souladu s Metodickým pokynem o etické přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. V souladu s ust. § 46 odst. 6 tohoto zákona tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programů, jež jsou její součástí či přílohou, a veškeré jejich dokumentace (dále souhrnně jen „Dílo"), a to všem osobám, které si přejí Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli způsobem, který nesnižuje hodnotu Díla, a za jakýmkoli účelem (včetně užití k výdělečným účelům). Toto oprávnění je časově, teritoriálně i množstevně neomezené. Každá osoba, která využije výše uvedenou licenci, se však zavazuje udělit ke každému dílu, které vznikne (byť jen zčásti) na základě Díla, úpravou Díla, spojením Díla s jiným dílem, zařazením Díla do díla souborného či zpracováním Díla (včetně překladu), licenci alespoň ve výše uvedeném rozsahu a zároveň zpřístupnit zdrojový kód takového díla alespoň srovnatelným způsobem a ve srovnatelném rozsahu, jako je zpřístupněn zdrojový kód Díla.

V Praze dne 10. května 2018 ……

Odkaz na tuto práci

John, Ondřej. *Aplikace pro vzdálené sledování polohy mobilního zařízení.* Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2018.

Abstrakt

Tato bakalářská práce popisuje vznik Android aplikace pro vzdálené sledování polohy mobilního zařízení. Zabývám se nejprve operačním systémem Android a způsoby získávání informací o poloze mobilních zařízení, poté analyzuji podobná, již existující řešení. Hlavní náplní práce je popis celého procesu vzniku této aplikace, od analýzy přes návrh a samotnou implementaci až po testování a zveřejnění výsledné aplikace ke stažení. Výstupem je snadno použitelná aplikace sloužící k nalezení ztraceného telefonu i bez připojení k Internetu, mobilní zařízení se ale díky ní může proměnit třeba i v jednoduchý GPS alarm do auta. Aplikace je dostupná ke stažení v obchodě Google Play a její zdrojový kód je zveřejněn na GitHubu.

Klíčová slova mobilní aplikace, implementace, sledování polohy, lokalizace, Android, GPS, SMS

Abstract

This bachelor thesis describes development of an Android application for remote localization of mobile devices. At first, I write about Android system and principles of getting location on mobile devices, then I compare some of existing, similar solutions. The main task of the thesis is a description of the entire development process, from analysis, design and implementation to testing and publication of the completed application. Output is an easy-to-use application for finding a lost phone even without connecting to the Internet, mobile device could also be used as a simple GPS car alarm with this app. Application is available at Google Play Store, its source code is published on GitHub.

Keywords mobile application, implementation, location tracking, localization, Android, GPS, SMS

Obsah

Úvod **1**

1 Cíl práce **3**

2 Operační systém Android **5**
2.1 Vývoj a historie 5
2.2 Běhové prostředí 5
2.3 Proces kompilace aplikací 7

3 Lokalizace zařízení **9**
3.1 Global Positioning System 9
3.2 Triangulace mobilních vysílačů 11
3.3 Triangulace WiFi sítí 11

4 Analýza existujících řešení **13**
4.1 Google Find My Device (Android Device Manager) 13
4.2 Cerberus 15
4.3 Avast Antivir a ochrana mobilu 17
4.4 Where's My Droid 19
4.5 TrackLoc – SMS Phone Tracker 21
4.6 Porovnání analyzovaných řešení 23

5 Analýza a návrh **25**
5.1 Analýza požadavků 25
5.2 Případy užití 28
5.3 Doménový model 28
5.4 Komunikace se zařízením na dálku 30
5.5 Aktivní sledování polohy 34
5.6 Návrh uživatelského rozhraní 34

5.7 Architektura aplikace 39

6 Implementace **43**
6.1 Použité nástroje a technologie 43
6.2 Použité knihovny 45
6.3 Uživatelské rozhraní 48
6.4 Zjišťování polohy 49
6.5 SMS komunikace 50
6.6 Oprávnění 52

7 Testování **53**
7.1 Testování programátorem 53
7.2 Uživatelské testování použitelnosti 54

8 Vydání aplikace a zhodnocení **59**
8.1 Google Play Beta 59
8.2 GitHub 59
8.3 Zhodnocení a porovnání s ostatními nástroji 59
8.4 Možnosti rozšíření do budoucna 60

Závěr **61**

Literatura **63**

A Seznam použitých zkratek **67**

B Instalační příručka **69**
B.1 Instalace z Google Play 69
B.2 Instalace z přiloženého souboru 69

C Finální podoba aplikace **71**

D Obsah přiloženého CD **75**

Seznam obrázků

2.1 HTC Dream, první veřejně dostupný Android telefon [2] 6
2.2 Proces kompilace nativní Android aplikace [9] 7

3.1 Určení polohy pomocí GPS 10

4.1 Google Find My Device 14
4.2 Cerberus 16
4.3 Avast Antivir a ochrana mobilu 18
4.4 Where's My Droid 20
4.5 TrackLoc 22

5.1 Případy užití aplikace 27
5.2 Doménový model 29
5.3 Princip SMS komunikace mezi zařízeními 30
5.4 Princip získávání informací o poloze 32
5.5 Stavový diagram aktivního sledování polohy 33
5.6 Diagram aktivit popisující proces zahájení sledování polohy 33
5.7 Wireframe hlavního menu 35
5.8 Wireframe obrazovky „Sledování polohy“ 36
5.9 Wireframe obrazovky „Sledování polohy“ v orientaci na šířku . . . 36
5.10 Wireframe rozhraní pro vyhledání zařízení 37
5.11 Wireframe rozhraní pro konfiguraci SMS 38
5.12 Wireframe sekce s nápovědou 39
5.13 Wireframe znázorňující režim tabletu 40
5.14 Diagram znázorňující MVVM architekturu [24] 40
5.15 MVVM architektura použitá v aplikaci 41

6.1 Komponenta `BottomNavigationView` 45
6.2 Mapová komponenta z Google Play Services 45

C.1 Hlavní menu a konfigurace sledování polohy 71

C.2 Obrazovka vyhledání zařízení a SMS konfigurace 72
C.3 Layout sledování polohy v orientaci na šířku a režim tabletu 73
C.4 Nápověda sekce sledování polohy 74

Seznam tabulek

4.1 Srovnání analyzovaných aplikací 23

6.1 Přehled použitých virtuálních zařízení Android Emulator 44

7.1 Přehled zařízení použitých k testování při vývoji 53
7.2 Odpovědi uživatelů na otázky před uživatelským testem 55

Úvod

Mobilní zařízení zažívají během poslední doby veliký rozmach. Jsme svědky nárůstu prodejů smartphonů, tabletů a dalších „chytrých“ zařízení během posledních několika let. Jelikož jejich cena není nízká, nastává potřeba je před případnou ztrátou nebo krádeží adekvátně chránit. Výbava současných smartphonů umožňuje nalezení jejich polohy i na dálku, a to vše díky připojení do sítě nebo s využitím GPS modulu.

Rozhodl jsem se proto vytvořit aplikaci Location Tracker pro OS Android, která bude umožňovat sledovat polohu zařízení a automaticky informovat uživatele, pokud se poloha daného zařízení změní. Výstup práce je přínosem pro všechny uživatele hledající snadno použitelný nástroj, který jim pomůže v případě nouze jejich ztracený telefon najít.

Téma této bakalářské práce jsem si zvolil především proto, že jsem nenašel dostupnou aplikaci, která by zcela splňovala moji představu o tom, jak by taková aplikace měla fungovat a vypadat. Rozhodl jsem se proto vytvořit vlastní řešení, které bude řešit některé nedostatky stávajících aplikací a zároveň bude co nejjednodušší na používání.

V práci probírám celý proces vzniku mobilní aplikace, od sběru požadavků, analýzy, přes samotnou implementaci a testování, až po zveřejnění výsledného produktu. Prostor je věnován také operačnímu systému Android, pro který je aplikace určena, možnostem lokalizace chytrých zařízení a porovnání s již existujícími aplikacemi.

Kapitola 1

Cíl práce

Cílem této práce je návrh, implementace, otestování a nasazení mobilní aplikace pro OS Android sloužící ke vzdálenému sledování polohy mobilního zařízení. V první kapitole se zabývám operačním systémem Android, pro který je aplikace vyvíjena. Zaměřuji se zde na historii systému a proces kompilace a běhu aplikací. Cílem druhé části je vysvětlit způsoby lokalizace mobilních zařízení a popsat jejich výhody a nevýhody. Teoretickou část práce zakončuji analýzou podobných, již existujích řešení.

Praktická část práce se zabývá celým procesem vývoje softwarového produktu, od sběru požadavků až po zveřejnění výsledné aplikace do Google Play Store. Důraz je kladen především na analýzu a návrh, popis implementace a závěrečné testování.

Výstupem je funkční mobilní aplikace Location Tracker, která je dostupná ke stažení v obchodě Google Play.

Kapitola 2

Operační systém Android

2.1 Vývoj a historie

Historie operačního systému Android sahá až do roku 2003, kdy byla ve Spojených státech amerických, v kalifornském Palo Altu, založena společnost Android Inc. Jejími zakladateli byli Andy Rubin, Rich Miner, Nick Sears a Chris White. Andy Rubin tehdy prozradil, že nově založená firma plánuje vyvíjet „chytřejší mobilní zařízení, která budou více zohledňovat polohu a priority jejích vlastníků“. Důležitý krok přišel o dva roky později, v roce 2005, kdy firmu koupil Google. Ve vývoji se pokračovalo i nadále a 22. 10. 2008 byl představen první mobilní telefon postavený na OS Android – HTC Dream, známější pod názvem T-Mobile G1 (viz obrázek 2.1). Vybaven byl hardwarovou QWERTY klávesnicí, třípalcovým dotykovým displejem, ARM procesorem Qualcomm a systémem Android ve verzi 1.0. Od dubna 2009, kdy byla vydána verze Android 1.5 Cupcake, je každá nová majoritní verze systému pojmenována po sladkostech. [1]

V současnosti je Android součástí přibližně 74% všech mobilních zařízení, jedná se tedy o nejrozšířenější mobilní operační systém [3]. Dne 17. května 2017 Google oznámil během své konference Google I/O, že Android pohání po celém světě více než dvě miliardy aktivních zařízení [4]. Tento počet každým dnem narůstá a dnes už se s tímto systémem setkáváme nejen v mobilních telefonech a tabletech, ale také v televizích, chytrých hodinkách, automobilech a mnoha dalších zařízeních.

2.2 Běhové prostředí

Systém Android je postaven na upraveném linuxovém jádře, aplikace pro něj jsou psány v programovacím jazyce Java. V poslední době se začíná značně prosazovat také jazyk Kotlin. Ani jeden z těchto jazyků se nekompiluje do nativního kódu pro cílový procesor, ale do tzv. Java bytecode. Programy napsané

v Javě se typicky kompilují do `.class` souborů, které obsahují již zmíněný Java bytecode. Každá podporovaná platforma má svoji implementaci Java VM, která umožňuje zpracovat univerzální `.class` soubory do strojového kódu, kterému cílový procesor „rozumí“. Hlavní výhodou tohoto přístupu je fakt, že aplikace napsané v jazyce Java je možné spouštět na mnoha platformách bez nutnosti úpravy kódu.

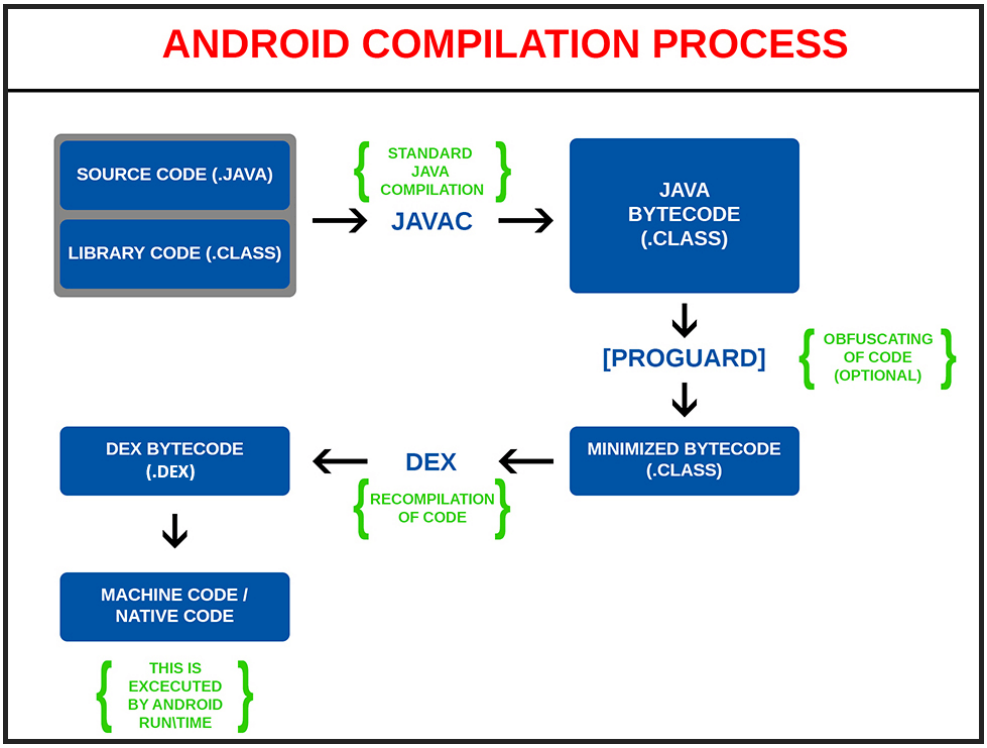
Pro účely Androidu byl však vyvinut vlastní VM – Dalvik, za kterým stojí vývojář Dan Bornstein. Ten narozdíl od standardních implementací Java VM na vstupu vyžaduje `.dex` (Dalvik Executable) soubory, které obsahují bytecode specifický pro systém Android. Takové aplikace se pak dají spouštět pouze na zařízeních s tímto systémem. Dalvik se používá pro běh aplikací od prvních verzí Androidu, ve verzi 2.2 Froyo pak přinesl funkci JIT (Just-In-Time) kompilace. Tato technika umožňuje překlad aplikace do strojového kódu během jejího běhu, což přineslo lepší výkon aplikací zároveň se snížení nároků na baterii [5]. Dalvik byl součástí systému Android až do verze 4.4 KitKat, ve verzi 5.0 Lollipop ho nahradil ART (Android Runtime). Ten byl již součástí Androidu 4.4, bylo ale nutné vynutit jeho používání ručně ve vývojářském nastavení.

Android Runtime přinesl další výrazné zlepšení, mezi které patří především efektivnější garbage collector, pokročilé možnosti při ladění chyb (debugging) a vyšší výkon aplikací. Narozdíl od Dalviku využívá AOT (Ahead-Of-Time) kompilace. Při použití této techniky se aplikace zkompilují jednorázově při jejich instalaci, nikoli při každém jejich spuštění. Výhodou tohoto přístupu je rychlejší běh a start samotných aplikací, nevýhodou je prodloužení doby jejich instalace. Android 7.0 Nougat v tomto ohledu přinesl další změnu, k AOT přibyla opět JIT kompilace. Tento přístup se využívá až do současnosti. [6] [7]

Obrázek 2.1: HTC Dream, první veřejně dostupný Android telefon [2]

2.3 Proces kompilace aplikací

Kompilace aplikací začíná transformací Java kódu psaného programátorem pomocí překladače `javac` (případně Kotlin kódu pomocí `kotlinc`). Výsledný `.class` soubor obsahující Java bytecode je poté volitelně zpracován nástroji jako je např. Proguard nebo R8, které slouží k obfuskaci a minifikaci kódu. Java bytecode je následně převeden do Dalvik Executable bytecode pomocí DX nebo D8, výsledkem je `.dex` soubor. Posledním krokem je zabalení zkompilovaného kódu do společně s dalšími nezbytnými soubory, mezi které mohou patřit soubory s grafikou aplikace, textovými řetezci a dalším obsahem. Výsledný APK balíček se nakonec digitálně podepíše a je připraven pro distribuci k uživatelům. Tento proces je díky nástrojům, jako je např. Gradle [8], automatizován a vývojář tak jednotlivé mezikroky pouze konfiguruje. Celý proces je znázorněn na obrázku 2.2.



Obrázek 2.2: Proces kompilace nativní Android aplikace [9]

Kapitola 3

Lokalizace zařízení

V této kapitole se zaměřuji na technologie, pomocí kterých je možné určovat polohu mobilních zařízení. Zabývám se zde principy jejich fungování, jednotlivými parametry, jako je například přesnost zaměření a možnostmi využití těchto technologií v realizované aplikaci.

3.1 Global Positioning System

GPS je zkratka pro Global Positioning System – pravděpodobně nejznámější globální družicový polohový systém, který po celém světě využívají automobilové navigace, lodní navigace, mobilní zařízení, nebo třeba sledovací systémy. V aplikaci je využíván jako hlavní zdroj dat o poloze zařízení, především kvůli vysoké přesnosti.

3.1.1 Historie

Počátek systému GPS sahá až do 60. let 20. století, kdy námořnictvo Spojených států amerických provádělo experiment, jehož cílem bylo sledovat polohu amerických ponorek nesoucích jaderné nálože. S využitím šesti satelitů obíhajících okolo zemských pólů a „Dopplerova jevu“ při příjmu satelitního signálu byly americké ponorky schopné určit svoji polohu během několika minut. [10]

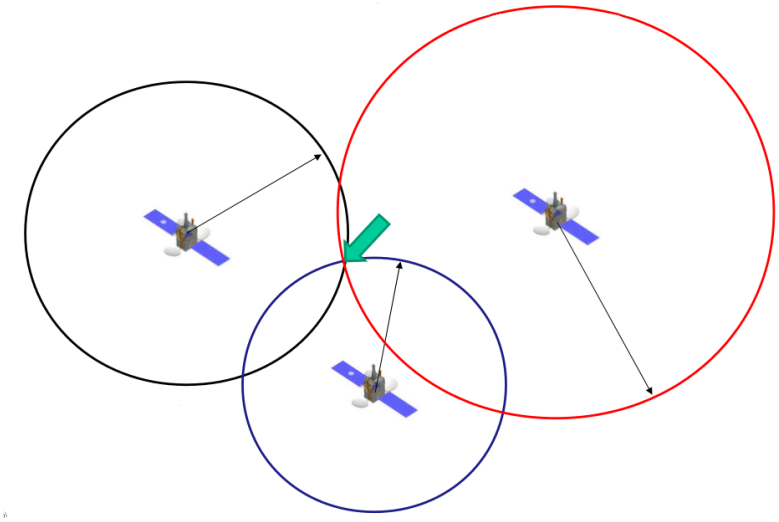
V roce 1973 vzneslo ministerstvo obrany Spojených států požadavek na vývoj stabilního a spolehlivého navigačního systému, který dostal nejprve jméno NAVSTAR. Technologie byla původně určená pouze vojenskému personálu k usnadnění určení jejich polohy. O pět let později byly na oběžnou dráhu vyslány první čtyři satelity systému GPS. Až v roce 1995 byl systém uveden do plného provozu, ke kterému potřebuje alespoň 24 družic. [11]

Systém existoval zpočátku pouze ve verzi PPS (Precise Positioning Service), která dosahuje maximální přesnosti a je proto využívána americkým ministerstvem obrany. Varianta SPS (Standard Positioning Service) byla spuštěna až o osm let později a je dostupná civilnímu obyvatelstvu. Civilní vari-

anta původně obsahovala úmyslnou chybu, která snižovala přesnost zaměření na cca 100 metrů pomocí mírného posunutí hodin v družicích. V roce 2000 došlo k odstranění chyby, což vedlo přibližně k pětinásobně přesnější lokalizaci. Dnes je systém celosvětově využíván a stal se synonymem pro navigační systém, ačkoliv se v posledních letech pracuje na dalších systémech, jako je například evropský projekt Galileo, ruský GLONASS nebo čínský Beidou. [11]

3.1.2 Princip fungování

Standarní varianta (SPS) systému GPS pomáhá určit polohu s přesností na cca 8 metrů. Každý ze satelitů vysílá signál pro přijímače, které jsou schopny zjistit svoji polohu pomocí výpočtu rozdílu času odeslání signálu ze satelitu a času příjmu. Družice nesou atomové hodiny, které poskytují velice přesný čas, což je nutné k zajistění vysoké přesnosti celého systému. Aby byl přijímač schopný vypočítat svoji aktuální polohu včetně výšky, potřebuje přijímat signál alespoň od tří satelitů najednou, za předpokladu, že tento přijímač disponuje vlastními atomovými hodinami, viz ilustrace 3.1. V případě dostupnosti signálu alespoň od čtyř družic požadavek na vlastní atomové hodiny odpadá a přijímač má v tu chvíli všechny potřebné informace k výpočtu zeměpisné šířky a délky, nadmořské výšky a času. [12]



Obrázek 3.1: Ilustrace principu určení polohy pomocí GPS. Přijímač se nachází v průsečíku všech tří kružnic, které znázorňují vzdálenost přijímače od jednotlivých satelitů. [13]

3.2 Triangulace mobilních vysílačů

Další metodou, pomocí které mobilní zařízení mohou získávat údaje o svojí zeměpisné poloze, je takzvaná triangulace nejbližších BTS. Jako BTS (Base Transceiver Station) se označují mobilní vysílače poskytující pokrytí mobilním signálem, který je využíván pro telefonní komunikaci. Mobilní telefon nebo jiné zařízení připojené do GSM sítě zná identifikátory vysílacích stanic, ke kterým je připojeno. Tyto identifikátory zařízení poté zašle prostřednictvím Internetu geolokační službě, která má k dispozici databázi s umístěním jednotlivých vysílačů a díky tomu může vypočítat přibližnou polohu zařízení, kterou následně zašle zpět.

Přesnost této techniky značně závisí na hustotě okolních vysílačů. Ve městech, kde je typicky hustota vysílačů vyšší, se přesnost pohybuje v řádech desítek až stovek metrů, v méně osídlených oblastech může v některých případech dosahovat i jednotek kilometrů.

Výhodou určení polohy pomocí BTS je nižší energetická náročnost oproti systému GPS, používá se často také v budovách, kde je satelitní GPS signál obvykle nedostupný. Na Android zařízeních se tato technika často kombinuje s dalšími zpřesňujícími daty, jako je například pozice okolních WiFi sítí. Výsledkem je pak vyšší přesnost určené polohy. Nevýhodou je nutnost připojení k Internetu, bez kterého není možné telefon tímto způsobem lokalizovat a již zmíněná nižší přesnost, především v oblastech s malou hustotou BTS.

3.3 Triangulace WiFi sítí

Triangulace bezdrátových WiFi sítí funguje na velice podobném principu jako lokalizace pomocí BTS, popisovaná v předchozí části. Zařízení se systémem Android sbírají data o okolních WiFi sítích a odesílají je na servery společnosti Google, pokud má uživatel povolené Google polohové služby [14]. Tato data pak pomáhají ostatním uživatelům Google služeb ke správnému nalezení jejich polohy.

Tato technika poskytuje oproti triangulaci mobilních vysílačů přesnější výsledky, především v oblastech s vyšším počtem okolních WiFi sítí. Mezi její výhody patří také dostupnost v budovách a nižší náročnost na baterii oproti lokalizaci pomocí GPS.

Kapitola 4

Analýza existujících řešení

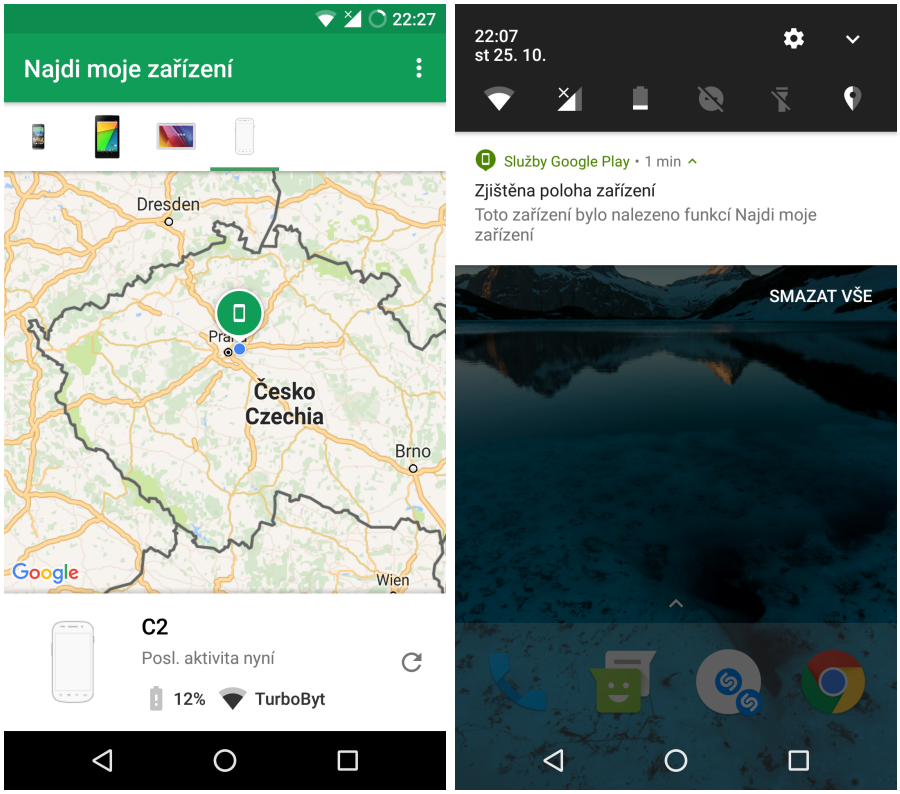
Práci jsem začal hledáním již existujících Android aplikací, které by splňovaly moje požadavky. V práci se zabývám pouze systémem Android, veškeré aplikace byly získány prostřednictvím Google Play Store. Jiné systémy mohou poskytovat pokročilé nástroje pro ochranu a nalezení zařízení již v základu (např. Apple Find My iPhone).

4.1 Google Find My Device (Android Device Manager)

Autor: Google LLC
Počet instalací: 10 000 000 – 50 000 000
Analyzovaná verze: 2.1.005 (1. 9. 2017)

Google Find My Device [15] je služba sloužící k nalezení ztraceného nebo ukradeného telefonu. Dříve byla také známá pod názvem Android Device Manager. Google ji ale v květnu 2017 přejmenoval a začlenil do Google Play Protect, balíku sdružujícího bezpečnostní služby zaměřené na ochranu jednotlivých zařízení. Aplikace je v současnosti integrována do systému Android a pokud je uživatel v zařízení přihlášený Google účtem, je automaticky aktivována.

Služba Find My Device se skládá ze tří částí. První z nich je samotná lokalizační služba, která je schopná v případě potřeby ohlásit svoji polohu nebo umožnit ovládání zařízení na dálku. Tato část se stává aktivní po spárování telefonu s Google účtem, od uživatele nejsou vyžadovány žádné speciální kroky navíc. Uživatel tak často ani nemusí vědět, že je jeho zařízení takto zabezpečeno a hlídáno. Jestliže dojde k úspěšné lokalizaci, na vyhledáváném zařízení se zobrazí informační notifikace. V případě, že nechceme službu používat, je možná její deaktivace v uživatelském nastavení v sekci Zabezpečení > Správa zařízení. Toto umístění se může na některých zařízeních lišit.



Obrázek 4.1: Android aplikace Google Find My Device, lokalizační notifikace

Druhá část je realizována formou webového rozhraní dostupného na adrese `https://www.google.com/android/find`. Zde se po přihlášení Google účtem otevře přehledné ovládací centrum, pomocí něhož můžeme lokalizovat svoje telefony, tablety a další Android zařízení. Rozhraní je jednoduché na používání, obrazovce dominuje mapa a v levé části se nachází seznam všech přístrojů, na kterých je daný Google účet používán. Po výběru z nabídky se webový klient pokusí kontaktovat běžící službu na konkrétním zařízení a zjistit jeho polohu. Pokud se operace zdaří, na mapě se ihned zvýrazní poslední známá poloha a v levém panelu uvidíme stav baterie v procentech a případný název (SSID) WiFi sítě, ke které je vybrané zařízení v současnosti připojeno. Tím ale možnosti aplikace nekončí, nabízí také spuštění hlasitého zvukového alarmu, vzdálené uzamknutí zařízení a vymazání všech dat. Funkce uzamknutí umožňuje nastavit heslo, po jehož zadání se zařízení odblokuje a je zde také podpora pro zobrazení libovolného vzkazu a kontaktního telefonního čísla na obrazovce uzamčení. Pokud vybereme poslední možnost, tedy smazání všech dat, jsme upozorněni, že po tomto kroku již není možné telefon nadále vyhledat.

Třetí částí je Android aplikace distribuovaná přes Google Play Store, která ovšem neslouží k samotnému zabezpečení telefonu, ale podobně jako přes

webový klient pomocí ní můžeme lokalizovat jiné telefony. Hodí se tedy především v případech, kdy nemáme přístup k počítači ale potřebujeme ztracený přístroj lokalizovat co nejdříve. Funkčnost je stejná jako v případě webového klienta, aplikace je ale lépe přizpůsobená pro mobilní zařízení než webové rozhraní.

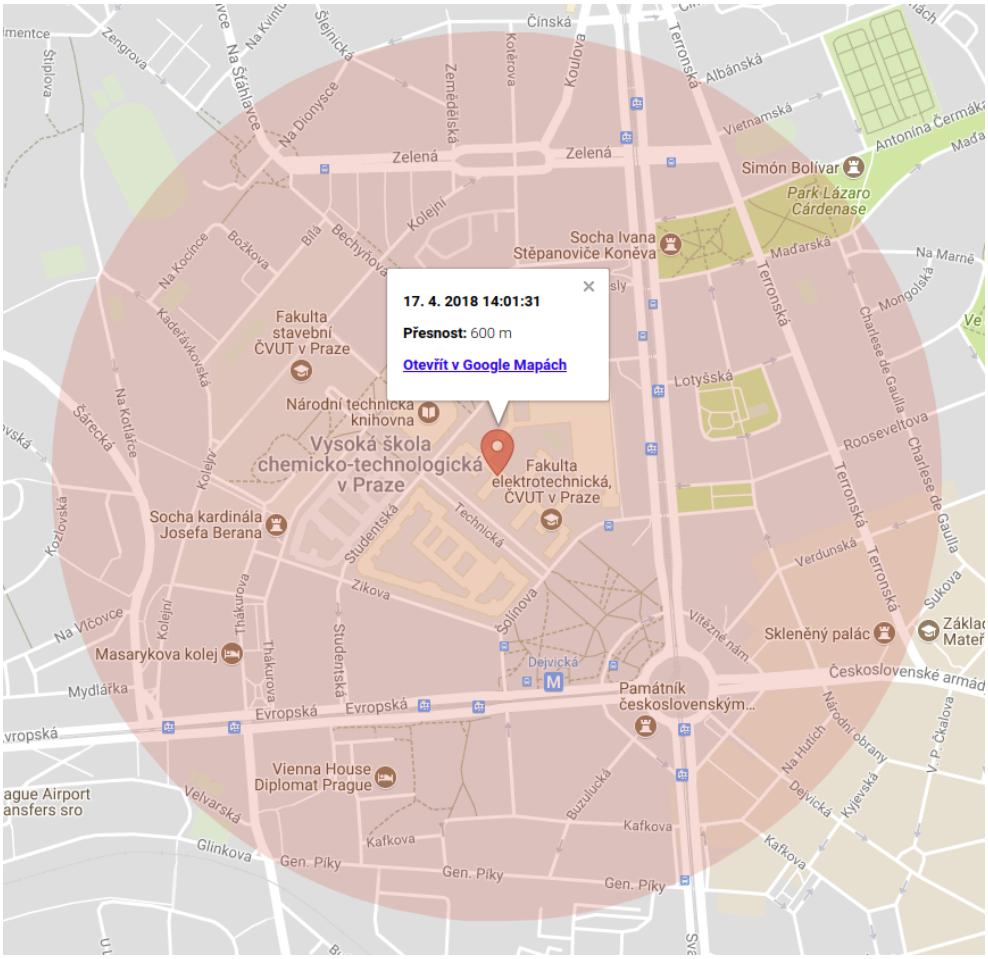
Zhodnocení Find My Device je zdařilá služba, která má výhodu především v tom, že je integrována přímo v Androidu bez nutnosti manuální instalace, konfigurace a vytváření dalších uživatelských účtů. Může tak pomoci lidem, kteří na zabezpečení svého zařízení nemysleli předem a nyní se ho snaží získat zpět. Uživatelské rozhraní je přívětivé, zejména pozitivně hodnotím webové rozhraní, které je přehledné a snadno ovladatelné. V porovnání s konkurencí se však jedná o jednoduchý nástroj s téměř nulovou možností konfigurace a chybí zde pokročilejší funkce, jako např. pořízení snímku z fotoaparátu nebo výpis hovorů a SMS zpráv. Informační notifikaci o vyhledání telefonu považuji za zbytečnou. Případného zloděje může upozornit na to, že je telefon vyhledáván a ten tak může zařízení odpojit od Internetu, od mobilní sítě nebo ho rovnou resetovat do továrního nastavení, což znemožní jeho další lokalizaci. Jako zásadní nevýhodu považuji fakt, že přístroj lze lokalizovat pouze pomocí internetového připojení, bez kterého spojení nefunguje. Pokročilejší konkurence umožňuje ovládání pomocí SMS zpráv, k úspěšnému určení polohy tak stačí, aby byl vyhledávaný telefon v místě s dostupným GSM signálem, což v některých případech může značně ovlivnit, zda se zařízení podaří úspěšně nalézt či nikoliv. Kdo však pokročilé funkce nevyžaduje, ten bude pravděpodobně s touto službou spokojen, svůj účel plní dobře.

4.2 Cerberus

Autor: LSDroid
Počet instalací: 1 000 000 – 5 000 000
Analyzovaná verze: 3.5.2 (11. 10. 2017)

Cerberus [16] se řadí mezi komplexní nástroje na vzdálenou lokalizaci telefonu na platformě Android. Aplikace je k dispozici ke stažení z Google Play Store a je stále aktivně vyvíjena a aktualizována. Po instalaci má uživatel týden na bezplatné otestování, pokud je spokojen a chce Cerberus využívat i nadále, musí si zakoupit licenci formou ročního předplatného.

Služba je rozdělená na dvě části. První částí je Android aplikace, která zajišťuje zabezpečení zařízení, na kterém je nainstalována. Při prvním spuštění jsme požádáni o přihlášení k Cerberus účtu, pokud ho ještě nemáme, je možné účet založit. Předností aplikace je bohaté nastavení, můžeme nastavit heslo vyžadované při každém spuštění, SMS heslo požadované pro ovládání pomocí zpráv, oprávnění pro smazání obsahu úložiště a mnoho dalšího. Cer-



Obrázek 4.2: Ukázka mapy s nalezeným zařízením z webového rozhraní Cerberus

berus disponuje pokročilými funkcemi, mezi které patří například automatické pořízení fotografie přední kamerou pokud uživatel zadá špatné heslo pro odemčení, možnost nastavit zasílání upozornění (e-mail, SMS) pokud je vyměněna SIM karta nebo vlastní AutoTask systém, pomocí kterého jdou nastavit akce spouštěné při zaznamenání nějaké události (např. připojení na WiFi síť).

Pokud jsme zařízení ztratili, budeme pro lokalizaci používat webové rozhraní na adrese `https://www.cerberusapp.com/dashboard`. To je uspořádáním vzhledově podobné webové části aplikace Find My Device. Nachází se zde mapa, ovládací panel a informační panel na vrchní straně. Mapa zobrazuje polohu zařízení, informační panel nás informuje o stavu sledování, úrovni nabití baterie a obsahuje indikátor připojení. Zařízení ovládáme pomocí příkazů, které posíláme z ovládacího panelu. Příkazů je mnoho, mezi nejzajímavější

patří například pořizování fotografií a videa, možnost spustit libovolnou aplikaci nebo schopnost skrýt Cerberus v seznamu aplikací. Jeden z příkazů nám umožňuje získat historii polohy, nejsme tedy omezeni pouze na aktuální lokaci, ale můžeme snadno zjistit, kde se zařízení nacházelo v minulosti. Pokud máme s Cerberus účtem spárováno více zařízení, můžeme mezi nimi snadno přepínat.

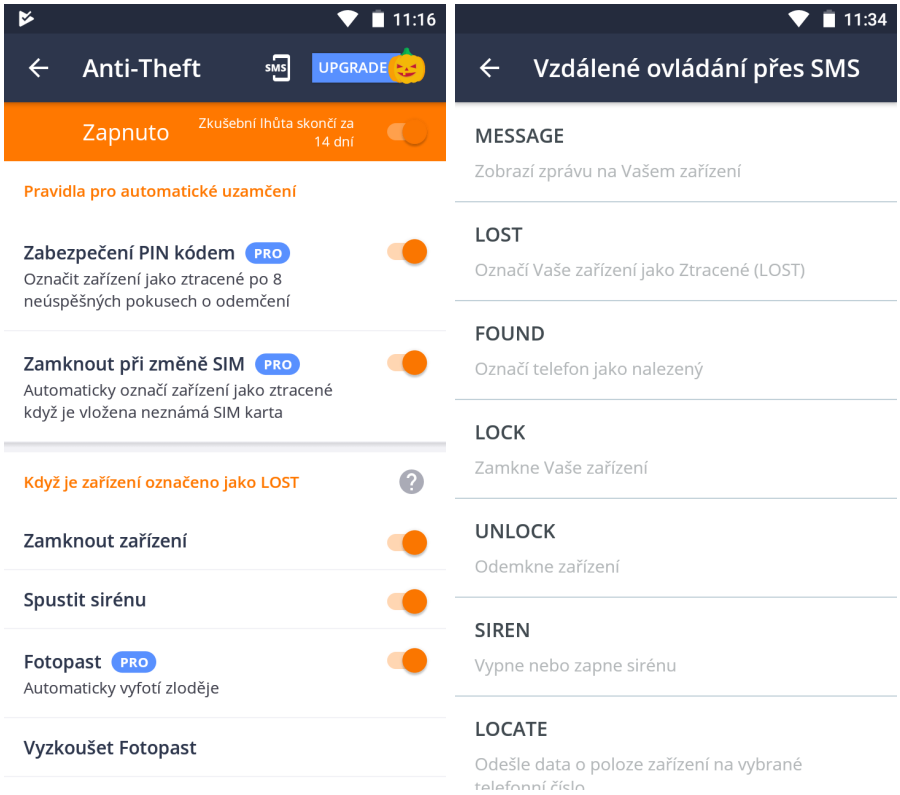
Zhodnocení Jedná se o jeden z nejpokročilejších nástrojů tohoto typu. Zejména oceňuji bohaté možnosti konfigurace a spoustu pokročilých funkcí, které můžou v kritické chvíli pomoci v hledání ztraceného zařízení. Významným plusem je možnost ovládání jak přes Internet, tak přes SMS zprávy. Uživatelské rozhraní je intuitivní a díky logickému členění nastavení není problém se v aplikaci orientovat. Aplikace je placená, což představuje asi největší nevýhodu, která může spoustu potenciálních uživatelů odradit.

4.3 Avast Antivir a ochrana mobilu

Autor: Avast Software – Antivirus and VPN
Počet instalací: 100 000 000 – 500 000 000
Analyzovaná verze: 6.7.1 (26. 10. 2017)

Avast Mobile Security [17] (český název Avast Antivir a ochrana mobilu) od tuzemských vývojářů Avast Software je řešení pro kompletní ochranu a zabezpečení mobilního zařízení. Jak už název napovídá, Avast zajišťuje zabezpečení nejen proti zcizení či ztrátě, jako některá konkurenční řešení, ale jeho součástí je i antivirus, firewall, blokátor hovorů a nepřeberné množství dalších bezpečnostních funkcí. Možnost zařízení lokalizovat na dálku je součástí tzv. Anti-Theft balíku, který je dostupný pouze v placené verzi aplikace.

Pokud jsme Avast čerstvě nainstalovali, musíme podporu Anti-Theft funkcionality nejdříve aktivovat. Aktivace se skládá ze tří kroků, nejdříve vybereme PIN kód, kterým budeme potvrzovat vstup do nastavení Anti-Theft a zároveň vybereme Google účet, který bude sloužit k obnově PIN kódu, pokud bychom ho zapomněli. V druhém, volitelném kroku můžeme založit Avast účet, pomocí kterého se budeme přihlašovat do webového rozhraní, sloužícího k ovládání telefonu na dálku. Alternativní možností je komunikace přes SMS zprávy, která je rovněž podporována. Poslední krok po nás požaduje povolení Android Správce zařízení, díky kterému aplikace dostane povolení k obnovení telefonu do továrního nastavení, sledování neúspěšných pokusů o odemčení zařízení a k dalším funkcím. Pokud aktivace proběhla v pořádku, dostaneme se do nastavení Anti-Theft modulu. Zde máme možnost konfigurovat velké množství dostupných funkcí. Nejzajímavější je možnost nastavit akce, které se provedou automaticky, jakmile označíme zařízení za ztracené (LOST). Ve výchozím nastavení se telefon automaticky uzamkne, spustí sirénu, aktivuje se tzv. Fo-



Obrázek 4.3: Anti-Theft modul Android aplikace Avast

topast a zaznamená se poslední známá poloha. Funkce pojmenovaná Fotopast automaticky pořídí snímek přední kamerou, který zašle na e-mail a webové rozhraní Avastu. Všechny tyto funkce je možné individuálně aktivovat nebo deaktivovat nezávisle na sobě.

Webové rozhraní My Avast, dostupné na adrese `https://my.avast.com` umožňuje po přihlášení příslušným Avast účtem zobrazit stav všech spárovaných přístrojů. V přehledu zařízení se zobrazuje souhrnné info o zařízení, mezi které patří čas poslední komunikace, verze aplikace Avast a operačního systému cílového zařízení, poslední známá IP adresa a stav baterie. Prostřednictvím ovládacího panelu můžeme zaslat jednotlivé příkazy (Vymazat data, Nahrát zvukový záznam, Vyfotit obrázek a Získat výpis kontaktů, hovorů a SMS ze zařízení), lokalizovat telefon, uzamknout ho, zapnout sirénu a nebo ho označit jako ztracený (LOST), což provede hromadně všechny akce, které byly nakonfigurovány v nastavení aplikace. Uživatelské rozhraní vzdálené lokalizace telefonu je podobné jako u konkurenčních služeb, jako mapové podklady využívá Google Maps, příjemným bonusem je i zobrazení fotografie nalezené lokality získané ze služby Google Street View.

Zhodnocení Avast Antivir a ochrana mobilu je komplexní balík bezpečnostních funkcí, které si kladou za cíl udržet zařízení a jeho data v bezpečí. Aplikace má povedené uživatelské rozhraní, je stabilní a dobře přizpůsobitelná. Během testování jsem nenarazil na žádný vážnější problém. V porovnání s ostatními analyzovanými aplikacemi Avast trochu zaostává co se týče rychlosti odezvy webového rozhraní, požadavek na lokalizaci bylo občas nutné poslat vícekrát i přes to, že telefon byl připojen na WiFi síť s rychlým připojením. Je ale možné, že tyto problémy mohly být způsobeny pouze momentálním vytížením serverů, ačkoli bylo testování prováděno opakovaně. Stejně jako konkurenční bezpečnostní balíky, i Avast si za pokročilé funkce, mezi které patří právě i Anti-Theft modul, nechá zaplatit. Řešení je vhodné pro zájemce o celkovou ochranu telefonu včetně antiviru, pokud ale uživatel hledá jednoduchý nástroj pro nalezení polohy ztraceného telefonu, je na trhu mnoho vhodnějších aplikací.

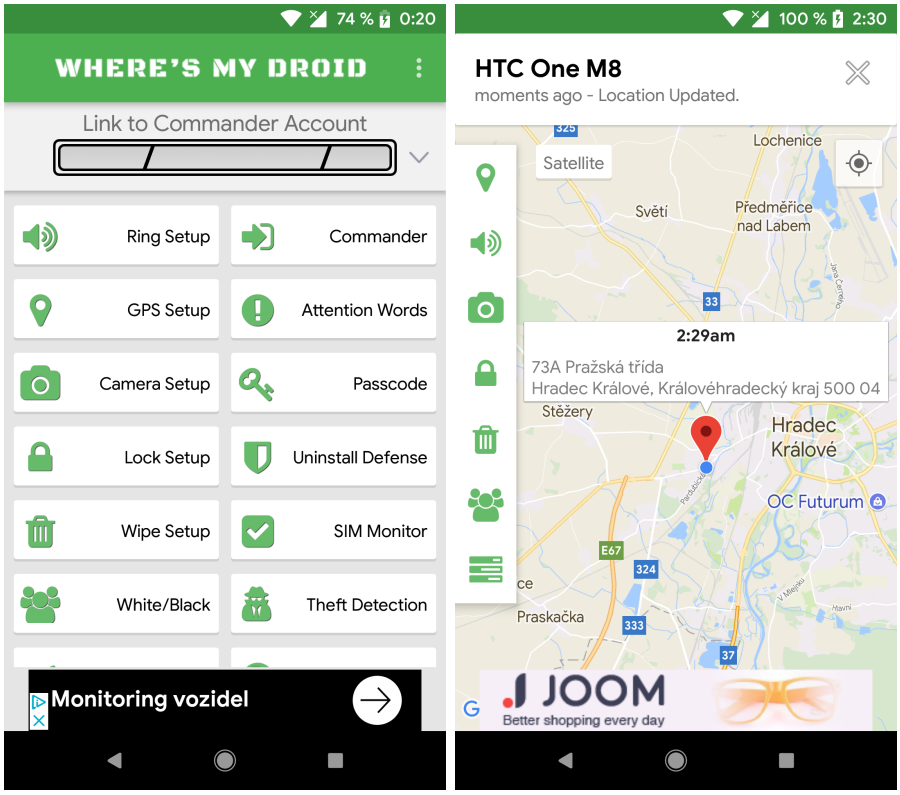
4.4 Where's My Droid

Autor: Alienman Technologies LLC
Počet instalací: 10 000 000 – 50 000 000
Analyzovaná verze: 6.3.0 (19. 1. 2018)

Where's My Droid [18] je jednou z prvních aplikací sloužících k nalezení a ovládání Android zařízení na dálku. K dnešnímu dni používá Where's My Droid více než 10 milionů lidí dle statistik počtu stažení z Google Play Store. Základní verze je dostupná ke stažení zcela zdarma, ale obsahuje reklamy a většina funkcí není dostupná. V případě zájmu o pokročilejší funkce je uživatel nucen využít placenou verzi, které jsou v tomto případě dvě – „Pro“ a „Elite“, která narozdíl od první jmenované odemyká všechny funkce. Rozdíl je i ve formě platby, zatímco verze „Pro“ stojí jednorázově 99,99 Kč, verze „Elite“ stojí 28,99 Kč měsíčně. Pro účely testování byla využita pouze neplacená verze.

Jakmile aplikaci úspěšně nainstalujeme a poprvé spustíme, uvítá nás průvodce počátečním nastavením. Nejdříve je nutné odsouhlasit licenční podmínky, poté jsme požádáni o přidělení základních oprávnění. Těch je vyžadováno celkem šest – přístup k poloze, SMS zprávám, kontaktům, telefonním hovorům, fotoaparátu a nakonec k úložisti souborů. Aplikace u každé skupiny oprávnění poskytuje krátké vysvětlení, proč tato oprávnění vyžaduje a jaké funkce díky nim bude poskytovat. Další krok nám umožňuje volitelné spojení s „Commanderem“ – rozhraním pro ovládání a lokalizaci zařízení. Následně jsme seznámeni s formou speciálních SMS zpráv, které mohou být použity na ovládání aplikace z jiného telefonu, případně pro získání informace o pozici cílového zařízení. Tím průvodce nastavením končí a my jsme konečně přesměrováni do hlavní nabídky.

Hlavní menu Where's My Droid je tvořeno množstvím tlačítek ve dvou



Obrázek 4.4: Uživatelské rozhraní aplikace Where's My Droid

sloupcích, které uživatele přesměrují do nastavení jednotlivých funkcí. Klíčová je především část s konfigurací GPS lokalizace, prostřednictvím které je možné přizpůsobovat jaké informace má telefon zasílat v případě, že je požadována jeho poloha. Kromě GPS souřadnic se ve výchozím nastavení zasílá také odkaz do Google Maps s nalezenou polohou nebo notifikace v případě, že telefon dosáhne nízké úrovně baterie. Aplikace podporuje ve verzi zdarma ještě funkce spuštění hlasitého alarmu, uzamknutí aplikace pomocí hesla a zasílání oznámení o výměně SIM karty. Veškeré další funkce, jmenovitě například zasílání fotografií, uzamknutí zařízení, vymazání všech dat nebo automatické hlídání pomocí GPS (Geofence), jsou dostupné jen po zakoupení placené verze.

Část aplikace sloužící k nalezení a ovládání zařízení je nazvána „Commander" a je dostupná na adrese `https://commander.wheresmydroid.com/`, případně přímo jako součást mobilní aplikace. Příliš se neliší od konkurenčních řešení – po přihlášení se dostaneme do ovládacího panelu, kde vidíme poslední známou pozici zařízení na mapě včetně přesnosti získané polohy a máme možnost zařízení ovládat. Kromě vyhledání zařízení je podporováno zapnutí hlasitého alarmu, pořízení fotografie, uzamknutí zařízení, vymazání dat, získání kontaktů a výpisu hovorů.

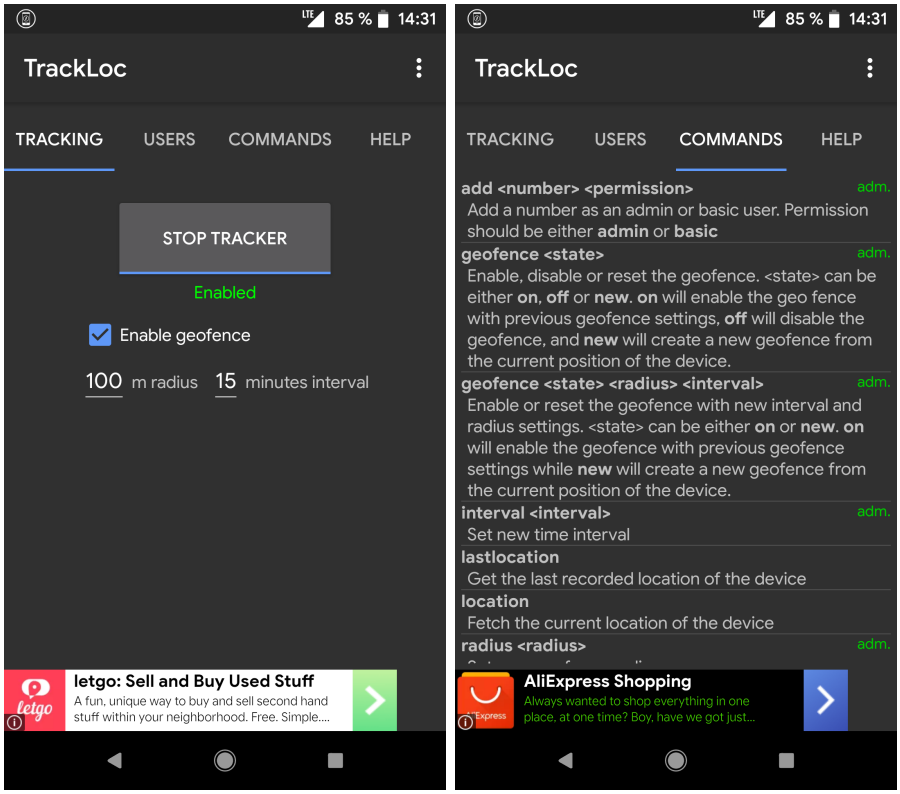
Zhodnocení Aplikace Where's My Droid po celou dobu testování fungovala bez problému, SMS odpovědi s polohou fungovaly spolehlivě, líbila se mi taktéž možnost nastavit zasílaná data. V porovnání s ostatními je bohužel v celkovém dojmu nejslabší. Jednak její uživatelské rozhraní, ač celkem přehledné, nevypadá příliš hezky a je na něm znát jeho stáří. Nelíbilo se mi také, že nejsou na první pohled oddělené funkce, které mi byly, jako uživateli neplacené verze, přístupné a ty, které byly zamčené. Nejvíce ovšem Where's My Droid v mých očích sráží forma reklam a propagace placených verzí. Při přechodu mezi nabídkou a nastavením jednotlivých funkcí se často zobrazila reklama přes celou obrazovku nebo nabídka placených verzí. Ty jsou navíc podle mého názoru drahé, existuje množství podobných řešení, které nabízejí za stejnou nebo i nižší cenu více možností. Celkově bych aplikaci nedoporučil, dnes již existuje spousta lepších alternativ.

4.5 TrackLoc – SMS Phone Tracker

Autor: Henrik Simonsen
Počet instalací: 10 000 – 50 000
Analyzovaná verze: 1.1 (1. 10. 2017)

Na nástroj TrackLoc [19] jsem narazil během hledání aplikace, která by umožňovala automatické hlášení pomocí SMS v případě, že dané zařízení opustí předem definovaný prostor (geofencing) a zároveň byla dostupná zdarma. Jelikož přesně to sliboval v popisu na Google Play Store TrackLoc, rozhodl jsem se ho vyzkoušet. V porovnání s předešlými aplikacemi se jedná o méně známou aplikaci (10 000+ stažení), stojí za ní norský vývojář Henrik Simonsen. Existuje i placená verze „TrackLoc Pro“, která stojí 84,99 Kč. Testoval jsem pouze bezplatnou variantu.

Po prvním spuštění jsme vyzváni k zadání telefonního čísla, na které budou zasílána upozornění o změně polohy. Rozhraní aplikace je velmi strohé a jednoduché, řešeno je pomocí jedné obrazovky o čtyřech záložkách. Výchozí záložka obsahuje nastavení samotného sledování polohy. Můžeme konfigurovat pouze vzdálenost od současné polohy, po jejímž překročení se odešle upozornění, jak často bude poloha hlídána a případně funkci geofencing můžeme zcela vypnout. V takovém případě pak aplikace pouze reaguje na příchozí SMS zprávy. Po zapnutí sledování polohy se několik desítek vteřin nic neděje, poté začne aplikace zaměřovat současnou polohu, kterou následně nastaví jako střed hlídané plochy. Na další záložce se nachází přehled všech telefonních čísel, na které jsou zasílány zprávy a od kterých je naopak povoleno zprávy s příkazy přijímat. Jednotlivá čísla můžeme nastavit s právy „admin“, od kterých je povoleno přijímat všechny zprávy, případná další telefonní čísla s nastavením „basic user“ mohou využívat pouze menší část SMS příkazů. Třetí záložka obsahuje přehled všech SMS příkazů, na které TrackLoc reaguje. Podporováno



Obrázek 4.5: Uživatelské rozhraní aplikace TrackLoc

je samozřejmě zasílání aktuální nebo poslední známé polohy, změna poloměru sledování nebo ukončení sledování polohy. Na poslední záložce je nápověda, která detailně vysvětluje funkce a fungování aplikace.

Vývojář poskytuje i zpoplatněnou variantu, která oproti bezplatné neobsahuje reklamy a nabízí více funkcí. Mezi ně patří zasílání upozornění nejen pomocí SMS, ale navíc i prostřednictvím e-mailu, notifikací o nízkém stavu nabití baterie, ovládání pomocí „prozvonění“ aj.

Zhodnocení Tato aplikace se ze všech testovaných nejvíce přiblížila mým představám a požadavkům. Je velice malá (po nainstalování na testovací zařízení zabírala pouze 1,96 MB), svižná a především poskytuje funkci geofencing i v neplacené verzi. Design není příliš povedený, rozhraní je pouze v odstínech šedé, některé komponenty jsou nelogicky zarovnané a jako celek působí nedodělaně. Hlavním nedostatkem je ale nemožnost ručního nastavení rozsahu na mapě, ve kterém se zařízení může pohybovat bez zaslání upozornění. Zaměření polohy se navíc spustí až po dlouhé prodlevě. Uživatel tak nemá dobrý přehled o tom, jak velký pohyb bude stačit k odeslání varování. Naopak se mi líbí možnosti konfigurace telefonních čísel, přehled o příkazech a nápověda, která

je dostupná offline přímo v aplikaci. Možnosti nastavení jsou jinak velmi omezené a mínusem je také reklamní proužek a zasílání informací o baterii pouze v placené verzi. TrackLoc se mi jinak líbil, sráží ho ale nedodělky v uživatelském rozhraní a omezené možnosti, především v neplacené verzi.

4.6 Porovnání analyzovaných řešení

V tabulce 4.1 je uvedeno porovnání klíčových parametrů jednotlivých aplikací.

Tabulka 4.1: Srovnání analyzovaných aplikací

	Find My Phone	Cerberus	Avast	Where's My Droid	TrackLoc
SMS ovládání	ne	ano	ano	ano	ano
web klient	ano	ano	ano	ano	ne
geofencing	ne	ne	ne	ne	ano
reklamy	ne	ne	ano	ano	ano
cena	zdarma	43€*	229,99Kč* 54,99Kč**	99,99Kč 28,99Kč**	zdarma 84,99Kč

* cena za roční předplatné

** cena za měsíční předplatné

Kapitola 5

Analýza a návrh

Důkladná analýza a návrh je součástí vývojového procesu každého softwaru, nejinak tomu bylo i v případě tvorby této aplikace. Tato část vývoje je nesmírně důležitá, ovliňuje totiž výslednou podobu programu a především díky pečlivému návrhu je možné vytvořit dobře fungující, stabilní a udržovatelnou aplikaci. V rámci analýzy zde uvádím především přehled funkčních a nefunkčních požadavků a definuji případy užití (use cases). Další části se věnují samotnému návrhu.

Veškeré diagramy použité v této části práce byly vytvořeny pomocí nástroje Enterprise Architect [20] od společnosti Sparx Systems.

5.1 Analýza požadavků

5.1.1 Funkční požadavky

F1. Automatické hlášení polohy při změně polohy Bude podporováno zasílání varovných SMS zpráv, jestliže zařízení opustí předem definovaný prostor. Uživateli bude umožněno konfigurovat daný prostor, tedy jeho umístění a velikost.

F2. Hlášení polohy na vyžádání uživatelem Aplikace bude reagovat na příchozí SMS požadavky. Jakmile obdrží požadavek na zaslání polohy, během co nejkratší doby zjistí svoji polohu a zašle ji odesílajícímu.

F3. Konfigurace dat v zasílaných zprávách Uživatel aplikace bude mít možnost nastavit, které informace obdrží zpět po požadavku na lokalizaci. GPS souřadnice budou povinný prvek, který se bude zasílat vždy. Mezi volitelné informace budou zařazeny tyto: adresa nalezeného umístění, čas určení polohy, přesnost polohy, způsob lokalizace (např. GPS), stav baterie, název připojené WiFi a IP adresa. Pokud je některý z těchto údajů neznámý, aplikace to oznámí.

F4. Seznam SMS příkazů Aplikace bude obsahovat přehled SMS příkazů s požadovaným formátem a vysvětlivkami.

F5. Rozhraní pro nalezení jiného zařízení Aplikace bude nabízet uživatelsky přívětivý způsob, jak nalézt jiné zařízení, které má tuto aplikaci taktéž nainstalovanou. Umožní zadat telefonní číslo hledaného zařízení a jeho polohu v případě úspěšné lokalizace zobrazí na mapě.

F6. Nastavení SMS hesla Aby se zabránilo zasílání polohy nepovolaným osobám, bude možné nastavit tzv. „SMS heslo“, které bude vyžadováno v příchozí zprávě. Pokud heslo bude chybět nebo bude nesprávné, nedojde k provedení požadovaného příkazu.

F7. Hlášení změn stavu baterie Bude realizováno zasílání SMS zpráv informujících uživatele o poklesu stavu nabití baterie nebo o změně stavu nabíjení.

F8. Ukončení sledování polohy při nízké baterii V případě, že baterie zařízení bude téměř vybitá, automatické sledování polohy bude ukončeno. Aplikace bude nadále reagovat na příchozí SMS požadavky. Tuto funkci bude možné uživatelem vypnout podle potřeby.

5.1.2 Nefunkční požadavky

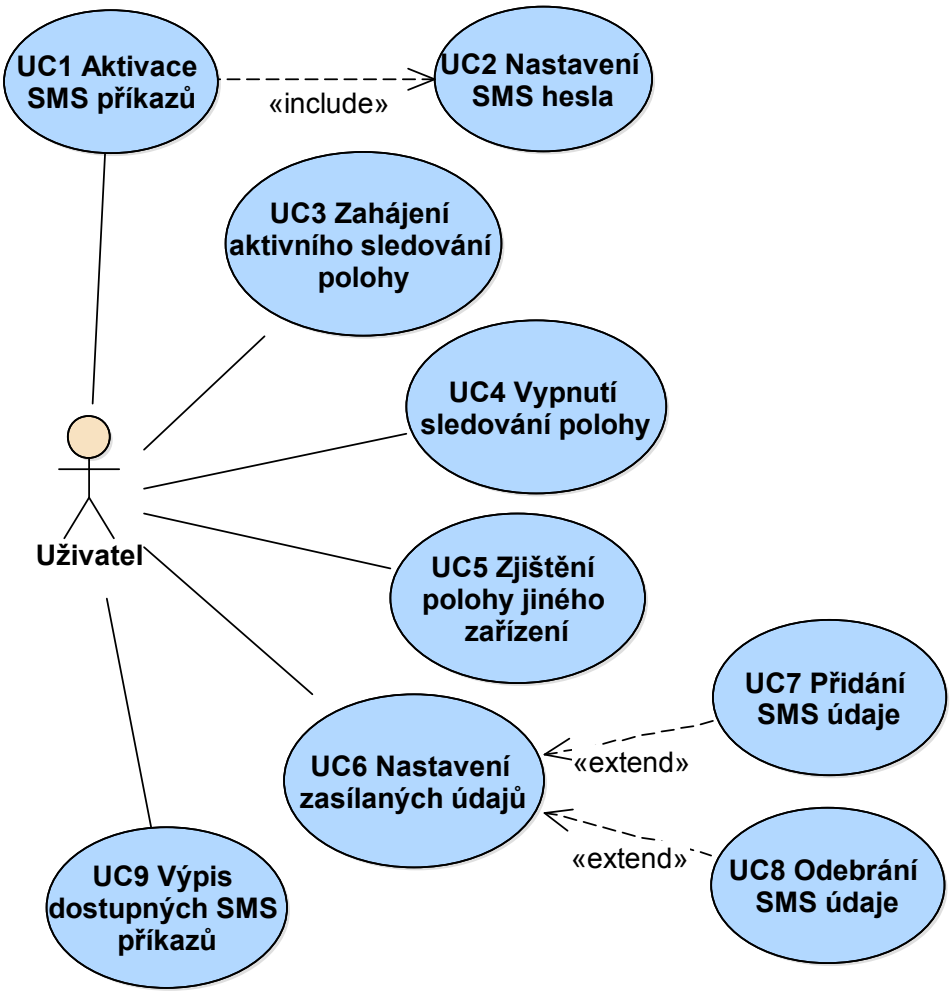
N1. Podpora minimálně 90% aktivních zařízení Aplikace bude fungovat na takovém počtu verzí OS Android, aby byla dostupná nejméně pro 90% aktivních zařízení. Pro stanovení aktuálně využívaných verzí systému se bude vycházet z dat od společnosti Google. [21]

N2. Podpora pro různé velikosti obrazovky Uživatelské rozhraní aplikace se bude přizpůsobovat displeji zařízení. Ovládací prvky budou vždy viditelné a nebude docházet k chybám v rozvržení jednotlivých prvků.

N3. Lokalizace bez připojení k Internetu Zařízení bude schopno nahlásit svoji polohu i bez připojení k Internetu.

N4. Lokalizace pomocí GPS Pro zjištění polohy bude preferován systém GPS. Pokud se nepodaří do stanovené doby určení polohy pomocí GPS, bude možné využít ostatních způsobů lokalizace.

N5. Stabilita Aplikace bude fungovat bez pádů, které by narušovaly a omezovaly možnosti jejího využití. Bude nutné počítat s možnými chybovými stavy a uživatele na ně vhodnou formou upozornit.



Obrázek 5.1: Případy užití aplikace

5.2 Případy užití

Případy užití slouží k tomu, abychom si dokázali lépe představit způsoby používání výsledné aplikace. Jedná se o diagramy znázorňující vnější fungování softwaru – tedy z pohledu koncového uživatele. Během jejich definování dochází k upřesnění požadované funkcionality jak pro programátora, tak i pro zákazníka, který software poptává. Diagramy případů užití tak mimo jiné pomáhají snížit riziko nedorozumění mezi zadavatelem a realizátorem výsledného produktu. Níže uvádím popis jednotlivých případů užití, viz diagram 5.1.

UC1 Aktivace SMS příkazů Nastavení aplikace tak, aby byla schopná korektně reagovat na příchozí SMS příkazy. To vyžaduje od uživatele především povolení přístupu k SMS zprávám a poloze v systému.

UC2 Nastavení SMS hesla Uživatel bude mít možnost nastavit vlastní heslo, které bude vyžadováno pro vykonání SMS příkazů.

UC3 Zahájení aktivního sledování polohy Aplikace poskytne jednoduché rozhraní pro konfiguraci a aktivaci nepřetržitého sledování polohy daného zařízení.

UC4 Vypnutí sledování polohy V případě, že uživatel nemá v tuto chvíli zájem monitorovat toto zařízení, je možné sledování polohy vypnout.

UC5 Zjištění polohy jiného zařízení Uživatel má tuto aplikaci nainstalovanou na fyzicky dostupném zařízení, prostřednictvím kterého chce najít polohu dalšího zařízení (také s touto aplikací).

UC6 Nastavení zasílaných údajů Konfigurace dat, která budou obsažena v odpovědi zařízení na požadavek o poloze.

UC7 Přidání SMS údaje Zařazení údaje (např. adresa) mezi zasílané.

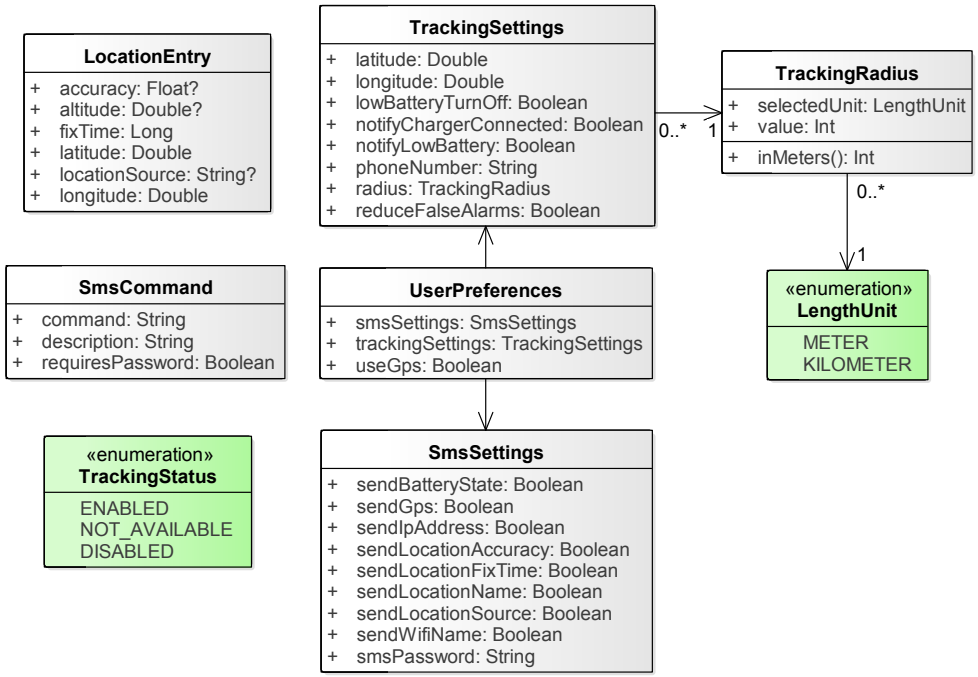
UC8 Odebrání SMS údaje Odebrání údaje ze seznamu zasílaných dat.

UC9 Výpis dostupných SMS příkazů Uživatel chce zjistit, jaké příkazy může zasílat, jejich popis a přesný formát.

5.3 Doménový model

Doménový model si lze představit jako model tříd, který je ochuzený o všechny „implementační detaily“ a jsou v něm obsaženy pouze ty entity, které mají souvislost s daným problémem. Podobně jako u diagramu případů užití, i doménový model můžeme využívat při komunikaci se zákazníkem k lepšímu pochopení problematiky na obou stranách.

Doménový model pro tuto aplikaci naleznete na obrázku 5.2.



Obrázek 5.2: Doménový model

LocationEntry je klíčová entita používaná v celé aplikaci. Obsahuje data o poloze zařízení, vždy bude obsahovat minimálně zeměpisnou šířku, zeměpisnou délku a čas získání polohy. V závislosti na okolnostech (zdroj polohy) může obsahovat také další údaje, jako je nadmořská výška nebo přesnost polohy.

TrackingStatus je výčtový typ, který reprezentuje stav sledování polohy. Řídí se podle něj uživatelské rozhraní obrazovky s konfigurací sledování, reakce na hlášení nové polohy aj.

TrackingSettings je třída uchovávající veškerá data o parametrech probíhajícího sledování polohy. Jsou v ní obsaženy zeměpisné souřadnice a poloměr sledované oblasti a další doplňující nastavení.

TrackingRadius je entita reprezentující definici poloměru sledované oblasti.

LengthUnit je výčet všech povolených jednotek pro konfiguraci poloměru sledované oblasti.

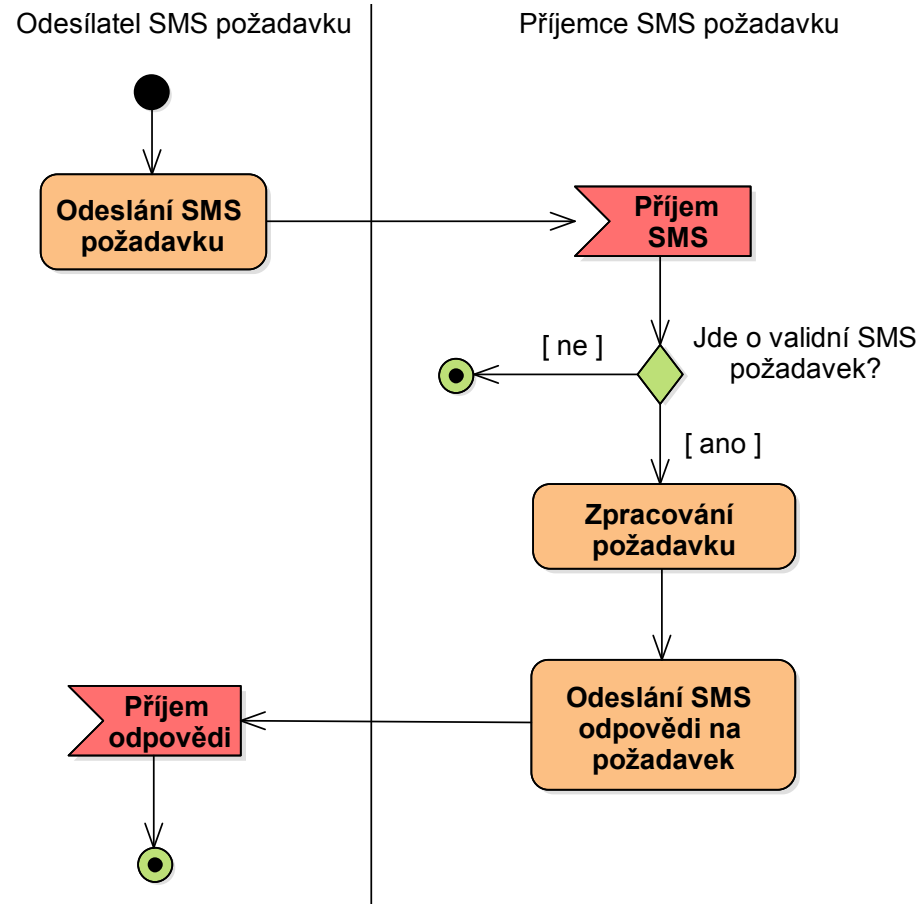
SmsCommand je entita reprezentující SMS příkaz. Obsahuje klíčové slovo, které je nutné uvést ve zprávě pro aktivaci příkazu, popis pro uživatele

zobrazený ve výpisu všech SMS příkazů a údaj o tom, zda je pro jeho použití vyžadováno SMS heslo.

SmsSettings je třída, ve které jsou uložena data o formátu odpovědí na příchozí požadavky o zaslání polohy.

UserPreferences je třída poskytující přístup k veškerým uživatelským nastavením, mimo jiné již zmíněným nastavením sledování polohy a formátu SMS hlášení polohy.

5.4 Komunikace se zařízením na dálku



Obrázek 5.3: Princip SMS komunikace mezi zařízeními

Veškerá komunikace s jiným zařízením bude možná bez připojení k Internetu, pouze pomocí SMS zpráv („SMS příkazů“) v přesně daném tvaru. Zpráva

vždy musí obsahovat nejprve klíčové slovo `LT` (zkratka názvu aplikace) na začátku zprávy, které označuje, že nejde o klasickou zprávu, ale o SMS příkaz určený pro tuto aplikaci. Součástí příkazů je také SMS heslo. Díky tomu je možné zamezit neoprávněnému ovládání telefonu nebo zjišťování naší polohy od třetích osob. Jako poslední následuje klíčové slovo označující název samotného příkazu, tedy např. `FIND` pro získání informací o aktuální poloze. Všechny výše zmíněné části jsou odděleny mezerou. Jelikož je v SMS zprávě snadné dopustit se drobného překlepu, přebytečné mezery na začátku nebo mezi částmi zprávy jsou při zpracování příkazu odstraněny tak, aby aplikace rozpoznala příkaz správně. Klíčová slova je také možné zadávat jak velkými, tak malými písmeny, což ale neplatí pro požadované SMS heslo, které musí být zadáno přesně včetně malých/velkých písmen. Validní příkaz (takový příkaz, který je syntakticky zprávně a obsahuje správné heslo) pro zjištění aktuální polohy tak může vypadat například takto – `LT passw757 FIND`, ale stejná akce se provede i po přijetí příkazu v následujícím tvaru – `lt passw757 Find`.

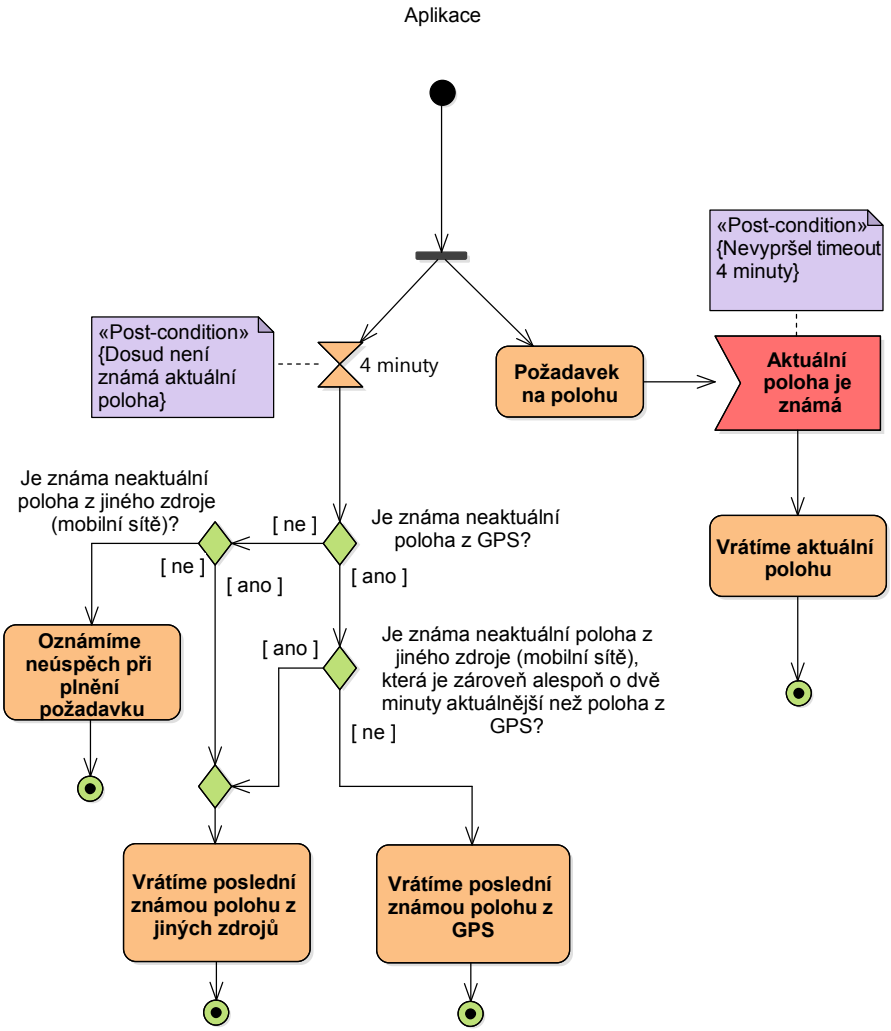
Aplikace na straně příjemce SMS zprávy po přijetí příkazu začne vykonávat požadovanou činnost (např. zjišťovat svoji polohu). Poté, co jsou potřebná data získána, je odpověď zpracována a odeslána odesílateli. Ta může být ve tvaru větné odpovědi (např. „Nepodařilo se zjistit polohu“) nebo souboru dat, která jsou z důvodu úspory místa uvedena za sebou a oddělena speciálním oddělovačem, typicky středníkem, což se využívá především pokud zasíláme zpět údaje o poloze zařízení. Diagram popisující příjem SMS zpráv naleznete na obrázku 5.3.

V seznamu níže jsou uvedeny příkazy, které aplikace bude v první verzi podporovat.

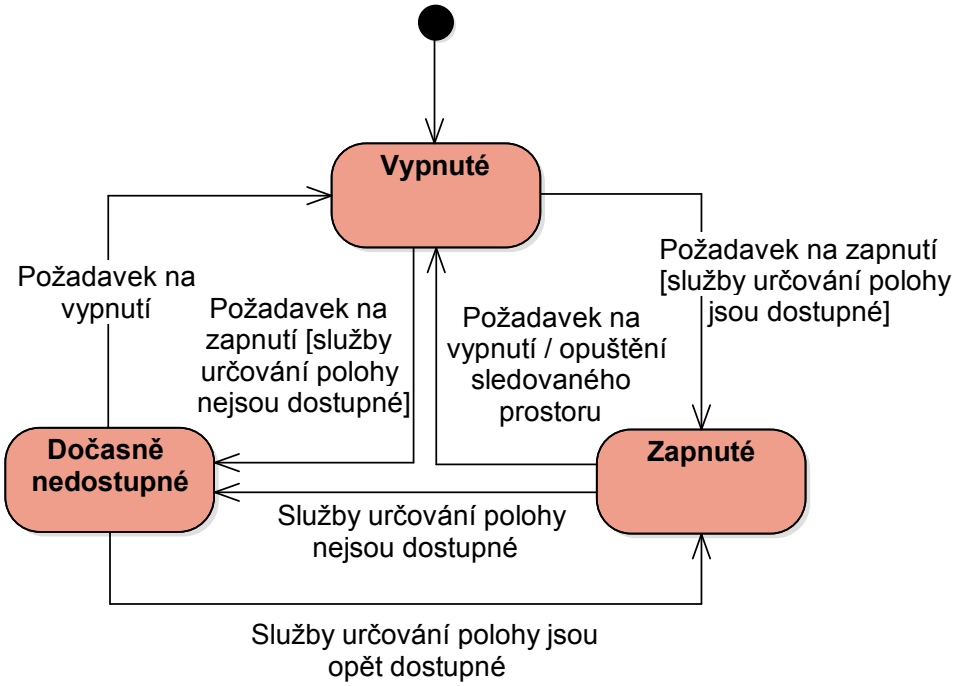
`FIND` – Po přijetí tohoto příkazu začne zařízení zjišťovat svoji polohu. Má na to časový limit 4 minuty, pokud během nich nedojde k získání polohy, použije se poslední známá poloha zařízení. V ojedinělých případech nemusí být dostupná ani poslední známá poloha, odesílateli se v takovém případě zašle oznámení o chybě při získávání informací o poloze zařízení. Podrobněji způsob získání polohy popisuje diagram 5.4.

`GPS` – Tento příkaz funguje na stejném principu jako předchozí příkaz `FIND`, narozdíl od něj ale nebere v potaz uživatelské nastavení formátu SMS zpráv, ale vždy pošle pouze tři údaje – zeměpisnou šířku, zeměpisnou délku a čas lokalizace.

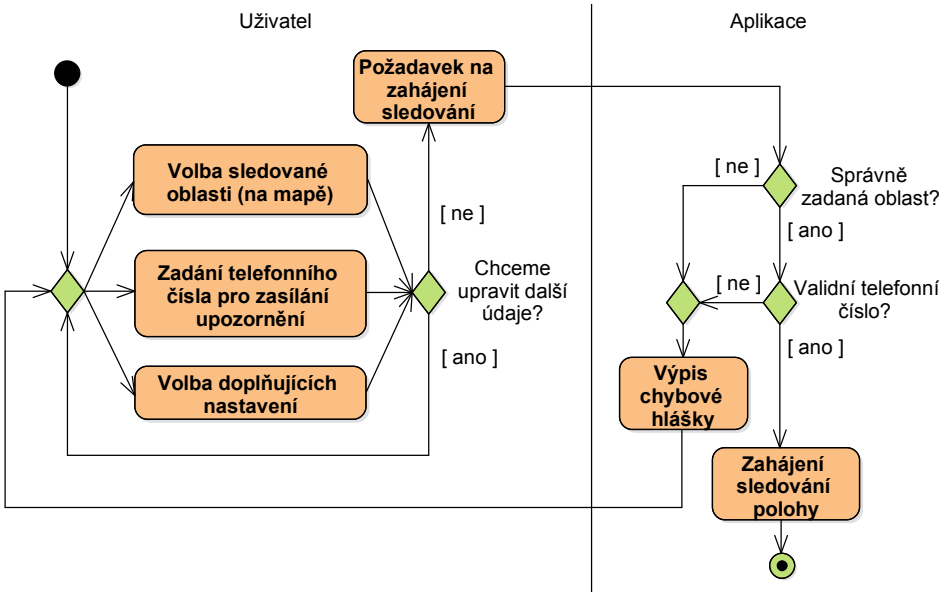
`BATTERY` – SMS příkaz, který použijeme v případě, že chceme zjistit pouze aktuální stav nabití baterie daného zařízení.



Obrázek 5.4: Princip získávání informací o poloze



Obrázek 5.5: Stavový diagram aktivního sledování polohy



Obrázek 5.6: Diagram aktivit popisující proces zahájení sledování polohy

5.5 Aktivní sledování polohy

Sledováním polohy se v rámci této aplikace rozumí nepřetržité získávání informací o poloze zařízení a hlídání, zda je současná poloha v tzv. sledované oblasti. Tato funkce je známá také pod anglickým názvem „geofencing“, což znamená provedení určené akce při vstupu nebo opuštění určeného prostoru. Sledovanou oblast může uživatel konfigurovat na mapě, lze nastavit umístění dané oblasti (střed) a její velikost (poloměr). Po spuštění sledování polohy začne zařízení sbírat informace o aktuální poloze a pokud zjistí, že zařízení opustilo stanovenou oblast, zašle upozornění na nastavené SMS číslo a ukončí sledování. Sledování je poté ukončeno z důvodů úspory baterie, aplikace však nadále bude reagovat na všechny příchozí SMS požadavky.

Funkci aktivního sledování polohy tedy využijeme v případech, kdy chceme být co nejdříve informováni o případném pohybu. Díky této aplikaci tak můžeme zařízení využít například jako jednoduchý GPS alarm do auta, díky kterému se ihned dozvíme o sebemenším pohybu vozidla. Mezi další způsob využití je možné zařadit hlídání telefonu nebo tabletu v případě, kdy odejdeme pryč z domova atd. Jelikož zařízení musí být při získávání polohy „probuzené“, vybíjí se o něco rychleji oproti standby režimu, při aktivním sledování polohy je tedy vhodné zajistit nabíjení, případně nevyužívat sledování na příliš dlouhou dobu v případě napájení z baterie.

Možné stavy sledování polohy jsou popsány na obrázku 5.5. Průchod aplikací při aktivaci sledování polohy je znázorněn diagramem 5.6.

5.6 Návrh uživatelského rozhraní

Před implementací aplikace zbývá poslední část, kterou je návrh uživatelského rozhraní. Návrh slouží k definování způsobů, jak by aplikace měla vypadat a reagovat na uživatelské vstupy. Pokud bych tuto část při vývoji vynechal, vedlo by to k nižší kvalitě výsledného rozhraní a pravděpodobně bych se nevyhnul předělávání některých částí aplikace, protože nedostatky by byly objeveny až během implementace.

Na začátku jsem používal náčrtky na papír, později, když už jsem měl představu, jak by mohl výsledek vypadat, jsem náčrtky převedl do digitální podoby na tzv. wireframes pomocí nástroje Pencil [22].

5.6.1 Rozvržení aplikace

Rozhraní této aplikace je členěno do následujících sekcí:

1. hlavní menu,
2. obrazovka konfigurace sledování polohy,
3. sekce pro nalezení zařízení,

4. konfigurace formátu SMS zpráv,
5. nastavení,
6. nápověda.

Výchozí obrazovkou je hlavní menu, které uživateli poskytuje důležité informace o stavu aplikace a slouží jako rozcestník do dalších částí. Aby uživatel nebyl nucen při přechodu mezi sekcemi pokaždé chodit zpět do hlavního menu, umístil jsem do spodní části navigační lištu, pomocí které se lze také přepínat mezi různými obrazovkami.

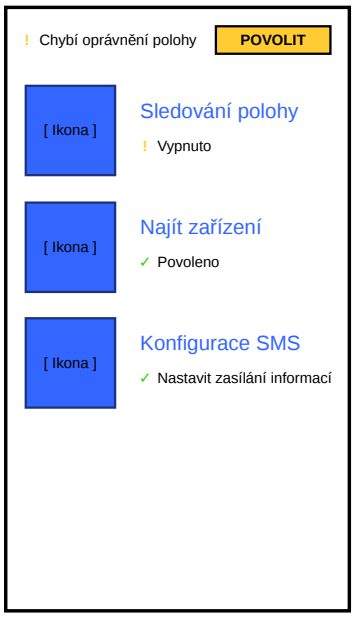
Barvy rozhraní aplikace jsem vybral pomocí nástroje Material Design Color Tool [23], jejich seznam je uveden níže:

Primární barva: Světle modrá – `#039be5`

Primární barva tmavá: Modrá – `#006db3`

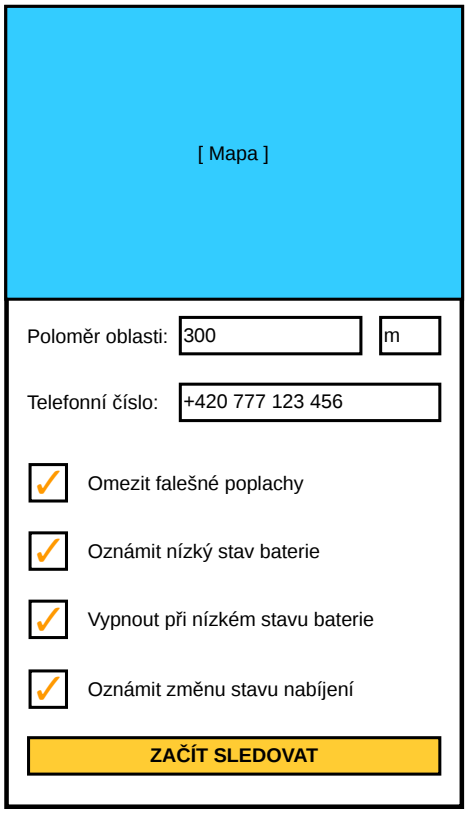
Doplňková barva: Oranžová – `#fb8c00`

5.6.2 Úvodní menu

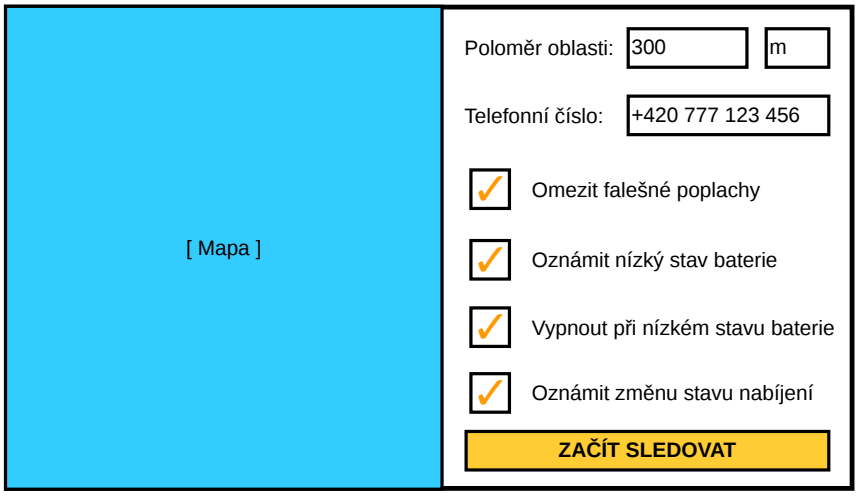


Obrázek 5.7: Wireframe hlavního menu

Hlavní menu obsahuje odkazy na ostatní části aplikace, u každého odkazu je uveden buď doplňující popis (např. „Nastavit zasílané informace“) nebo stav dané komponenty (např. „Není dostupné“ u aktivního sledování polohy).



Obrázek 5.8: Wireframe obrazovky „Sledování polohy“



Obrázek 5.9: Wireframe obrazovky „Sledování polohy“ v orientaci na šířku

V případě chybějících oprávnění je uživatel upozorněn zprávou umístěnou ještě nad samotnou nabídkou.

Pro lepší představu přikládám wireframe na obrázku 5.7.

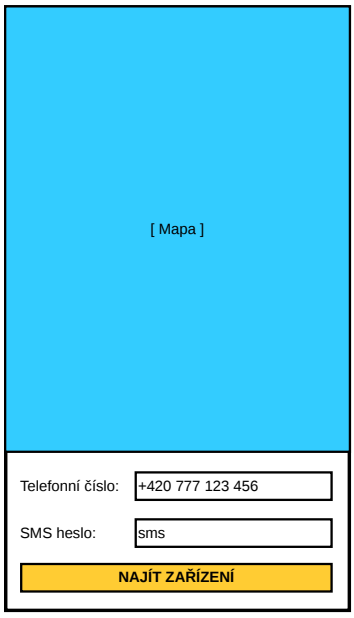
5.6.3 Sledování polohy

Sekce umožňující konfigurovat a zapnout/vypnout aktivní sledování polohy. Klíčovou součástí obrazovky je zde mapa, na které uživatel vidí svoji aktuální polohu a může pomocí ní nastavit sledovanou oblast. Pod mapou se nachází formulář, který slouží k nastavení poloměru sledované oblasti a zadání telefonního čísla, na které se budou zasílat veškeré informace o stavu sledování. Vespod se pak budou kromě tlačítka pro zapnutí sledování nacházet také checkboxy, pomocí kterých je možno nastavit zasílání doplňujících notifikací nebo chování při nízké baterii.

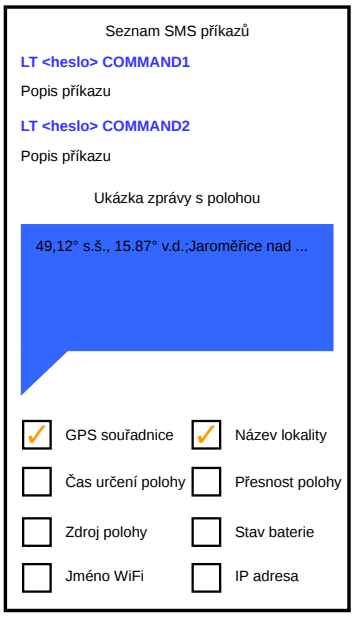
Tato část aplikace by ale vypadala v této podobě špatně při otočení telefonu na šířku – mapa by zabírala většinu obrazovky, formulář by byl zbytečně široký atp. Navrhnul jsem proto lehce odlišné rozvržení pro orientaci na šířku, ve kterém se mapa nachází v levé polovině obrazovky a zbytek ovládacích prvků zabírá pravou polovinu.

Obrazovka v obou orientacích displeje je znázorněna na obrázcích 5.8 a 5.9.

5.6.4 Najít jiné zařízení



Obrázek 5.10: Wireframe rozhraní pro vyhledání zařízení



Obrázek 5.11: Wireframe rozhraní pro konfiguraci SMS

Na obrázku 5.10 je znázorněno rozhraní pro snadné nalezení jiného zařízení. Většinu plochy zabírá mapa, pod kterou se nachází vstupní pole pro zadání telefonního čísla vyhledávaného zařízení a příslušného SMS hesla. Tlačítkem vespod se poté zahájí vyhledávání, pokud skončí úspěšně, lokace zařízení se zobrazí na mapě v podobě výrazné červené značky.

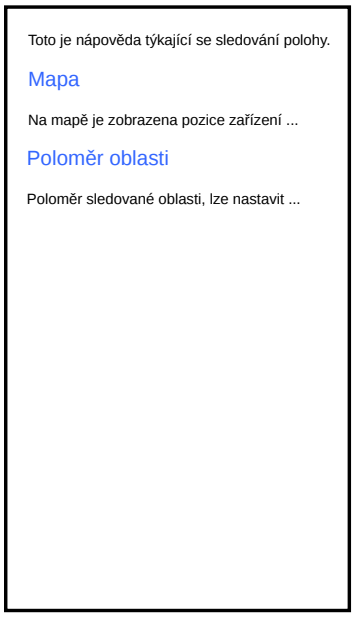
5.6.5 Konfigurace formátu SMS zpráv

Část aplikace, ve které uživatel nalezne přehled všech SMS příkazů včetně podrobného popisu a má zde možnost nastavit data zasílaná v SMS hlášeních polohy. Obrazovka obsahuje nejdříve seznam SMS příkazů, pod kterým se nachází ukázková SMS „bublina“ s textem reprezentujícím hlášení polohy. Vespod se pak nachází checkboxy, pomocí kterých lze přidat/odebrat položku z odpovědi, což se ihned projeví v ukázkové zprávě. Uživatel si tak díky příkladu snadněji představí, jak budou zprávy vypadat.

Wireframe obrazovky se nachází na obrázku 5.11.

5.6.6 Nápověda

Vzhledem k tomu, že může být složitější hned na začátku pochopit princip fungování aplikace, rozhodl jsem se přidat do aplikaci také jednoduchou nápovědu. Nápověda se vztahuje vždy ke konkrétní zvolené sekci. To znamená,



Obrázek 5.12: Wireframe sekce s nápovědou

že pokud se přepneme do nápovědy z obrazovky vyhledání zařízení, zobrazí se nápověda o vyhledávání jiných zařízení atd.

Obrazovka je znázorněna pomocí wireframe č. 5.12.

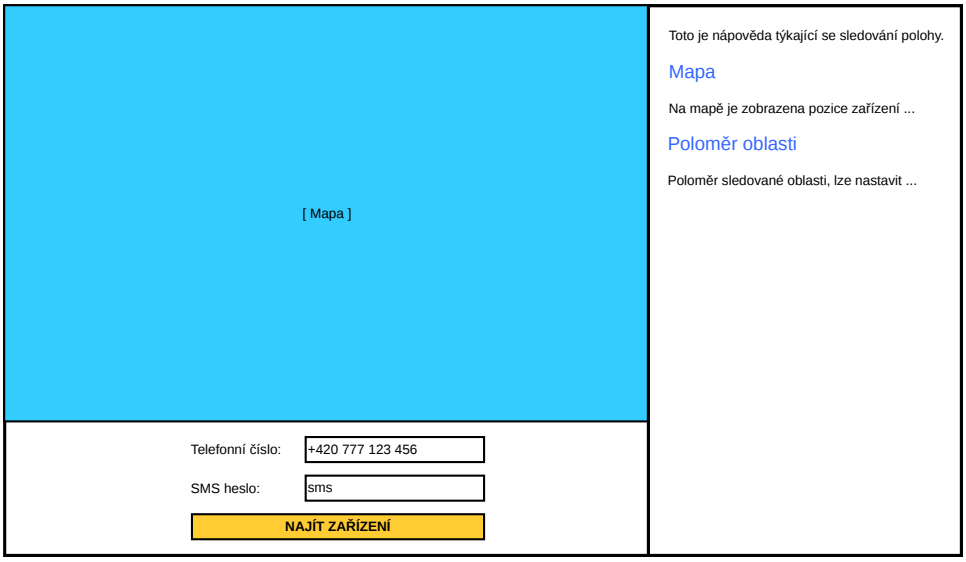
5.6.7 Přizpůsobení pro tablety

Rozhodl jsem se také pro optimalizaci aplikace pro zařízení s velikým displejem, jako jsou především tablety. Na tabletech v orientaci na šířku se zobrazí nápověda ke každé sekci na pravé straně displeje bez nutnosti dalšího prokliku, viz obrázek 5.13.

5.7 Architektura aplikace

Před začátkem vývoje během návrhu je vhodné promyslet si softwarovou architekturu aplikace a tu pak během implementace dodržovat. S dodržením tohoto pravidla snáze dosáhneme „čistého“ kódu, tedy takového, který je co nejjednodušší na pochopení a snadno udržovatelný a rozšiřitelný.

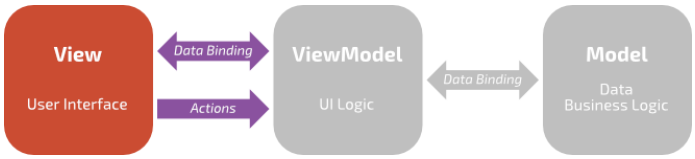
Prostředí Androidu nevynucuje používání žádné specifické architektury, existuje proto hned několik přístupů. Jedním z nich je **MVP** (Model-View-Presenter), což je varianta třívrstvé architektury oddělující datovou (Model), prezentační (View) a „business“ (Presenter) vrstvu. V případě interakce od uživatele prezentační vrstva komunikuje s presenterem, který pak obvykle



Obrázek 5.13: Wireframe znázorňující režim tabletu

provádí požadované operace s daty prostřednictvím modelu a nakonec zpět notifikuje prezentační vrstvu.

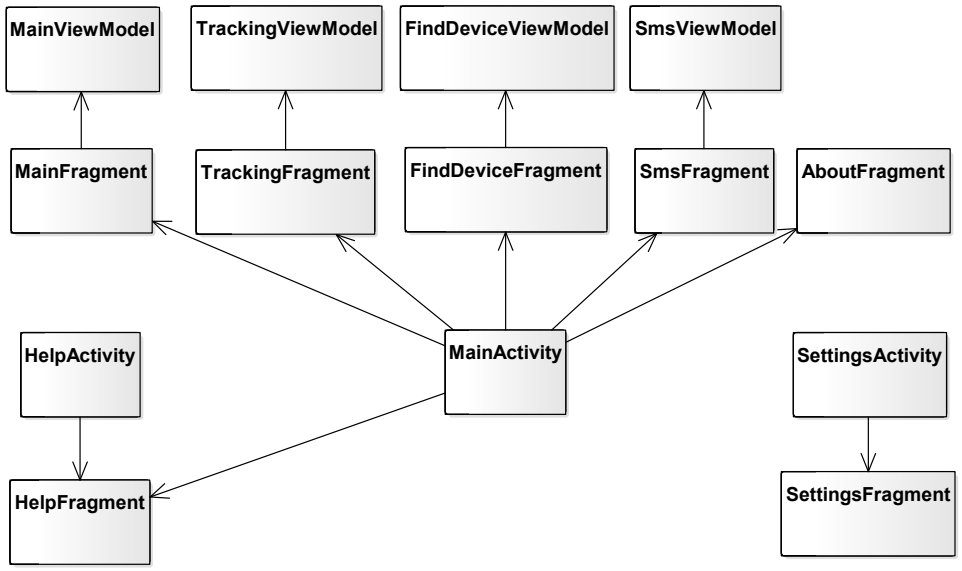
Mezi další architektury patří **MVVM** (Model-View-ViewModel), viz obrázek 5.14. Také v tomto případě jde o třívrstvou architekturu, hlavním rozdílem mezi MVP a MVVM je komunikace mezi prezentační a business (ViewModel) vrstvou. Zatímco v prvním jmenovaném případě je komunikace obousměrná, zde prezentační vrstva volá ViewModely a aktivně sleduje změny dat, která jsou ViewModelem poskytována. ViewModely tedy narozdíl od Presenterů nemají referenci na prezentační vrstvu, ta pouze reaguje na změny dat poskytovaných ViewModelem.



Obrázek 5.14: Diagram znázorňující MVVM architekturu [24]

Pro implementaci aplikace jsem nakonec zvolil architekturu MVVM, jelikož se mi s ní pracuje oproti MVP lépe. Mezi důvody bych zařadil jednoduchost propagace dat na více míst najednou nebo absenci reference na prezentační vrstvu z ViewModelu, což usnadňuje implementaci a pomáhá předcházet různým chybám, jako například neaktualizující se UI nebo úniky paměti (memory leaks).

Konkrétní podoba MVVM použitá v aplikaci je znázorněna na diagramu 5.15. Diagram popisuje provázání tříd uživatelského rozhraní (aktivit a fragmentů) a jejich napojení na ViewModely.



Obrázek 5.15: MVVM architektura použitá v aplikaci

Pro udržení pořádku ve zdrojovém kódu je nutné veškeré soubory rozdělit do více adresářů (balíčků) podle jejich funkce. Návrh adresářové struktury pro implementovanou aplikaci je uveden níže:

- `data` ..... entity a modelové třídy
- `di` ..... dependency injection třídy
- `location` ..... třídy zodpovědné za operace s polohou
- `screen` ..... implementace konkrétních obrazovek
  - `about` ..... obrazovka „O aplikaci“
  - `find` ..... obrazovka „Najít zařízení“
  - `help` ..... nápověda
  - `main` ..... vstupní obrazovka, hlavní menu
  - `settings` ..... nastavení aplikace
  - `sms` ..... obrazovka pro konfiguraci SMS
  - `tracking` ..... obrazovka pro konfiguraci sledování polohy
- `sms` ..... třídy pro práci s SMS
- `util` ..... pomocné třídy
- `view` ..... vlastní komponenty uživatelského rozhraní
- `viewmodel` ..... pomocné třídy pro práci s ViewModely

Kapitola 6

Implementace

V této kapitole popisuji postupy používané při samotné implementaci. Jsou zde popsány použité technologie, vývojové nástroje a knihovny. V dalších částech této kapitoly se věnuji specifikům při vývoji software pro systém Android a vysvětluji zde způsoby řešení jednotlivých problémů.

6.1 Použité nástroje a technologie

6.1.1 Programovací jazyk Kotlin

Pro implementaci aplikace jsem zvolil programovací jazyk Kotlin. Jedná se v současnosti o jeden ze dvou oficiálně podporovaných programovacích jazyků používaných pro vývoj aplikací určených pro operační systém Android [25]. Jelikož mým cílem je vytvořit nativní aplikaci, nikoli multiplatformní, neměl jsem důvod zvažovat jiné jazyky než Javu a Kotlin. Pro druhý jmenovaný jsem se nakonec rozhodl především z těchto důvodů:

Bezpečnější typový systém – Typový systém Kotlinu pomáhá předcházet nechvalně známým výjimkám `NullPointerException` díky rozlišování nullable a non-null typů. Kompilátor nedovolí přeložit kód, který by mohl vyústit v již zmíněnou výjimku (pokud není použit operátor `!!`). Výsledkem je tak často stabilnější aplikace.

Kompaktnější zápis – Kotlin obsahuje konstrukce, díky kterým je možné stejnou funkcionalitu zapsat přehledněji a zároveň na méně řádků, jednou z nich je např. `data class`.

Kompatibilní s Javou – Kotlin je 100% kompatibilní s Javou, což znamená, že je možné využívat veškeré pokročilé funkce Kotlinu, zároveň však máme možnost nadále využívat veškeré již existující knihovny a nástroje, které jsou napsané v Javě.

6.1.2 Android Studio

Android Studio [26] je v současnosti jediným oficiálně podporovaným vývojovým prostředím (IDE) pro vývoj Android aplikací. Na jeho tvorbě se podílí Google společně se společností JetBrains s.r.o. Jedná se o komplexní vývojové prostředí obsahující všechny nástroje, které vývojář potřebuje – inteligentní našeptávač, editor uživatelského rozhraní, debugger, profiler atd.

Jelikož se jedná o jediné oficiální prostředí pro Android vývoj a osobně mám zkušenosti s nástroji od firmy JetBrains, neměl jsem důvod hledat alternativu. Pro vývoj bylo používáno Android Studio ve verzích 3.1.1 a 3.1.2 na systému Ubuntu 17.10.

6.1.3 Android Emulator

Mobilní vývojář potřebuje průběžně testovat chování svých aplikací na mnoha zařízeních, která se liší verzí systému, velikostí a rozlišením displeje, senzorovou výbavou atd. Jelikož by bylo testování na větším množství fyzických zařízení zdlouhavé a nákladné, existuje emulátor – nástroj, prostřednictvím kterého je možné nakonfigurovat libovolné virtuální Android zařízení spustitelné na počítači. Google poskytuje oficiální Android Emulator jako součást Android SDK.

Během vývoje jsem používal prostřednictvím Android Emulatoru hned několik zařízení, jejich přehled je uveden v tabulce 6.1.

Tabulka 6.1: Přehled použitých virtuálních zařízení Android Emulator

Název	Rozlišení displeje	Verze OS	Poznámka
HVGA KitKat	320×480	4.4	obsahuje Google APIs
Nexus S Nougat	480×800	7.1.1	obsahuje Google APIs
Nexus 7 Nougat	800×1280	7.1.1	obsahuje Google APIs
Nexus 5 Oreo	1080×1920	8.1	obsahuje Google Play

6.1.4 Git

Git [27] je pokročilý verzovací systém. Verzovací nástroj umožňuje snadné sledovat historii vývoje, vracet se ke starším revizím zdrojového kódu, sdílet kód mezi více zařízení a mnoho dalšího.

Při implementaci byl používán jako verzovací systém právě Git ve verzi 2.17.0. Používal jsem ho v integraci s Android Studiem a webovým úložištěm repozitářů GitLab.

6.2 Použité knihovny

6.2.1 Android Support Library

Android Support Library [28] je oficiální knihovna, díky které je možné používat funkce uvedené v novějších verzích systému Android i na zařízeních se starším systémem. Její použití je dnes prakticky nutnost, jelikož je mezi uživateli stále značné množství telefonů a tabletů se starými verzemi systému.

Z knihovny jsem použil také řadu komponent, které nativní systém vůbec neobsahuje. Mezi ně patří např. `RecyclerView`, která se používá pro zobrazení seznamů, navigační lišta `BottomNavigationView` (na obrázku 6.1) nebo `ConstraintLayout`, který používám pro rozvržení prvků uživatelského rozhraní ve většině částí aplikace.

Obrázek 6.1: Komponenta `BottomNavigationView`

6.2.2 Google Play Services

Google Play Services [29] je další oficiální knihovnou od společnosti Google, poskytuje značné množství funkcí, mezi které patří push notifikace, Google Analytics, Android Pay a mnoho dalšího. V aplikaci však používám pouze Google Maps pro integraci mapové komponenty (viz obrázek 6.2).

Obrázek 6.2: Mapová komponenta z Google Play Services

6.2.3 Android Architecture Components

Poslední z řady použitých Google knihoven je Android Architecture Components [30], která usnadňuje implementaci MVVM architektury. Knihovna

je využívána napříč celou aplikací, neboť umožňuje používat ViewModely, které jsou klíčovou součástí MVVM. Kromě toho obsahuje také datový typ `LiveData`, pomocí kterého ViewModely mohou vystavovat svá data. Využití `LiveData` je však pouze volitelné, pro sledování změn dat jsem použil knihovnu RxJava popisovanou v kapitole 6.2.6.

6.2.4 Android-Support-Preference-V7-Fix

Pro sekci nastavení jsem zvolil přístup s využitím Androidem poskytované třídy `PreferenceFragment`, která slouží pro implementaci uživatelského rozhraní nastavení. Bohužel ale některé komponenty, které tato třída využívá, jako např. `EditTextPreference`, nezohledňují některé atributy, které fungují s komponentou `EditText`.

Tato knihovna, dostupná z [31], poskytuje soubor nových a „opravených“ tříd sloužících k implementaci nastavení, v tomto případě už bez zjevných chyb. Z knihovny jsem využil právě již zmíněnou „opravenou“ verzi komponenty `EditTextPreference`, která mi umožnila bez problému nastavit atribut `maxLength`, viz ukázka kódu 1.

```
<com.takisoft.fix.support.v7.preference.EditTextPreference
    android:key="sms_password"
    android:maxLength="15"
    android:title="@string/settings_sms_password" />
```

Výpis kódu 1: Použití `EditTextPreference` z knihovny Android-Support-Preference-V7-Fix

6.2.5 Dagger 2

Při návrhu aplikace jsem se rozhodl využít tzv. dependency injection, což je technika umožňující odebrat zodpovědnost třídám za vytváření jejich závislostí. Konkrétní třída tedy neřeší „jak“ závislost získat, pouze nám říká kterou závislost ke své funkci potřebuje.

Hledal jsem tedy knihovnu, která by toto umožňovala. Nakonec jsem se rozhodl pro Dagger 2 [32], což je asi nejpoužívanější dependency injection framework využívaný při Android vývoji. Výpis kódu 2 ukazuje příklad použití Daggeru při implementaci aplikace.

6.2.6 RxJava

Poslední z využívaných knihoven je RxJava [33]. Knihovna se často používá na psaní vícevláknového kódu, v této aplikaci ji však používám jako nástroj usnadňující implementaci Observer patternu, což je návrhový vzor usnadňující

propagaci událostí na více míst najednou. Pricipem tohoto přístupu je, že existuje `Observable` – datový typ, který při změně obsažených dat automaticky notifikuje všechny odběratele (`Observer`) přihlášené k odběru.

Knihovnu RxJava využívám nejčastěji pro poskytování dat z ViewModelů do prezentační vrstvy aplikace, viz ukázka kódu 3.

```
@Module
class AppModule {
    // Zde definujeme, jak se má instance poskytovat
    @Provides
    @Singleton
    fun provideNotificationController(context: Context):
                                NotificationController {
        return NotificationController(context)
    }
}

class App : Application() {
    @Inject // Zde vyžadujeme instanci
    lateinit var notificationController: NotificationController
}
```

Výpis kódu 2: Použití knihovny Dagger 2

```
class SmsViewModel : ViewModel() {
    fun observeExampleMessage(): Observable<String> = smsSubject

    fun onMessageSettingsEntryChanged(key: String,
                                      isEnabled: Boolean) {
        smsController.updateSmsSettingsEntry(key, isEnabled)
        // Propagace dat k odběratelům (např. SmsFragment)
        exampleSmsSubject.onNext(getExampleMessageText())
    }
}
class SmsFragment : Fragment() {
    override fun onResume() {
        // Nyní odebíráme, obdržíme vždy aktuální data
        viewModel.observeExampleMessage()
                .subscribe { txtExample.text = it }
    }
}
```

Výpis kódu 3: Použití knihovny RxJava 2

6.3 Uživatelské rozhraní

6.3.1 Členění aplikace

Jelikož vytvářím nativní aplikaci, využívám `Activity` (dále aktivity) z Android frameworku a třídu `Fragment` (dále fragmenty) z Android Support Library. Aktivita slouží jako vstupní bod, jedná se o část aplikace, která je viditelná pro uživatele. Každá Android aplikace poskytující nějakou formu uživatelského rozhraní musí obsahovat alespoň jednu aktivitu. Pokud se chceme přepnout do jiné aktivity, je předchozí aktivita skrytá, viditelná uživateli je v danou chvíli vždy jen jedna.

Tím se od aktivit liší fragmenty, které jsou koncipovány jako znovupoužitelné prvky uživatelského rozhraní. Jedna aktivita tak může obsahovat v závislosti na požadavcích i několik viditelných fragmentů najednou. Toho se využívá například při přizpůsobování aplikací na tablety, které mohou díky své velikosti najednou zobrazovat větší počet prvků uživatelského rozhraní než malé telefony. Výhodou fragmentů oproti aktivitám je i nižší náročnost na systém, vytvoření fragmentu trvá kratší dobu než vytvoření aktivity.

Aplikace obsahuje tyto aktivity:

1. **MainActivity** – hlavní aktivita obsahující všechny běžně používané prvky aplikace,
2. **SettingsActivity** – aktivita s fragmentem pro uživatelské nastavení,
3. **HelpActivity** – aktivita obsahující fragment s nápovědou.

Součástí jsou také tyto fragmenty:

1. **MainFragment** – hlavní menu,
2. **TrackingFragment** – obrazovka pro konfiguraci sledování polohy,
3. **FindDeviceFragment** – fragment s mapou sloužící k nalezení jiného zařízení,
4. **SmsFragment** – konfigurace SMS zpráv,
5. **SettingsFragment** – fragment poskytující uživatelské nastavení,
6. **HelpFragment** – fragment s nápovědou, na telefonech je součástí aktivity `HelpActivity`, na tabletech v orientaci displeje na šířku se může zobrazovat přímo v `MainActivity`,
7. **AboutFragment** – informace o aplikaci.

6.3.2 Použité layouty

Pro implementaci uživatelského rozhraní používám layouty fragmentů a dalších prvků definované pomocí XML souborů. Dříve zmíněné aktivity a fragmenty při inicializaci vykreslí svoje rozhraní právě na základě layoutů z XML. Android poskytuje mnoho druhů layoutů, což jsou speciální komponenty, do kterých lze vkládat prvky uživatelského rozhraní (textové popisky, tlačítka, checkboxy atd.), ale i další layouty. V následujícím seznamu uvádím popis layoutů použitých pro implementaci.

ConstraintLayout – Layout poskytovaný z Android Support Library. Umožňuje pozicovat prvky relativně vůči sobě podobně jako `RelativeLayout`, oproti němu ale poskytuje přesnější možnosti umístění (např. zarovnání pravého okraje prvku k pravému okraji jiného prvku). Díky tomu často není potřeba vnořovat více layoutů do sebe, ale postačí pouze jeden `ConstraintLayout`. V aplikaci je použit pro rozvržení většiny fragmentů.

LinearLayout – Jednoduchý layout, který umístěné prvky vkládá jeden po druhém za sebe. Podporuje nastavení horizontální orientace a vertikální, výchozí je horizontální. Orientace určuje, zda budou komponenty vkládáné pod sebe nebo vedle sebe. `LinearLayout` používám například pro definici prvku s popisem SMS příkazu, jako kořenový layout obrazovky sledování polohy pro orientaci displeje na šířku nebo pro rozvržení SMS obrazovky.

ScrollView – Tato komponenta umožňuje vložení jednoho prvku (typicky layoutu), který poté lze scrollovat, pokud jeho rozměry přesáhnou velikost displeje. Orientace je vždy vertikální, v případě potřeby horizontálního scrollování můžeme využít `HorizontalScrollView`. `ScrollView` používám v obrazovkách pro konfiguraci sledování polohy a SMS nastavení.

6.4 Zjišťování polohy

Pokud chce aplikace využívat přístup k poloze zařízení, má v zásadě dvě možnosti – buď využije třídu `LocationManager`, která je přímou součástí Android frameworku, nebo „Fused Location Provider API“, což je součást Google Play Services.

LocationManager – Jedná se o nativní součást systému Android. Umožňuje přesně definovat, jaký požadujeme zdroj polohy – máme na výběr použití GPS, polohy z mobilních sítí nebo pasivní zdroj (obdrží aktualizaci polohy jen v případě, že ji aktivně vyžaduje jiná aplikace). Tento přístup s sebou nese výhodu větší kontroly nad tím, jak přesnou informaci o poloze obdržíme a jak brzo bude dostupná. Nevýhoda použití tohoto

přístupu je v nutnosti ručně řešit přepínání zdrojů polohy a může vést také k vyšší spotřebě energie, pokud využíváme GPS po dlouhou dobu.

Fused Location Provider – API poskytované knihovnou Google Play Services, které nabízí jednodušší rozhraní pro programátora díky tomu, že se v tomto případě automaticky řeší výběr nejvhodnějšího zdroje polohy na základě požadavků na přesnost atd. Výhodou je oproti použití `LocationManager` jednodušší implementace a knihovna slibuje především nižší nároky na baterii zařízení. Nevýhodou je zde jen omezená možnost kontroly nad použitým zdrojem, což může vést k nedostupnosti informací o poloze když je potřebujeme, nebo nižší přesnosti dat.

Před začátkem vývoje aplikace jsem vyzkoušel oba přístupy a nakonec jsem se rozhodl pro první možnost, tedy využití `LocationManager`. Hlavním důvodem byla nižší spolehlivost řešení z Google Play Services, které poskytovalo aktualizace polohy s mnohem menší frekvencí a často i přesností. Jelikož jsou aktuální a přesné informace o poloze pro tuto aplikaci klíčové, vybral jsem si první přístup.

6.5 SMS komunikace

Aplikace ke správnému fungování potřebuje přijímat, zpracovávat a odesílat SMS zprávy. V této části proberu způsoby implementace přijímání zpráv, jejich zpracování a způsoby odesílání.

Příjem zpráv je co se týče náročnosti implementace paradoxně složitější než jejich odesílání. Nejprve musíme vytvořit a zaregistrovat `BroadcastReceiver`, což je speciální Android komponenta, která se volá v případě události. Aby se náš `BroadcastReceiver` volal jen v případech, kdy potřebujeme, musíme ho zaregistrovat do `AndroidManifest.xml` způsobem uvedeným v ukázce 4. Klíčový je především obsah mezi tagy `<intent-filter>`, který říká, že receiver má reagovat na událost `SMS_RECEIVED` zasílanou v případě příchozí SMS zprávy. Výpis kódu č. 5 pak ukazuje způsob, jakým z příchozí události (intentu) dostaneme text SMS zprávy.

```
<receiver
    android:name=".sms.SmsReceiver"
    android:permission="android.permission.BROADCAST_SMS">
    <intent-filter android:priority="999">
        <action android:name=
            "android.provider.Telephony.SMS_RECEIVED" />
    </intent-filter>
</receiver>
```

Výpis kódu 4: Registrace `BroadcastReceiver` pro příjem zpráv

```
override fun onReceive(context: Context, intent: Intent?) {
    val extras = intent?.extras
    if (extras != null) {
        var sender: String? = null
        val smsArray = extras.get(KEY_PDUS) as Array<*>
        val format = extras.get(KEY_FORMAT) as? String
        val output = StringBuilder()

        for (smsData in smsArray) {
            val sms = createFromPdu(smsData as ByteArray, format)
            // Postupně skládáme text zpráv pokud jsou rozdělené
            output.append(sms.messageBody)
            sender = sms.originatingAddress
        }

        if (sender != null) {
            val action = smsController
                .processIncomingSms(sender, output.toString())
            if (action !is SmsController.SmsAction.None) {
                // Zpráva obsahuje příkaz pro naši aplikaci
                smsController.onNewSmsAction(action)
                onActionRequired(action)
                abortBroadcast()
            }
        }
    }
}
```

Výpis kódu 5: `BroadcastReceiver` pro příjem zpráv

Odesílání zpráv je podstatně jednodušší, je ale nutné dát si pozor na to, zda zpráva nepřekročila povolenou délku jedné SMS. V tom případě se musí poslat jako multi-part SMS (SMS zpráva rozdělená na více částí). Vytvořil jsem proto metodu, která zjistí délku vstupního textu a podle toho se rozhodne, zda pošle obyčejnou zprávu, nebo multi-part. Metoda je uvedena v ukázce kódu 6.

Veškerá SMS komunikace s aplikací, ať už příchozí či odchozí je uchovávána v SMS klientu uživatele. Uživatel tak dostává notifikace o příchozích zprávách, i když se jedná o speciální SMS příkaz pro tuto aplikaci. Stejně tak si může přečíst odchozí odpovědi, např. s hlášením polohy. Jedná se o omezení systému Android, které nejde běžnými prostředky nijak obejít. Jediná aplikace, která má oprávnění skrývat zprávy je ta, kterou si uživatel zařízení vybere jako výchozího SMS klienta.

```
private fun sendSms(phone: String, text: String) {
    val textDivided = smsManager.divideMessage(text)
    if (textDivided.size > 1) {
        // Text se nevejde do jedné zprávy, rozdělíme
        smsManager.sendMultipartTextMessage(phone, null,
            textDivided, null, null)
    } else {
        // Text se do jedné zprávy vejde
        smsManager.sendTextMessage(phone, null,
            text, null, null)
    }
}
```

Výpis kódu 6: Implementace odesílání SMS zpráv

6.6 Oprávnění

Systém Android vyžaduje od všech aplikací, které používají potenciálně zneužitelné funkce, aby nejprve uživatele požádaly o oprávnění tyto funkce používat. Tento přístup se s časem vyvíjel, až do verze Android 5.1 Lollipop bylo nutné při instalaci odsouhlasit všechna oprávnění, která byla aplikací deklarována v manifestu (soubor `AndroidManifest.xml`). Bez toho nebylo možné aplikaci instalovat a používat. Zásadní změna přišla až s verzí Android 6.0 Marshmallow – aplikace nyní při instalaci nevyžadují žádná oprávnění, ale měly by tak učinit až za běhu v případě, že uživatel chce využít část aplikace, která oprávnění vyžaduje. Tento přístup se nazývá „runtime permissions“. Aby uživatel nebyl zahlcen spoustou žádostí o oprávnění, jsou některá „bezpečná“ oprávnění (např. přístup k Internetu nebo informacím o stavu WiFi připojení) udělena automaticky.

Jelikož používám dvě „nebezpečné“ skupiny oprávnění – přístup k poloze a SMS zprávám, runtime permissions využívám. Je nutné správně reagovat na případy, kdy uživatel oprávnění přidělit odmítne. V takovém případě se zobrazí informační hláška o nutnosti přidělení oprávnění pro používání dané funkce.

Kapitola 7

Testování

Posledním krokem před vydáním aplikace je testování. Aplikaci jsem průběžně testoval během vývoje, nejčastěji na emulátoru z důvodu snadné manipulace s polohou, ale i na fyzických zařízeních uvedených v tabulce 7.1. Po dokončení vývoje byla aplikace podrobena testu použitelnosti (usability test). Výsledky testování a objevené nedostatky včetně způsobů řešení shrnuji v následujících částech této kapitoly.

Tabulka 7.1: Přehled zařízení použitých k testování při vývoji

Název	Displej	Verze OS	Typ zařízení
HTC One M8	1080×1920 5”	8.1	smartphone
ASUS ZenPad Z10	2048×1536 9,7”	7.0	tablet

7.1 Testování programátorem

Během vývoje jsem narazil na pár problémů a chyb, které jsou uvedeny dále.

Problémy s načítáním mapy

V průběhu používání a testování aplikace jsem si všimnul poměrně výrazného (cca 0,5 s, může se lišit podle zařízení) zpoždění při přepnutí na obrazovku obsahující mapovou komponentu. Použití profiling nástroje v Android Studiu potvrdilo, že problém je v komponentě obsahující Google mapu. Znatelné je zpoždění při prvním přepnutí do mapy, poté již dojde k inicializaci Google Play Services a následující přepnutí jsou rychlejší (cca 0,2 s).

Bohužel se mi zatím nepodařilo najít uspokojivé řešení tohoto problému, který tak přetrvává i ve vydané aplikaci.

Chyba přidělování oprávnění

Jak již bylo řečeno v kapitole 6.6, v aplikaci se potřebná oprávnění přidělují až za běhu uživatelem (na zařízeních s verzí Android 6.0 nebo novější). Během testování jsem narazil na chybu, kdy aplikace nedostala oprávnění ke čtení a zasílání SMS, ale pouze ke čtení. Chyba se projevila pouze od verze Android 8.0. Problém byl způsoben tím, že aplikace požadovala pouze čtení SMS a nikoliv i zasílání. Tato dvě oprávnění spadají do stejné skupiny, jiné verze systému při schválení jednoho oprávnění automaticky povolily všechna oprávnění skupiny, verze Android 8.0 a novější už nikoliv.

Problém byl jednoduše vyřešen požadavkem na obě oprávnění najednou. Z pohledu uživatele nedošlo k žádné změně, dialog se žádostí povolení oprávnění je stále stejný.

Pád aplikace kvůli značce na mapě

Při testování jsem narazil na chybu, která způsobovala pád celé aplikace po přepnutí na obrazovku s konfigurací sledování polohy. Problém způsobovala značka sloužící k vyznačení polohy zařízení na mapě. Mapová komponenta umožňuje pro vzhled značky použít pouze bitmapy, nikoliv už vektory. Protože značka byla vektorová, došlo k pádu celé aplikace. Problém se ale projevoval pouze na zařízeních s verzí systému Android 5.0 a novější, na starších systémech vše fungovalo bez problému. Při sestavování aplikace totiž dojde automaticky k vygenerování bitmap z vektorových obrázků pro starší verze systému, které vektory nepodporují. Na starších systémech tudíž byla značka bitmapová a k problému nedocházelo.

Chybu jsem vyřešil prostým převedením značky z vektoru do bitmapy, která je tak nyní bez problému využívána ve všech verzích systému.

7.2 Uživatelské testování použitelnosti

Jakmile jsem dokončil implementaci aplikace a osobně jsem ji otestoval, nechal jsem aplikaci otestovat třemi uživateli. Cílem tohoto testu, tzv. „testu použitelnosti“, je získat informace o tom, jak je rozhraní aplikace srozumitelné pro uživatele, kteří s ní nikdy nepřišli do styku. Tento test se často provádí, jelikož vývojář nemůže zcela objektivně posoudit, jak je aplikace jednoduchá na používání, neboť ji sám vytváří a ví tedy přesně jak které prvky fungují. Na základě zpětné vazby se pak vyvíjený software upraví tak, aby byl pro uživatele jednodušší na ovládání.

Průběh testu

Všem uživatelům, kteří se testu zúčastnili, byl nejprve krátce vysvětlen účel aplikace. Následně jim byly položeny následující otázky:

- Popište zkušenosti s používáním systému Android. (začátečník, běžný uživatel, pokročilý uživatel).
- Uveďte název zařízení, na kterém se aplikace bude testovat a verzi operačního systému.

Dále jim byl poskytnut seznam činností, které měli během testu vykonat:

1. Zapněte sledování polohy.
2. Najděte jiné zařízení prostřednictvím aplikace.
3. Najděte jiné zařízení prostřednictvím SMS.
4. Nastavte data zasílaná ve zprávách.
5. Změňte SMS heslo.

Po dokončení testu byli účastníci požádáni o zodpovězení následujících otázek:

- Jak byste zhodnotili celkovou přívětivost aplikace?
- Co by se dalo na aplikaci zlepšit?
- Co se Vám na aplikaci nejvíce líbilo?

Výsledky testování

Tabulka 7.2: Odpovědi uživatelů na otázky před uživatelským testem

	Zkušenosti s OS Android	Zařízení	Verze OS
Uživatel 1	běžný uživatel	Huawei P9 Lite	7.0
Uživatel 2	pokročilý uživatel	HTC One M9	7.0
Uživatel 3	začátečník	Huawei Ascend P6	4.2.2

Zapněte sledování polohy

- Uživatel 1 vykonal test bez problému.
- Uživatel 2 vykonal test bez problému.
- Uživatel 3 vykonal test bez problému.

Najděte jiné zařízení prostřednictvím aplikace

- Uživatel 1 vykonal test bez problému.
- Uživatel 2 vykonal test bez problému.
- Uživatel 3 vykonal test bez problému.

Najděte jiné zařízení prostřednictvím SMS

- Uživatel 1 test vykonal, nalezení informace o SMS příkazu ale trvalo delší dobu.
- Uživatel 2 test vykonal, nalezení informace o SMS příkazu ale trvalo příliš dlouho. K vykonání testu pomohla až nápověda.
- Uživatel 3 test vykonal, očekával ale proklik ze seznamu příkazů do SMS aplikace.

Nastavte data zasílaná ve zprávách

- Uživatel 1 test vykonal, nalezení prvků pro nastavení dat ale trvalo delší dobu, jelikož byly vidět až po scrollování obrazovky.
- Uživatel 2 vykonal test bez problému.
- Uživatel 3 test vykonal, nalezení prvků pro nastavení dat ale trvalo delší dobu, jelikož byly vidět až po scrollování obrazovky.

Změňte SMS heslo

- Uživatel 1 vykonal test bez problému.
- Uživatel 2 test vykonal bez problému. Poznamenal, že by SMS heslo mohlo být zobrazeno nad seznamem s SMS příkazy.
- Uživatel 3 test vykonal, nalezení nastavení ale trvalo delší dobu.

Zpětná vazba od uživatelů po absolvování testu

Jak byste zhodnotili celkovou přívětivost aplikace?

- Uživatel 1: Aplikace mi přijde velice jednoduchá na používání, jen mi dělala trochu problémy SMS obrazovka.
- Uživatel 2: Část se sledováním polohy je neintuitivní, obrazovka s nastavením SMS by měla obsahovat zvolené SMS heslo.
- Uživatel 3: Moc jsem se neorientoval v SMS obrazovce, není jasné, že pod ukázkovou zprávou se dají nastavit data ve zprávě. Jinak se mi aplikace používala dobře.

Co by se dalo na aplikaci zlepšit?

- Uživatel 1: SMS obrazovka.
- Uživatel 2: Vylepšil bych grafiku obrazovky se sledováním polohy.
- Uživatel 3: SMS obrazovka – aby se po kliknutí na SMS příkaz otevřela SMS aplikace s předvyplněným příkazem. Podle mého by chtělo zvýraznit nastavení dat v SMS.

Co se Vám na aplikaci nejvíce líbilo?

- Uživatel 1: Celkově se mi aplikace až na drobnosti líbila a dobře se mi používala.
- Uživatel 2: Líbila se mi možnost posílat SMS příkazy a možnost nastavení zaslaných dat.
- Uživatel 3: Líbí se mi možnost najít mobil i na dálku jen pomocí SMS. Celkově aplikace vypadá dobře.

Shrnutí

Výsledky testování ukázaly, že největší problémy dělá obrazovka s konfigurací SMS zpráv. Uživatelé si stěžovali především na to, že ovládací prvky pro nastavování dat ve zprávách jsou vidět až po scrollování dolů. Na první pohled není jasné, že se obrazovkou dá scrollovat. Některým uživatelům také dělalo problém najít správný tvar SMS příkazu pro nalezení jiného zařízení.

Naopak s nastavením sledování polohy nebo hledáním zařízení přes aplikaci problémy nebyly, uživatelé ihned pochopili co mají udělat. Tyto části aplikace jsou proto ponechány beze změny.

Na základě výsledků uživatelského testování jsem provedl tyto úpravy obrazovky pro konfiguraci SMS zpráv:

- Seznam SMS příkazů se nyní nachází ve spodní částí obrazovky. Je stále částečně vidět, uživatelé proto snadněji pochopí, že se obrazovka dá scrollovat a v případě potřeby si SMS příkaz v seznamu najdou.
- Bylo přidáno zobrazení SMS hesla.
- Po vybrání SMS příkazu je uživatel přesměrován do aplikace pro zasílání SMS zpráv s předvyplněným příkazem.

Kapitola 8

Vydání aplikace a zhodnocení

8.1 Google Play Beta

Po dokončení implementace a uživatelského testování jsem aplikaci publikoval na Google Play Store, standardní kanál pro distribuci Android aplikací. Vydána je zatím v režimu otevřeného beta testování, narozdíl od aplikací zveřejněných v „ostré“ verzi je zařazena do sekce „Předběžný přístup“. Od produkčního kanálu se liší především zařazením do speciální sekce mezi ostatní aplikace ve vývoji a nemožností psát uživatelské recenze. Rozhodl jsem se pro publikaci v beta kanálu kvůli odhalení možných chyb a sbírání zpětné vazby před ostrým nasazením.

Aplikace je vydána na Google Play Store pod názvem „Location Tracker“, dostupná je na adrese `https://play.google.com/store/apps/details?id=cz.ojohn.locationtracker`. Postup instalace, ať už z Google Play, nebo ruční instalací z přiloženého balíčku, je podrobněji popsán v příloze B.

8.2 GitHub

Při vývoji byl používán verzovací systém Git, jak je uvedeno v kapitole 6.1.4. Používal jsem ho v kombinaci s privátním GitLab repozitářem, po dokončení první vydané verze jsem zdrojové kódy publikoval na GitHub, kde jsou zdrojové kódy aplikace veřejně ke stažení na adrese `https://github.com/ondrej-john/location-tracker`.

8.3 Zhodnocení a porovnání s ostatními nástroji

Podařilo se mi navrhnout a implementovat aplikaci, která uživatelům umožňuje snadno sledovat mobilní zařízení a aktivně upozorňovat, pokud svoji polohu změní. Především funkce aktivního sledování a upozornění mi u konkurenčních nástrojů chyběla, pokud byla dostupná, často jen v placené verzi.

Aplikace je zcela zdarma, zabírá minimum místa v telefonu (instalační balíček verze 0.1 má pouhých 2,2 MB) a je open-source – každý si tudíž může ověřit, že aplikace žádné údaje o uživateli nijak nezneužívá a ukládá je pouze po nezbytně dlouhou dobu.

Oproti některým ostatním nástrojům, porovnávaným v kapitole 4, postrádá moje řešení některé pokročilé funkce, například možnost pořídit fotografii, nahrát zvukový záznam nebo ovládání pomocí webového rozhraní. Cílem této práce ale bylo vytvořit aplikaci umožňující hlášení polohy, která bude jednoduchá na používání a všechny funkce bude poskytovat zcela zdarma a bez reklam, což výsledné řešení splňuje.

8.4 Možnosti rozšíření do budoucna

Jak již bylo zmíněno, aplikace je zatím vydána pouze v beta verzi. Před nasazením v produkční verzi je třeba především otestovat aplikaci na větším počtu zařízení a odhalit případné skryté chyby. Mimo to plánuji v budoucnu přidat do aplikace nové funkce, které jsou vyjmenovány v následujícím seznamu:

Lepší způsob zabezpečení – V současnosti aplikace brání zasílání informací nepovolaným osobám pomocí SMS hesla. Tento způsob zabezpečení je pro účely této aplikace dostačující a je používán i většinou konkurenčních řešení, ale není ideální. SMS zprávy jsou totiž přenášeny v nešifrované podobě včetně hesla, navíc ostatní aplikace mohou také SMS zprávy odchytávat a číst jejich obsah. Zvažuji proto implementaci doplňujícího zabezpečení, nejspíše formou uživatelem spravovaného „whitelistu“ – seznamu telefonních čísel, která budou mít povoleno zasílat SMS příkazy.

Databáze zařízení – Uživatel by měl možnost přidat telefonní čísla svých zařízení do seznamu, z kterého by pak snadno vybíral místo zadávání telefonních čísel ručně. Ke každému telefonnímu číslu by měl možnost přidat doplňující údaje, jako je název zařízení, zda z něj bude možno přijímat SMS příkazy (viz „whitelist“ v předchozím bodě) atd.

Další SMS příkazy – Aplikace v současném stavu reaguje na tři SMS příkazy. Do budoucna plánuji přidat další příkazy, pomocí kterých by bylo možné sledování polohy na dálku vypnout, zapnout, konfigurovat některé jeho parametry (jako je upozorňování na stav baterie apod.) nebo např. povolit/zakázat využití GPS.

Detekce změny SIM karty – Tato funkce by umožnila zaslat varovnou SMS zprávu na předem zadané telefonní číslo, pokud by byla detekována změna SIM karty v odcizeném zařízení. Varovná zpráva by obsahovala údaje o nové SIM kartě včetně telefonního čísla, bylo by tedy možné pokračovat ve sledování polohy zařízení i nadále.

Závěr

Zadáním práce bylo navrhnout mobilní Android aplikaci pro zjišťování polohy zařízení na dálku, poté aplikaci naimplementovat, řádně otestovat a nakonec vydat veřejně ke stažení. Součástí zadání také byla analýza způsobů určování polohy na mobilních zařízeních a srovnání dalších pěti aplikací, které řeší podobný problém. Všechny body zadání se podařilo splnit a výstupem práce tak je funkční mobilní aplikace ke sledování polohy pro systém Android.

Proces vývoje začal analýzou podobných řešení a posléze sběrem funkčních a nefunkčních požadavků. Poté, co byl dokončen návrh uživatelského rozhraní, bylo nutné navrhnout softwarovou architekturu. Nedílnou součástí práce byla samozřejmě samotná implementace softwaru, při které ale díky pečlivému návrhu naštěstí nedocházelo k žádným větším problémům. Po otestování a provedení následných menších úprav na základě zpětné vazby od uživatelů už nic nebránilo zveřejnění aplikace k otestování širšímu okruhu uživatelů.

Aplikace je v současnosti dostupná na Google Play Store pro veřejné testování. Ve svém volném čase plánuji aplikaci dále rozvíjet, především implementovat nové funkce, přidávat další SMS příkazy a ladit uživatelské rozhraní tak, aby bylo co nejsnadnější na používání. Jakmile se podaří otestovat aplikaci větším počtem uživatelů a vylepšit ji na základě zpětné vazby, bude uvolněna ke stažení v produkční verzi.

Literatura

1. CALLAHAM, John. The history of Android OS: its name, origin and more. *Android Authority* [online]. 2018 [cit. 2018-04-15]. Dostupné z: https://www.androidauthority.com/history-android-os-name-789433.
2. *HTC Dream (T-Mobile G1)* [online]. 2017 [cit. 2018-05-03]. Dostupné z: https://mobilenet.cz/obrazek/historie-htc-257086.
3. *Mobile Operating System Market Share Worldwide* [online]. 2018 [cit. 2018-04-17]. Dostupné z: http://gs.statcounter.com/os-market-share/mobile/worldwide.
4. POPPER, Ben. Google announces over 2 billion monthly active devices on Android. *The Verge* [online]. 2017 [cit. 2018-04-15]. Dostupné z: https://www.theverge.com/2017/5/17/15654454/android-reaches-2-billion-monthly-active-users.
5. BORNSTEIN, Dan. Dalvik JIT [online]. 2010 [cit. 2018-04-15]. Dostupné z: https://android-developers.googleblog.com/2010/05/dalvik-jit.html.
6. ART and Dalvik [online]. 2017 [cit. 2018-04-15]. Dostupné z: https://source.android.com/devices/tech/dalvik/.
7. Implementing ART Just-In-Time (JIT) Compiler [online]. 2017 [cit. 2018-04-15]. Dostupné z: https://source.android.com/devices/tech/dalvik/jit-compiler.
8. GRADLE INC. *Gradle Build Tool* [online] [cit. 2018-04-15]. Dostupné z: https://gradle.org/.
9. DOSHI, Bhavya. *The Android Compilation Process* [online]. 2017 [cit. 2018-05-03]. Dostupné z: http://www.theappguruz.com/app/uploads/2017/04/the-android-compilation-process.png.

10. NASA. *Global Positioning System History* [online]. 2017 [cit. 2018-04-16]. Dostupné z: https://www.nasa.gov/directorates/heo/scan/communications/policy/GPS_History.html.

11. The Evolution of GPS. *Illumin* [online] [cit. 2018-04-16]. Dostupné z: http://illumin.usc.edu/70/the-evolution-of-gps/.

12. *Satellite Navigation – GPS – How It Works* [online]. 2015 [cit. 2018-04-16]. Dostupné z: https://www.faa.gov/about/office_org/headquarters_offices/ato/service_units/techops/navservices/gnss/gps/howitworks/.

13. BURGGRAEF, Levent. *The principle of GPS positionig with three satellites.* [online]. 2011 [cit. 2018-04-16]. Dostupné z: http://wiki.awf.forst.uni-goettingen.de/wiki/index.php/File:GPS_principle.png.

14. *Manage location settings for Android apps* [online] [cit. 2018-04-16]. Dostupné z: https://support.google.com/accounts/answer/6179507?hl=en sekce „Turn Google's Location services on or off".

15. GOOGLE LLC. *Find My Device* [online] [cit. 2017-10-25]. Dostupné z: https://play.google.com/store/apps/details?id=com.google.android.apps.adm.

16. LSDROID. *Cerberus proti krádeži* [online] [cit. 2017-10-26]. Dostupné z: https://play.google.com/store/apps/details?id=com.lsdroid.cerberus.

17. AVAST SOFTWARE. *Avast Antivir a ochrana mobilu* [online] [cit. 2017-10-26]. Dostupné z: https://play.google.com/store/apps/details?id=com.avast.android.mobilesecurity.

18. ALIENMAN TECHNOLOGIES LLC. *Where's My Droid* [online] [cit. 2018-04-14]. Dostupné z: https://play.google.com/store/apps/details?id=com.alienmanfc6.wheresmyandroid.

19. SIMONSEN, Henrik. *TrackLoc – SMS Phone Tracker* [online] [cit. 2018-04-14]. Dostupné z: https://play.google.com/store/apps/details?id=no.henrikvs.trackloc.free.

20. SPARX SYSTEMS. *Enterprise Architect* [online]. Verze 13.5 [cit. 2018-04-29]. Dostupné z: http://sparxsystems.com/products/ea/.

21. *Dashboards* [online] [cit. 2018-04-16]. Dostupné z: https://developer.android.com/about/dashboards/index.html#Platform sekce „Platform versions".

22. EVOLUS CO., LTD. *Pencil* [online]. Verze 3.0.4 [cit. 2018-04-29]. Dostupné z: https://pencil.evolus.vn/.

23. GOOGLE LLC. *Color Tool – Material Design* [online] [cit. 2018-04-29]. Dostupné z: https://material.io/color/.

24. VAN BILSEN, Erik. *Model-View-ViewModel* [online] [cit. 2018-04-29]. Dostupné z: https://bloggrijjy.files.wordpress.com/2018/01/mvvm-v.png.

25. Kotlin and Android. *Android Developers* [online] [cit. 2018-04-30]. Dostupné z: https://developer.android.com/kotlin/.

26. GOOGLE LLC, JETBRAINS S.R.O. *Android Studio* [online]. Verze 3.1.2 [cit. 2018-04-30]. Dostupné z: https://developer.android.com/studio/index.html.

27. *Git* [online]. Verze 2.17.0 [cit. 2018-04-30]. Dostupné z: https://git-scm.com/.

28. *Android Support Library* [online]. Verze 27.1.1 [cit. 2018-04-30]. Dostupné z: https://developer.android.com/topic/libraries/support-library/.

29. GOOGLE LLC. *Google APIs for Android* [online]. Verze 15.0.0 [cit. 2018-04-30]. Dostupné z: https://developers.google.com/android/reference/packages.

30. GOOGLE LLC. *Android Architecture Components* [online]. Verze 1.1.1 [cit. 2018-04-30]. Dostupné z: https://developer.android.com/topic/libraries/architecture/.

31. KŐRÖSSY, Gergely. *Android Support library - preference v7 bugfix* [online]. Verze 27.1.1.0 [cit. 2018-04-30]. Dostupné z: https://github.com/Gericop/Android-Support-Preference-V7-Fix.

32. GOOGLE LLC. *Dagger* [online]. Verze 2.15 [cit. 2018-04-30]. Dostupné z: https://google.github.io/dagger/.

33. REACTIVEX. *RxJava* [online]. Verze 2.1.10 [cit. 2018-04-30]. Dostupné z: https://github.com/ReactiveX/RxJava.

34. BOIANO, Gianluca. *android-udev-rules* [online] [cit. 2018-05-03]. Dostupné z: https://github.com/M0Rf30/android-udev-rules.

Příloha A

Seznam použitých zkratek

OS Operační systém

VM Virtual Machine

JIT Just-In-Time

AOT Ahead-Of-Time

GPS Global Positioning System

PPS Precise Positioning System

SPS Standard Positioning Service

GSM Global System for Mobile Communications

BTS Base Transceiver Station

SSID Service Set Identifier

SMS Short Message Service

PIN Personal Identification Number

MVP Model-View-Presenter

MVVM Model-View-ViewModel

IDE Integrated Development Environment

SDK Software Development Kit

XML Extensible Markup Language

API Application Programming Interface

Příloha B

Instalační příručka

B.1 Instalace z Google Play

1. Stáhněte aplikaci z Obchodu Google Play `https://play.google.com/store/apps/details?id=cz.ojohn.locationtracker`.
2. Pokud budete vyzváni k zapojení do testování, potvrďte.
3. Aplikace je nainstalována.

B.2 Instalace z přiloženého souboru

Pomocí nástroje `adb`

Pro instalaci je potřeba následující:

- nástroj `adb` (součástí Android SDK),
- připojený telefon k počítači přes USB,
- USB ovladač v případě systému Windows,
- správně nakonfigurovaná udev pravidla na systému Linux (například z [34]).

Postup instalace:

1. Připojte telefon k počítači.
2. Povolte na telefonu USB ladění (Nastavení – Vývojářské nástroje).
3. Otevřete terminál ve složce s instalačním APK souborem.
4. Zadejte do terminálu: `adb install locationtracker.apk`.

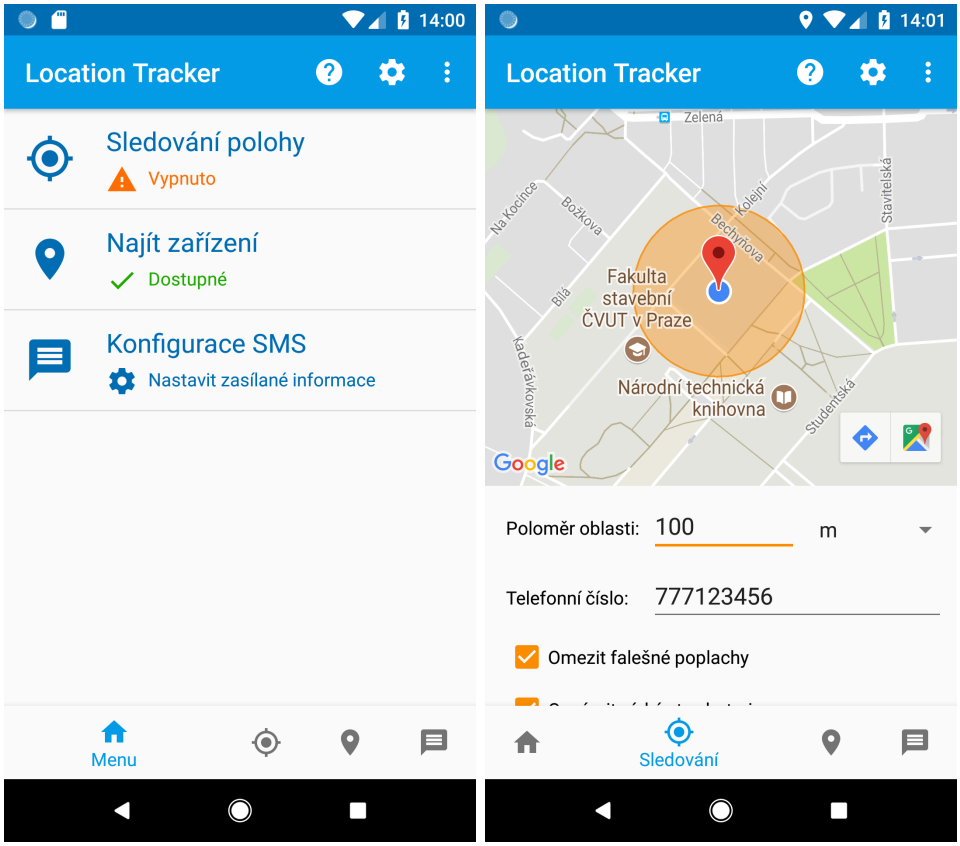
5. Pokud vše proběhlo v pořádku, na terminálu se objeví odpověď „Success“ a aplikace bude nainstalována.

Bez využití nástroje `adb`

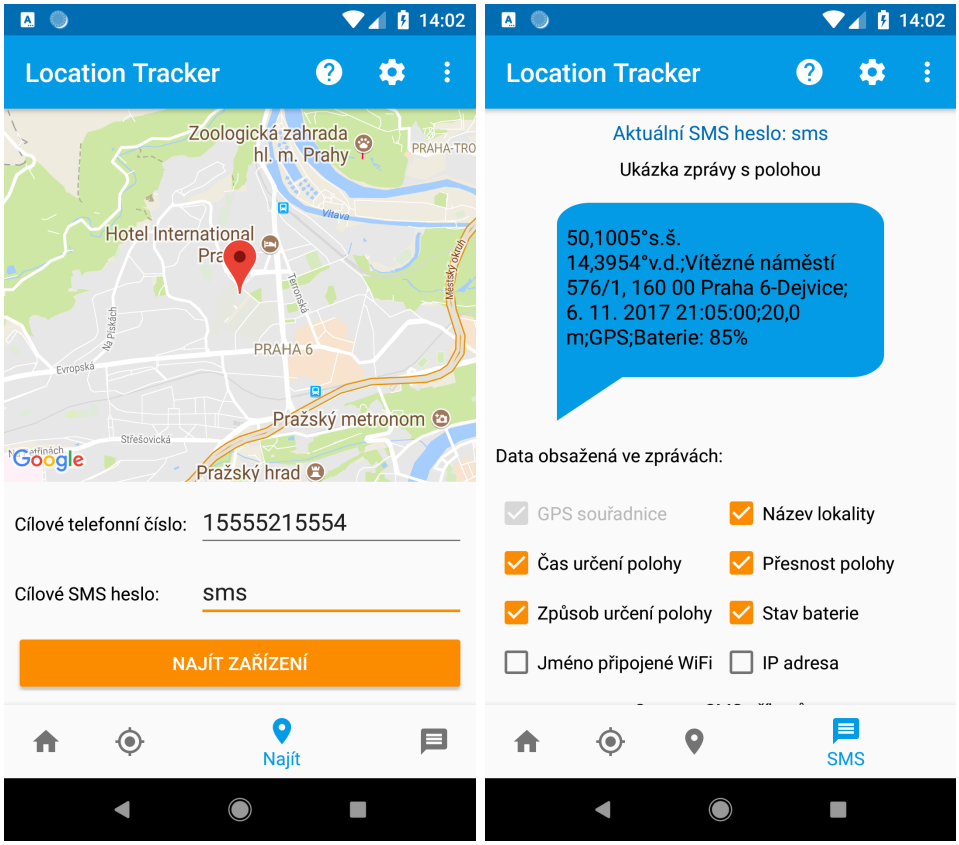
1. Zkopírujte instalační soubor `locationtracker.apk` do telefonu.
2. V telefonu povolte instalaci aplikací z neznámých zdrojů (Nastavení – Zabezpečení).
3. V telefonu nalezněte instalační balíček a spusťte ho.
4. Aplikace je nainstalována.

Příloha C

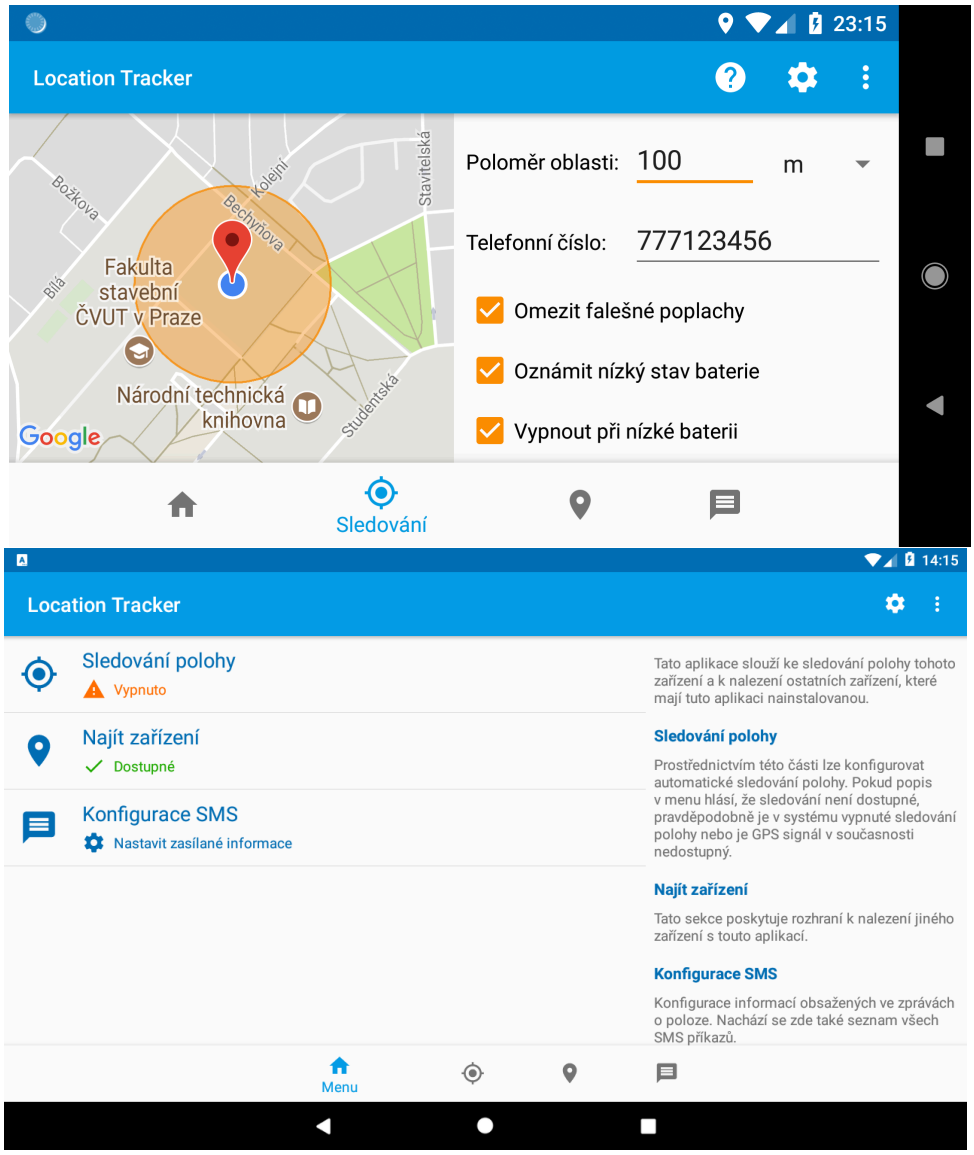
Finální podoba aplikace



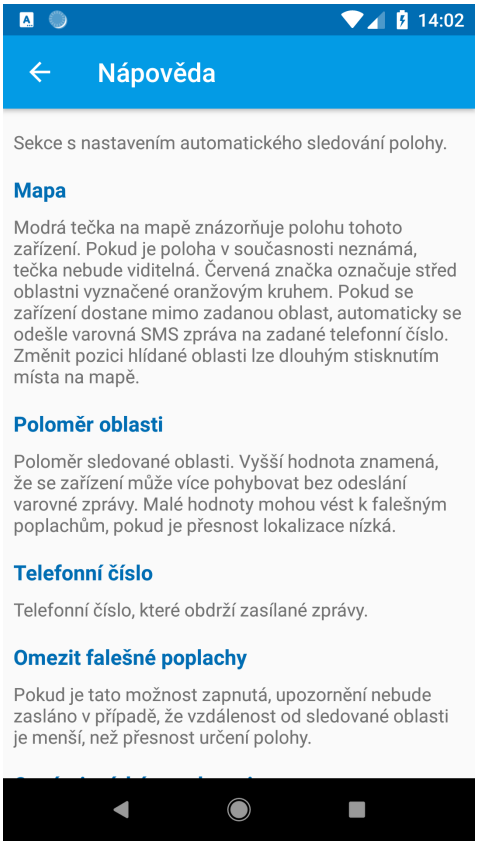
Obrázek C.1: Hlavní menu a konfigurace sledování polohy



Obrázek C.2: Obrazovka vyhledání zařízení a SMS konfigurace



Obrázek C.3: Layout sledování polohy v orientaci na šířku a režim tabletu



Obrázek C.4: Nápověda sekce sledování polohy

Příloha D

Obsah přiloženého CD

- `readme.txt` ..... stručný popis obsahu CD
- `locationtracker.apk` ..... instalační balíček aplikace
- `src`
  - `impl` ..... zdrojové kódy implementace
  - `thesis` ..... zdrojová forma práce ve formátu LaTeX
- `thesis.pdf` ..... text práce ve formátu PDF