Univerzita Hradec Králové

Fakulta informatiky a managementu

Katedra informatiky a kvantitativních metod

Sběr radiových fingerprintů na platformě iOS

Diplomová práce

Autor: Bc. Jan Dědek

Studijní obor: Aplikovaná informatika

Vedoucí práce: Ing. Pavel Kříž Ph.D.

Hradec Králové duben 2017

Prohlášení:

Prohlašuji, že jsem diplomovou práci zpracoval samostatně a s použitím uvedené literatury.

V Hradci Králové dne 24. dubna 2017 Bc. Jan Dědek

Poděkování

Děkuji vedoucímu diplomové práce Ing. Pavlu Křížovi Ph.D. za jeho cenné rady a odbornou pomoc.

Anotace

Tato diplomová práce se zabývá návrhem a implementací aplikace pro sběr radiových dat pro zařízení s operačním systémem iOS, samotným sběrem dat a vyhodnocením přesnosti získaných údajů. První a druhá kapitola je věnována historii technologie Bluetooth a praktickému využití Bluetooth Low Energy. Třetí kapitola obsahuje popis uživatelského rozhraní a funkcí aplikace Radio F-Prints, která je součástí této diplomové práce. Další část je zaměřena na popis struktury aplikace a zdrojového kódů, seznámení s důležitými třídami a jsou zde ukázky vlastního řešení vybraných problémů. Tato část je doplněna také o teoretický základ, nutný k pochopení vnitřních principů aplikace. Poslední část je věnována testování aplikace, sběru dat a vyhodnocení nasbíraných údajů.

Annotation

Title: Radio fingerprint acquisition on iOS platform

This diploma thesis deals with the design and implementation of the application for data collection on devices based on the iOS, data collecting and the evaluation of the accuracy of collected values. The first and second chapters are devoted to history of the Bluetooth technology and practical use of the Bluetooth Low Energy. Third chapter contains description of user interface and functions of Radio F-Prints application which is also a part of this diploma thesis. Next part is focused on description of the structure of the application and source code, introduction of important classes and examples of the solutions to chosen problems. This part contains a theoretical basis which is necessary for understanding the internal principles of the application. The last part is devoted to an application testing, data collecting and data evaluation.

Obsah

1 Úvod

V posledních letech došlo k velkému rozšíření chytrých zařízení, tedy přístrojů, do kterých mohou uživatelé instalovat software třetích stran. Velká část těchto zařízení je tvořena tablety a mobilními telefony, které mají uživatelé prakticky neustále při sobě. Současně dochází k postupnému zlevňování mobilního internetového připojení a také k navyšování počtu WiFi sítí, které umožňují uživatelům připojit se k internetu zcela zdarma v řadě veřejných prostor, jako jsou např. kavárny, nákupní centra, kongresové haly apod.

Díky tomu nejsou uživatelé omezení na používání off-line aplikací, ale mohou využívat i řešení, která za cenu požadavku neustálého připojení k internetu mohou nabídnout mnohem větší interaktivitu a mohou poskytnout aktuální data závislá na kontextu[1], ve kterém se uživatel právě nachází. Jedním z nich je možnost indoor navigace, tedy lokalizace uživatelů ve vnitřních prostorách budovy, kde je omezené nebo zcela vyloučené použití GPS. Díky neustálému připojení k internetu může uživatel dostávat řadu doplňujících informací, jako je např. název přednášky konané v posluchárně, kolem které právě prochází, nebo přehled restaurací v okolí, které odpovídají jeho preferencím.

Ke správné funkci podobných systémů je nutné provádět měření radiových dat, které mohou posloužit ke kalibraci jak jednotlivých vysílačů, tak i aplikací určených pro koncové uživatele. Tato práce se zabývá právě nástrojem pro měření síly signálu Bluetooth Low Energy zařízení a sběrem dalších podpůrných dat.

Cíle práce

Cílem této práce je návrh a implementace vlastního řešení pro zařízení se systémy iOS ke sběru informací o síle signálu Bluetooth Low Energy zařízení a dalších údajů, které mohou být užitečné pro kalibraci systému indoor navigace, popř. mohou být podrobeny další analýze za účelem zkoumání různých faktorů a vlivů, které mohou ovlivňovat jejich přesnost.

[1] Jako kontext jsou v tomto případě označovány skutečnosti, které mohou mít vliv na data, která mohou uživatele v dané chvíli zajímat. Může se jednat např. o geografickou polohu, denní dobu nebo třeba aktuální povětrnostní podmínky.

2 Technologie Bluetooth

2.1. Historie Bluetooth

První zmínku o této technologii můžeme nalézt již v roce 1994, kdy firma Ericsson, konkrétně její divize Mobile Communications Division, přišla s myšlenkou bezdrátového propojení mobilních telefonů a dalších periferií (PC World, 2009). V té době se k propojení zařízení běžně používal kabel RS-232 (**Obrázek 1**), jež měl být podle zadání firmy Ericsson nahrazen bezdrátovou variantou (Nordic Semiconductor, 2014).

Obrázek 1: Kabel RS-232

Zdroj: https://i.alza.cz/ImgW.ashx?fd=f3&cd=MK172

V roce 1998 byla firmami Ericsson, Nokia, Intel, Toshiba a IBM založena skupina The Bluetooth Special Interest Group (zkráceně BSIG). Cílem této skupiny bylo vytvoření univerzálního standardu. Už v následujícím roce došlo k vydání hned dvou verzí standardu a to 1.0a a 1.0b. V roce 2000 přichází na trh první komerčně prodávaný mobilní telefon a první sluchátka podporující technologii Bluetooth. Této technologie si začali všímat i další výrobci PC periferií a tak se na trhu objevilo velké množství hardware, podporující Bluetooth (tiskárny, myši, hands-free soupravy do automobilů atd.). Od té doby se tento protokol dočkal mnoha vylepšení a podpora Bluetooth se stala standardem. Bluetooth Low Energy bylo poprvé představeno v roce 2009 jako Bluetooth 4.0. které se v roce 2014 dočkalo ve verzi 4.2 vylepšení

bezpečnosti a rychlosti (Bluetooth SIG, 2017a). V roce 2016 došlo k vydání zatím poslední verze Bluetooth specifikace a to 5.0. Ta je zaměřena na další zvýšení rychlosti přenosu dat, dosahu signálu a snaží se vyhovět vzrůstajícím požadavkům internetu věcí. (Bluetooth SIG, 2017b).

2.2. Bluetooth Low Energy

Bluetooth low energy, neboli nízkoenergetické Bluetooth (zkráceně BLE) bylo vyvinuto speciálně pro internet věcí, tedy chytrá zařízení, která mezi sebou mohou komunikovat. Prioritou přitom není co nejvyšší přenosová rychlost, ale energetická nenáročnost a s tím spojená dlouhá výdrž baterie, která se pohybuje řádově v měsících až jednotkách let. Konkrétně u zařízení značky Estimote, která jsou rozmístěna v budově FIM a která byla použita pro testování během vývoje aplikace Radio F-Prints, je výrobcem deklarovaná výdrž baterie v řádu nižších jednotek let, dle typu konkrétního zařízení (Estimote Beacon, 2017).

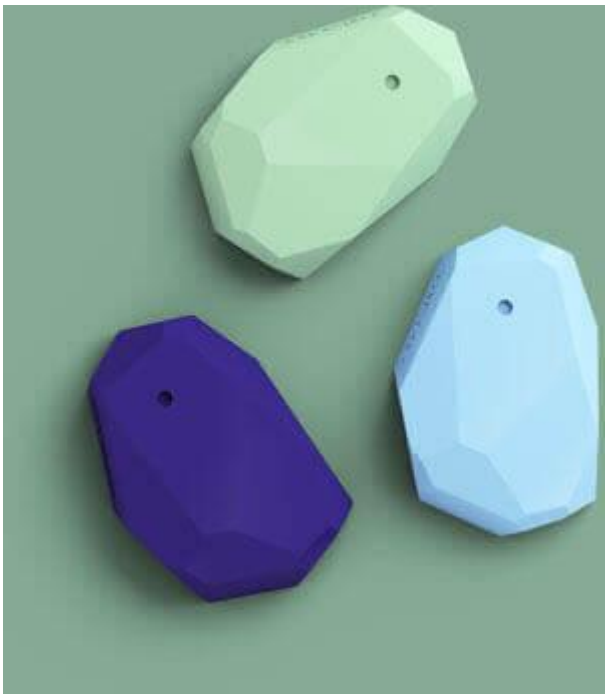
Pro potřeby navigace uvnitř budov jsou jako BLE vysílače nejčastěji použity beacony, což jsou miniaturní zařízení napájená zpravidla baterií, případně USB, která v pravidelných intervalech vysílají signál, který jsou schopna zachytit a zpracovat další zařízení. Četnost vysílání není pevně stanovena a je možné ji změnit. Řádově se pohybuje v desítkách vysílání za sekundu. Zvolená četnost má zásadní dopad na výdrž baterie. Vysílání probíhá zpravidla formou broadcastu. Odpadá zde tedy nutnost párování, protože zařízení nepotřebuje mít spojení s protistranou, pouze šíří svůj signál do okolí a nemá přitom informaci o tom, zda ho jiné zařízení zachytává a zpracovává.

Je zde také omezena velikost dat, které je BLE zařízení schopno odeslat. Zcela nevhodné je např. na streamování hudby, filmů atd. Aplikace Radio F-Prints pracuje pouze s hodnotou síly signálu RSSI[2] a MAC[3] adresou zařízení, která ho jednoznačně identifikuje.

[2] Z anglického Received Signal Strength Indication
[3] Z anglického Media Access Control

2.3. Použitý hardware

Jako Bluetooth vysílače byly při vývoji a testování aplikace Radio F-Prints využity beacony značky Estimote (***Obrázek 2***), které byly za tímto účelem zapůjčeny Univerzitou Hradec Králové. Aplikace však není při měření dat omezena na konkrétní typ hardware nebo jeho výrobce, ale je navržena pro sběr dat ze všech zařízení, provozovaných na technologii Bluetooth Low Energy.



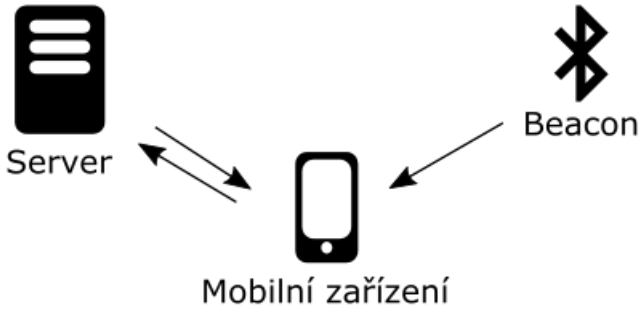
Obrázek 2: Beacony Estimote

Zdroj: https://estimote.com/assets/gfx/temp/floating-beacons1.67edff3f.jpg

2.4. Využití v praxi

Při použití v praxi může být systém vnitřní navigace navržen tak, aby kromě benefitů pro uživatele přinášel přidanou hodnotu i pro jeho provozovatele. Např. prodejce si může ukládat informace o pohybu zákazníků po prodejně a na základě těchto dat změnit rozmístění zboží v prostoru.

Podobným způsobem je navržena i aplikace Radio F-Prints. Ta pracuje se systémem, který je složen ze tří[4] částí. Jedná se o beacon, vzdálený server a iOS zařízení, které zpracovává přijatá data a odesílá je na server. Princip komunikace jednotlivých částí je znázorněn níže (***Obrázek 3***)



Obrázek 3: Použité komponenty a jejich vazby

Zdroj: vlastní

- beacon: Bluetooth vysílač, který šíří do prostoru signál zachytitelný ostatními zařízeními.
- serverová část: Zde se jedná o server, se kterým komunikuje iOS zařízení. Server mu poskytuje mapové podklady, konfigurační soubory a je s ním také synchronizována databáze.
- iOS zařízení: Hardware na kterém je nainstalována aplikace Radio F-Prints. Aplikace sbírá data z BLE a dalších zařízení a shromažďuje je v lokální databázi. Na vyžádání uživatele je provedena synchronizace se serverem. iOS zařízení také posílá na server požadavky na získání konkrétních mapových podkladů nebo konfiguračních souborů.

Při návrhu aplikace byla nejprve zvažována možnost použití offline map uložených v zařízení. Tím by byla odbourána nutnost připojení k internetu při měření a zobrazování dat. Po úvaze však bylo implementováno řešení pracující s online

[4] Pro zjednodušení uvažujme, že server slouží k získávání mapových podkladů, konfiguračních souborů a je na něm umístěna databáze, se kterou se zařízení synchronizuje. V praxi se ovšem může jedna o nezávislé části.

mapami a to zejména kvůli své větší variabilitě. Pokud např. dojde ke změně mapy z důvodu jejího zpřesnění nebo opravy chyby, není nutné vydávat aktualizaci aplikace a následně ji distribuovat jednotlivým uživatelům. Stačí pouze nahrát novou verzi mapy na server. Distribuce nové verze aplikace odpadá i v případě přidání nových lokací nebo testovacích scénářů, protože aplikace si po svém spuštění vždy stahuje jejich aktualizovaný seznam. V neposlední řadě se tímto řešením také šetří úložný prostor v zařízení.

3 Aplikace Radio F-Prints

Aplikace Radio F-Prints byla navržena a vyvinuta jako součást této diplomové práce. Slouží ke sběru dat z bezdrátových sítí pro účely pozdějšího využití při kalibraci systému vnitřní navigace v budově Fakulty informatiky a managementu Univerzity Hradec Králové, popř. pro jejich analýzu a další zkoumání. Pro kalibraci jsou podstatná zejména data z BLE zařízení, která jsou v podobě beaconů rozmístěna v budově FIM.

Platforma iOS byla při vývoji tohoto software zvolena pro svojí dostupnost a širokou uživatelskou základnu. Aplikace Radio F-Prints je navržena a vyvinuta pro provoz na běžně dostupných iOS zařízeních. Není tedy třeba vlastnit žádný speciální hardware, nebo provádět jakékoli zásahy do iOS zařízení.

Každé měření je opatřeno časovým razítkem a údajem o aktuální poloze uživatele v budově. Data jsou ukládána lokálně v iOS zařízení a uživatel si může vyžádat jejich synchronizaci se vzdáleným serverem.

3.1. Použité technologie a minimální požadavky

Aplikace Radio F-Prints je nativní univerzální aplikace pro platformu iOS, je tedy možné ji provozovat na zařízeních typu iPhone, iPad a iPod Touch. Aplikace je naprogramována v programovacím jazyce Swift verze 3, ve vývojovém prostředí Xcode 8. Ke svému spuštění vyžaduje iOS 10.1 nebo vyšší. Aplikace ke své instalaci potřebuje minimálně 32 MB volného úložného prostoru v zařízení. Doporučeno je však nejméně 100 MB, z důvodu provádění synchronizace databáze se vzdáleným serverem.

K ukládání dat je použita Couchbase databáze. Jedná se o jednoduchou NoSQL databázi, do které se data ukládají ve formátu JSON. K dispozici je i serverová část, díky které je z pohledu programátora velice jednoduché synchronizovat lokální databázi s daty uloženými na vzdáleném serveru. Couchbase databáze podporuje několik typů autentizace uživatelů (Couchbase, 2017a). V aplikaci Radio F-Prints je použita autentizace pomocí Google účtu, jehož ID je součástí dat ukládaných na univerzitní server.

Při měření je vyžadováno stálé připojení k internetu z důvodu načítání testovacích scénářů a aktuálních map lokací, ve kterých se uživatel pohybuje.

3.2. Měřená data a omezení

Nejdůležitějším zdrojem dat jsou Bluetooth Low Energy zařízení, která se pro svoje pozitivní vlastnosti, jimiž je zejména energetická nenáročnost a s tím spojená dlouhá výdrž baterie, používají pro indoor navigaci nejčastěji. Aplikace je zaměřená na získávání hodnot indikující sílu signálu těchto zařízení.

Další měřené hodnoty jsou spíše doplňkové a slouží ke zpřesnění hodnot naměřených z Bluetooth. Jedná se zejména o dostupné údaje o mobilním operátorovi, informace o WiFi síti, zeměpisné souřadnice (GPS, Glonass), data z gyroskopu, magnetometru a údaje o stavu zařízení, jako je např. úroveň nabití baterie, typu hardware nebo verze operačního systému.

Vzhledem k tomu, že iOS je poměrně uzavřený systém, nedovoluje z bezpečnostních důvodů získávat řadu údajů, které jsou na jiných platformách, jako je např. Android, bez problémů dostupné. Jedná se zejména o seznam WiFi sítí, sílu signálu jednotlivých přístupových bodů, počet GPS družic a data jednotlivých základnových stanic mobilních operátorů. Systém iOS zpřístupňuje některá data pouze zprostředkovaně, pomocí frameworků poskytnutých společností Apple.

U mobilních sítí jsou poskytovány pouze základní údaje o operátorovi, do jehož sítě je iOS zařízení aktuálně připojeno. Stejně tak u WiFi jsou dostupné pouze základní identifikátory připojené sítě. Vývojář je schopen také získat geografickou polohu zařízení v podobě zeměpisných souřadnic. Nemůže ale ovlivnit jakým způsobem iOS zařízení údaje získá. Je možné nastavit pouze požadovanou přesnost, na základě které zařízení samo vyhodnotí, který způsob určení polohy je v dané chvíli nejlepší a to jak z pohledu přesnosti určení polohy, tak z pohledu úspory energie. K dispozici je např. určení polohy pomocí GPS, systému Glonass nebo prostřednictvím dostupných WiFi sítí či buněk mobilního připojení (Apple, 2017a).

Pokud by vývojář chtěl získat přístup k podrobnějším datům, existuje možnost použití neoficiálních frameworků. K jejich fungování je však nutné provést tzv.

jailbreak zařízení, tedy odstranění softwarových zabezpečení operačního systému (iPhone Hacks, 2016). To představuje bezpečnostní riziko a navíc aplikace používající takovéto frameworky nemohou být koncovým uživatelům distribuovány přes oficiální App Store, protože porušují pravidla vydané společností Apple Inc. (Apple, 2017b).

3.3. Funkce a použití aplikace

Aplikace obsahuje tři základní funkce + nastavení parametrů měření. Jedná se o funkce *Měření*, *Přehled* a *Synchronizace*. Všechny funkce vyžadují připojení k internetu, protože při používání funkcí *Měření* a *Přehled* dochází ke stahování mapových podkladů a testovacích scénářů ze vzdáleného serveru. Funkce *Synchronizace* pak přijímá a odesílá data do vzdálené databáze.

Vzhled a chování uživatelského rozhraní je zvoleno tak, aby byly dodrženy zvyklosti ovládání iOS zařízení a uživatelé se uměli v aplikaci intuitivně pohybovat bez nutnosti proškolení (Apple, 2017c). Pro uživatelské rozhraní byly použity komponenty běžně používané v tomto typu aplikací. Pro větší přehlednost každá funkce obsahuje také úvodní obrazovku se stručným popisem a použitím. Uživatelské rozhraní se přizpůsobuje orientaci zařízení. Aplikaci je tedy možné provozovat jak v režimu na výšku, tak na šířku v tzv. landscape módu. Režim na šířku je vhodný zejména pro zobrazení tabulek obsahující delší text, který by se z důvodu nedostatku místa na displeji nemusel při orientaci na výšku zobrazit celý.

Pro přepínání funkcí je použita komponenta Tab Bar Controller. Ta umožňuje přepínání obrazovek pomocí panelu, který je umístěn v dolní části aplikace. Zde se nachází seznam dostupných obrazovek a volitelně také ikony, graficky reprezentující jednotlivé funkce. Přepínání mezi funkcemi je možné pouze na úvodní obrazovce a to z důvodu, aby uživatel přepnutím v nevhodnou dobu nepřerušil měření nebo synchronizaci dat.

Navigace uvnitř funkcí je realizována pomocí komponenty Navigation Controller. Jedná se o komponentu, která do horní části aplikace přidá panel obsahující název

obrazovky, tlačítko pro přechod na předchozí obrazovku[5] a volitelně tlačítko, které je možné použít ke spuštění libovolné funkce.

Aplikace Radio F-Prints obsahuje celkem čtyři komponenty Navigation Controller. Jedna je hlavní, která po spuštění aplikace zajistí zobrazení Tab Bar Controlleru pro přepínání mezi funkcemi. Další tři komponenty Navigation Controller jsou obsažené v jednotlivých funkcích a slouží pro navigaci mezi obrazovkami v rámci dané funkce. Toto řešení má výhodu v tom, že uživatelské rozhraní jednotlivých funkcí je na sobě vzájemně nezávislé. Změna rozvržení uživatelského rozhraní uvnitř jedné funkce se tak nedotkne funkcí ostatních. Přidáním nového panelu do komponenty Tab Bar Controller je také možné velice jednoduše aplikaci rozšířit.

3.3.1. Měření

Jedná se o nejdůležitější funkci celé aplikace, která zajišťuje samotný sběr dat dle parametrů zvolených v nastavení aplikace. Aplikace podporuje dva způsoby měření a to *Měření na místě* a *Procházka*. Níže jsou oba způsoby podrobně popsány. Jak již bylo zmíněno, tato funkce vyžaduje připojení k internetu z důvodu načítání mapových podkladů a testovacích scénářů.

Měření na místě:

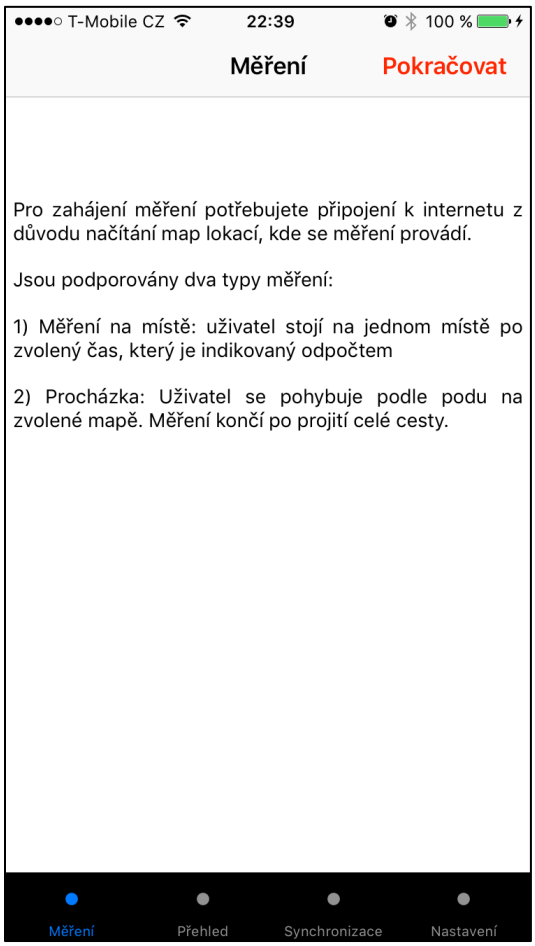
Jedná se o způsob měření, kdy si uživatel zvolí mapu podle lokality, ve které se nachází, na mapě označí svojí aktuální polohu a spustí měření. To trvá řádově desítky sekund a uživatel během něj nemění svojí polohu. Tento způsob měření sice pokryje pouze velmi malou část dané lokace, ale dojde ke sběru více hodnot, než kdyby uživatel danou lokací procházel.

Po zvolení funkce *Měření* na dolní liště je uživateli zobrazena úvodní obrazovka funkce, která obsahuje stručný popis funkce a postup měření (***Obrázek 4***). Je to obrazovka, ze které se uživatel může přepnout do ostatních funkcí nebo nastavení. Jedná se zároveň o obrazovku, která je uživateli zobrazena po spuštění aplikace.

[5] Pokud se uživatel v hierarchii obrazovek nachází na první z obrazovce, tlačítko „Zpět“ není zobrazeno

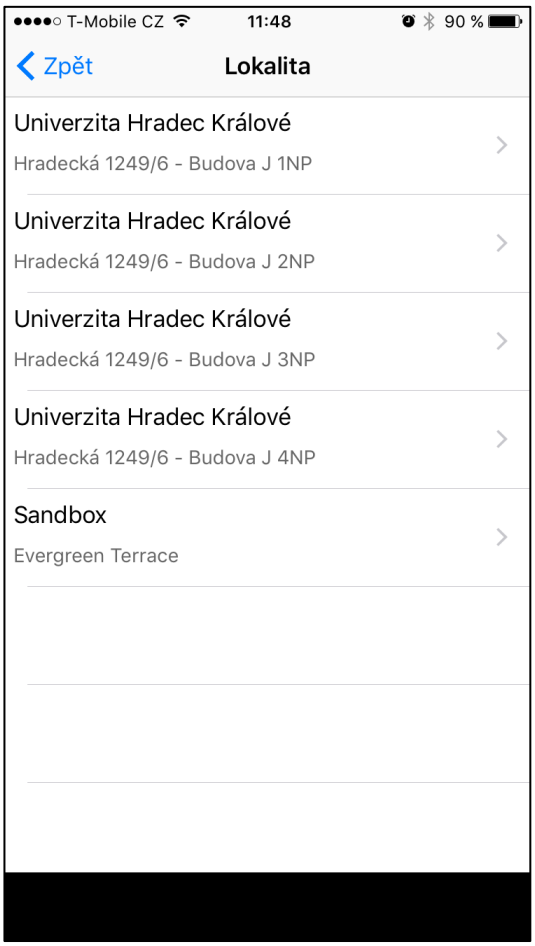
Stiskem tlačítka *Pokračovat* dojde k zobrazení výběru typu měření, kde uživatel zvolí *Měření na místě*. Tlačítko *Pokračovat* je dostupné pouze v případě, že je v nastavení povolen sběr dat z alespoň jednoho typu sítě.

Na další obrazovce se nachází seznam lokalit (***Obrázek 5***), kde je ve formě tabulky zobrazen seznam dostupných mapových podkladů. Každá položka obsahuj název lokality a upřesnění místa, např. číslo popisné a podlaží. Jak je z obrázku patrné, seznam obsahuje také mapu označenou jako *Sandbox*. Jedná se o mapu fiktivní lokace určenou pro testování aplikace.



Obrázek 4: Měření – popis funkce

Zdroj: vlastní, printscreen z mobilní aplikace



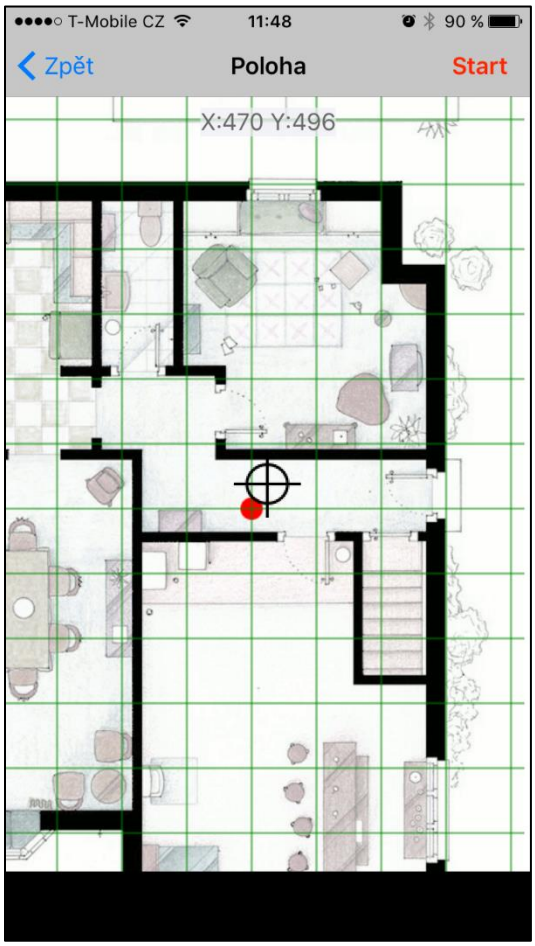
Obrázek 5: Měření – seznam lokalit

Zdroj: vlastní, printscreen z mobilní aplikace

Seznam mapových podkladů je načítán z konfiguračního souboru z internetu vždy po spuštění aplikace, nebo po změně URL adresy konfiguračního souboru

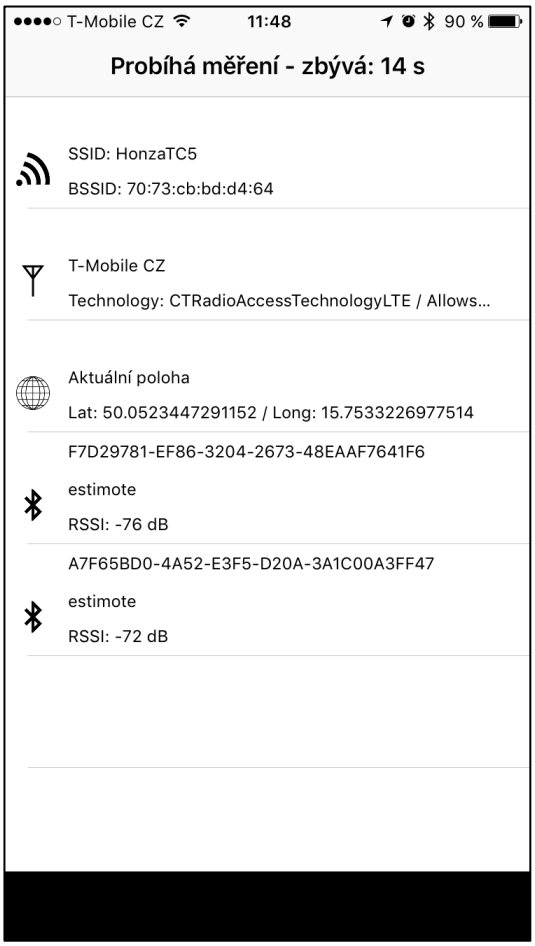
v nastavení. V případě potřeby přidat nebo ubrat lokaci stačí provést úpravu konfiguračního souboru, který je umístěn na serveru.

Výběr lokality provede uživatel klepnutím prstu na příslušný řádek tabulky. Tím dojde k přesunu na obrazovku pro výběr umístění na mapě lokace (***Obrázek 6***). Vzhledem k tomu že je mapa načítána ze vzdáleného serveru, může při změně obrazovky dojít ke krátké prodlevě, jejíž délka je závislá na rychlosti internetového připojení.



Obrázek 6: Měření – mapa lokace

Zdroj: vlastní, printscreen z mobilní aplikace



Obrázek 7: Měření – získávané hodnoty

Zdroj: vlastní, printscreen z mobilní aplikace

Uživatel následně vybere přesnou polohu místa, na kterém je prováděno měření. Výběr polohy se provádí manuálním posouváním mapy. Aktuálně vybraná poloha je indikována červeným bodem na mapě. V horní části obrazovky uprostřed se také nachází popisek, který obsahuje aktuálně vybrané souřadnice X a Y, které budou

přidány k naměřeným hodnotám. Mapu je možné přibližovat a oddalovat gestem rozevření, resp. sevření dvou prstů.

Po dokončení výběru umístění na mapě uživatel spustí samotný sběr dat tlačítkem *Start*, které je umístěné na horní liště vpravo. Tím dojde k přesunu na obrazovku, která zobrazuje průběh měření (***Obrázek 7***).

Ihned po načtení obrazovky dojde k zahájení sběru dat. Uživateli je zobrazena tabulka s aktuálními hodnotami, které zařízení průběžně získává. U dat které se mění v čase, jako jsou zeměpisné souřadnice a síla signálu Bluetooth Low Energy zařízení, je zobrazována poslední získaná hodnota. Do databáze jsou však ukládána všechna data, ke kterým je přidáno časové razítko v podobě celého čísla, indikující počet milisekund od počátku měření.

V horní části obrazovky je zobrazen počet sekund, které zbývají do konce měření. Během probíhajícího měření nemá uživatel možnost tuto obrazovku opustit. Po skončení odpočtu jsou naměřené hodnoty uloženy do lokální databáze a uživatel je přesměrován zpět na obrazovku s mapou (**Obrázek *6***), kde zůstanou vybrány souřadnice, na kterých bylo provedeno poslední měření.

Měření procházka:

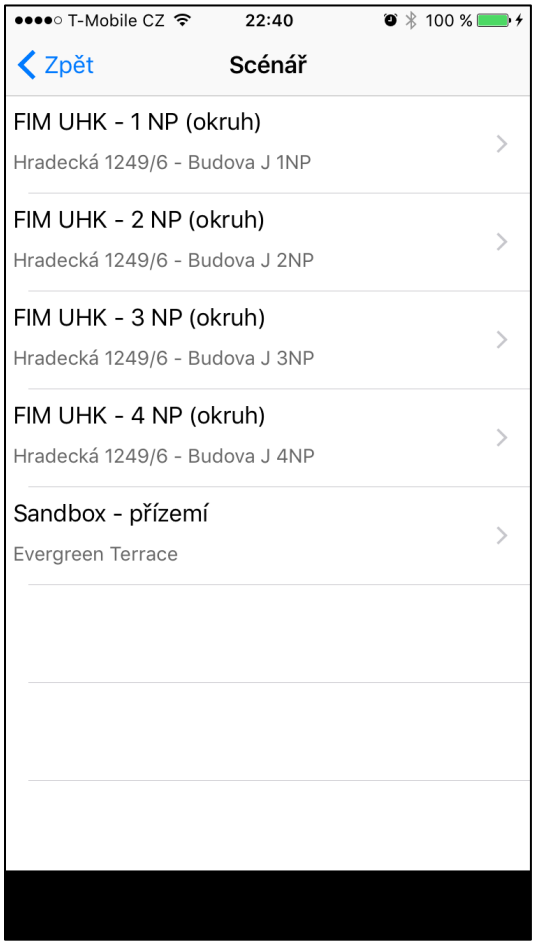
Jak již název tohoto typu měření napovídá, jedná se o měření, kdy uživatel prochází předem definovanou trasu dle vybraného scénáře a aplikace provádí průběžný sběr dat. Oproti funkci *Měření na místě* aplikace získá na konkrétních souřadnicích menší množství dat, protože se na nich zařízení nachází kratší dobu. Na druhou stranu tímto typem měření lze pokrýt mnohem větší plochu lokace, kde je měření prováděno.

Trasa, resp. umístění, kde se má uživatel v danou chvíli nacházet, je zobrazeno na mapě. Uživatel se přemisťuje podle pohybujícího se bodu na mapě a aplikace průběžně sbírá data. Ty jsou každé 2 sekundy uloženy jako samostatné měření. Tím ji zachována stejná struktura dat jako u *Měření na místě*. V přehledu je pak toto měření zobrazeno jako sled po sobě jdoucích jednotlivých měření. Rychlost pohybu

bodu na mapě je možné změnit v nastavení tak, aby odpovídal rychlosti běžné chůze uživatele.

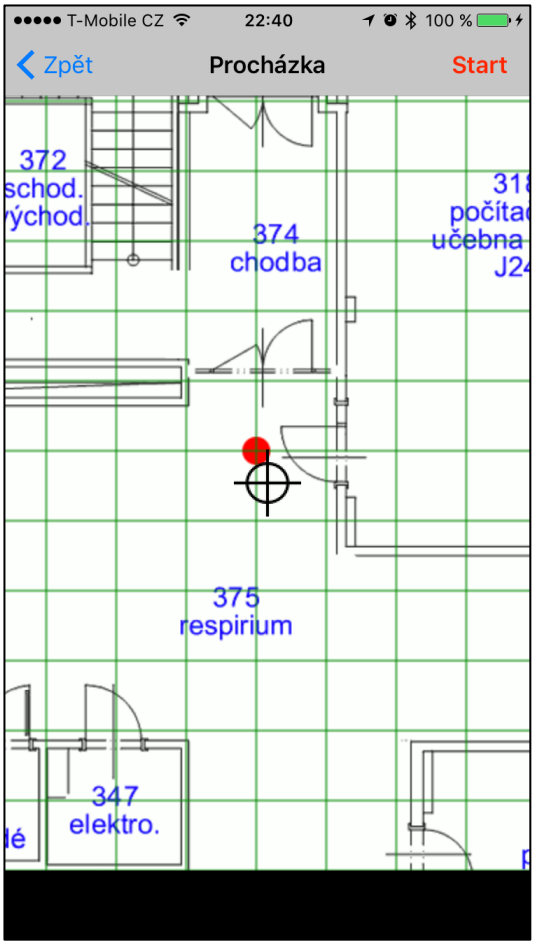
Stejně jako u *Měření na místě* se uživatel tlačítkem *Pokračovat* dostane z úvodní obrazovky na výběr typu měření, kde zvolí *Procházka*. Následně dojde k zobrazení seznamu aktuálně dostupných scénářů (***Obrázek 10***).

Každá položka ze seznamu obsahuje název scénáře a upřesnění lokality, které se scénář týká. Výběr požadovaného scénáře se provede klepnutím prstu na příslušný řádek tabulky. Tím dojde k zobrazení obrazovky s mapou a zároveň je zde možné zahájit měření (**Obrázek 9**).



Obrázek 8: Měření - seznam scénářů

Zdroj: vlastní, printscreen z mobilní aplikace



Obrázek 9: Měření - navigování uživatele

Zdroj: vlastní, printscreen z mobilní aplikace

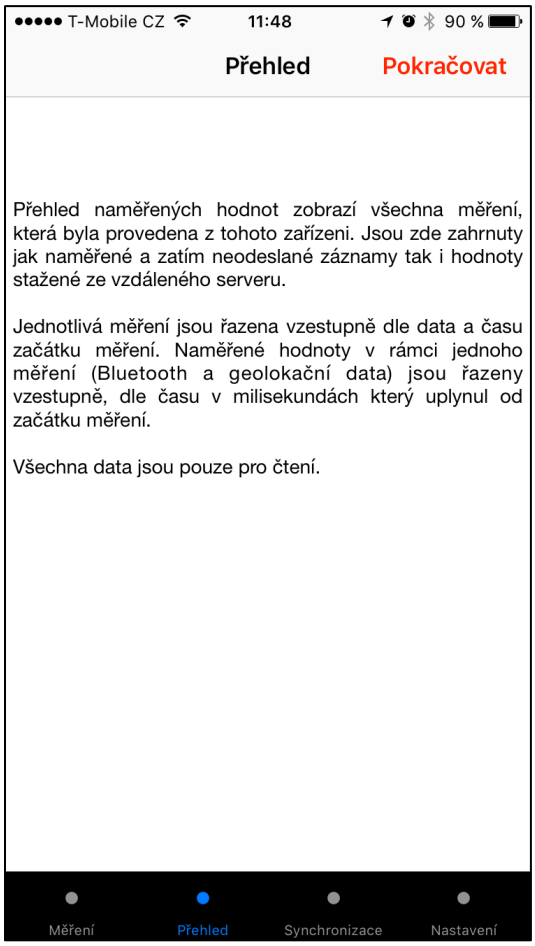
Po načtení mapy je zobrazena poloha, na které by se měl uživatel nacházet na začátku měření. Pokud tomu tak je, může být stisknutím tlačítka *Start* měření zahájeno. Bod na mapě se začne pohybovat dle souřadnic, které jsou zadány ve scénáři. Tlačítko *Start* je nahrazeno tlačítkem *Zrušit měření*, jehož stiskem je možné sběr dat kdykoli zastavit a přesunout se na úvodní obrazovku funkce. V tom případě nejsou naměřená data akceptována a uložena. Aby naměřené údaje byly relevantní, je důležité, aby pohyb uživatele odpovídal pohybu bodu na mapě. Po projití celé trasy je uživateli zobrazeno upozornění o úspěšném dokončení měření a zároveň dojde k uložení dat do lokální databáze.

3.3.2. Přehled

Záložka *Přehled* slouží k zobrazení dat z předešlých měření. Uživatel zde najde seznam všech měření, která byla provedena na zařízeních s operačním systémem iOS a která jsou uložena v lokální databázi. Jsou zde tedy data získaná při synchronizaci databáze i hodnoty, které ještě nebyly odeslány na server.

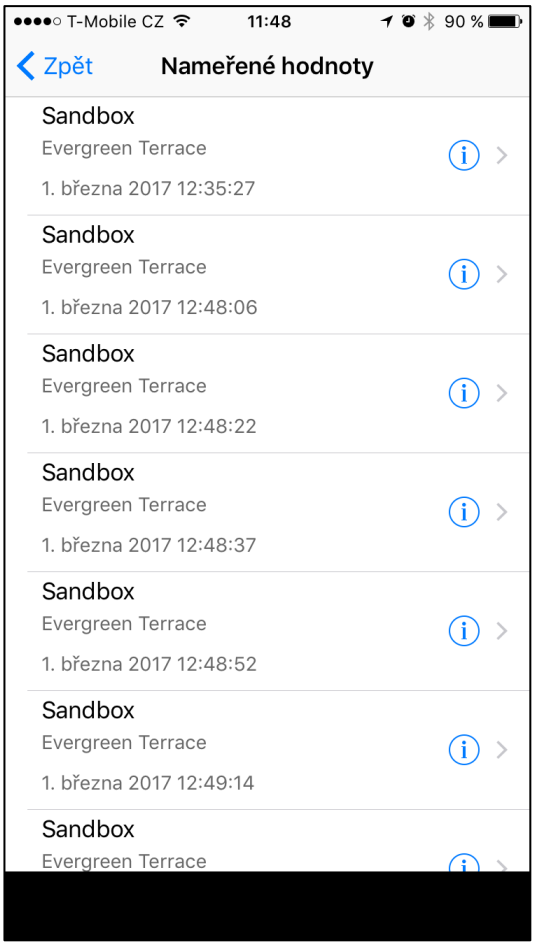
Po zvolení funkce *Přehled* na dolní liště je zobrazena její úvodní obrazovka. Zde uživatel najde její stručný popis použití (***Obrázek 10***). Tlačítkem *Pokračovat* dojde k přesunu na tabulku obsahující seznam naměřených hodnot (***Obrázek 11***). Pokud lokální databáze neobsahuje žádné naměřené hodnoty, dojde k zobrazení upozornění s textem: „Nejsou k dispozici žádná data k zobrazení. Nejprve synchronizujte databázi nebo proveďte měření“ a uživateli zůstane zobrazena obrazovka s popisem funkce.

Tabulka je seřazena sestupně podle času, kdy bylo měření provedeno. Každý záznam obsahuje název lokality, upřesnění místa (většinou se jedná o adresu a podlaží budovy) a datum a čas měření. Uživatel zde má možnost vymazat záznamy, které zatím nebyly synchronizovány se vzdálenou databází. Vymazání se provádí přejetím prstu zprava doleva na vymazávaném řádku a následným stiskem tlačítka *Delete*. Zobrazení hodnot z konkrétního měření je provedeno klepnutím na příslušný řádek. Tím dojde k zobrazení přehledu naměřených hodnot ve formě tabulky (***Obrázek 12***).



Obrázek 10: Přehled – popis funkce

Zdroj: vlastní, printscreen z mobilní aplikace



Obrázek 11: Přehled – seznam měření

Zdroj: vlastní, printscreen z mobilní aplikace

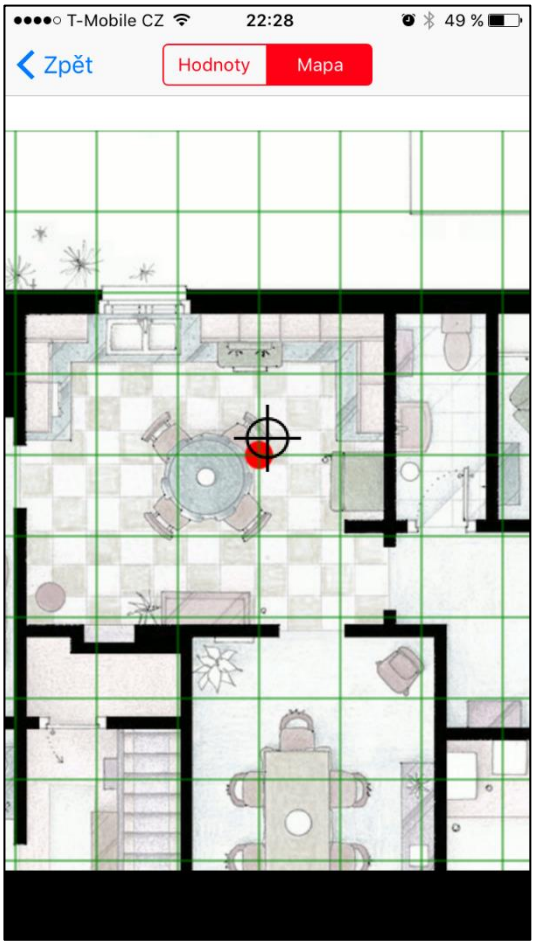
Přehled hodnot je rozdělen do pěti logických celků:

- Informace o měření
- Mobilní data
- WiFi scan
- Bluetooth scan
- Geolokační data

V horní části obrazovky uprostřed je přepínač, pomocí kterého si uživatel vybírá, zda chce zobrazit hodnoty, nebo mapu s označením místa, kde bylo měření provedeno (***Obrázek 13***). Zobrazenou mapu je možné přibližovat a oddalovat pomocí gesta rozevření, resp. sevření dvou prstů, stejně jako v případě výběru lokace při měření.

Obrázek 12: Přehled – naměřené hodnoty

Zdroj: vlastní, printscreen z mobilní aplikace



Obrázek 13: Přehled – lokalita

Zdroj: vlastní, printscreen z mobilní aplikace

Všechny zobrazené údaje jsou pouze pro čtení, aby nedocházelo ke zkreslování naměřených hodnot. Seznam všech hodnot, které jsou zobrazovány uživateli, je uveden v tabulce 1.

Název hodnoty	Popis
ID uživatele	Google ID kterým uživatel podepsal data při synchronizaci. U dat, která nebyla odeslána na server, není tento údaj vyplněný.
Lokalita	Název lokality kde bylo prováděno měření
Upřesnění místa	Upřesnění místa (většinou adresa a číslo podlaží)
ID mapy	Interní ID mapy (většinou je stejné jako lokalita)
Souřadnice na mapě	Souřadnice X a Y udávající přesné místo, kde bylo měření prováděno
Začátek měření	Datum a čas spuštění měření
Konec měření	Datum a čas ukončení měření
ID zařízení	ID zařízení, na kterém bylo měření prováděno
Hardware	Název hardware
Operační systém	Název a verze operačního systému
Stav baterie	Stav nabití baterie v době měření v intervalu 0.0 až 1.0 (úplně vybitá až úplně nabitá)
Napájecí režim	Napájecí režim v době měření unplugged – napájení z baterie charging – probíhá nabíjení baterie full – zařízení je připojeno ke zdroji a baterie je zcela nabita
Gyroskop	Hodnoty X, Y, Z získané z gyroskopu na konci měření
Magnetometr	Hodnoty X, Y, Z získané z magnetometru na konci měření
Sběr Bluetooth dat	Indikuje, zda byla během měření shromažďována data z dostupných BLE zařízení (zaškrtnuto)
Sběr WiFi dat	Indikuje, zda byla během měření shromažďována data z aktuálně připojené WiFi sítě (zaškrtnuto)
Sběr mobilních dat	Indikuje, zda byla během měření shromažďována data z mobilní sítě (zaškrtnuto)

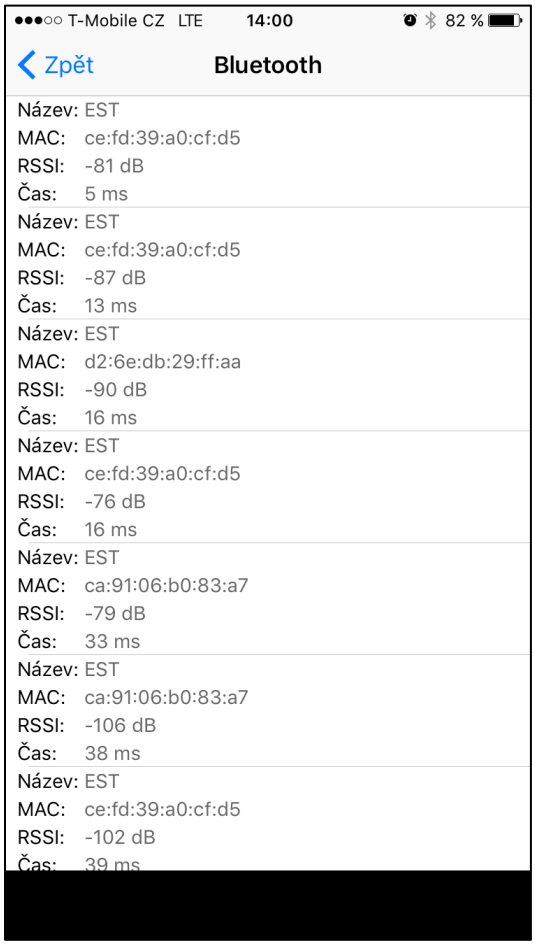
Sběr geolokačních dat	Indikuje, zda byla během měření shromažďována geolokační data (zaškrtnuto)
technologie	Název podporované mobilní technologie
Poskytovatel	Název domovského mobilního operátora uživatele
ISO	ISO kód země
MCC	Mobile country code – identifikátor země
MNC	Mobile network code – identifikátor mobilního operátora v dané zemi
Povoleno VOIP	Indikuje, zda mobilní operátor povoluje VoIP hovory
SSID	Název WiFi sítě
BSSID	MAC adresa přístupového bodu
Bluetooth data	Seznam naměřených hodnot dostupných BLE zařízení. Každý záznam obsahuje název zařízení, MAC adresu, naměřenou hodnotu a počet milisekund, který uplynul od začátku měření. Záznamy jsou seřazeny vzestupně podle počtu milisekund.
Geolokační data	Seznam naměřených zeměpisných souřadnic. Každý záznam obsahuje hodnoty longitude, latitude a počet milisekund, který uplynul od začátku měření. Záznamy jsou seřazeny vzestupně podle počtu milisekund.

Tabulka 1: Seznam naměřených hodnot

Zdroj: vlastní

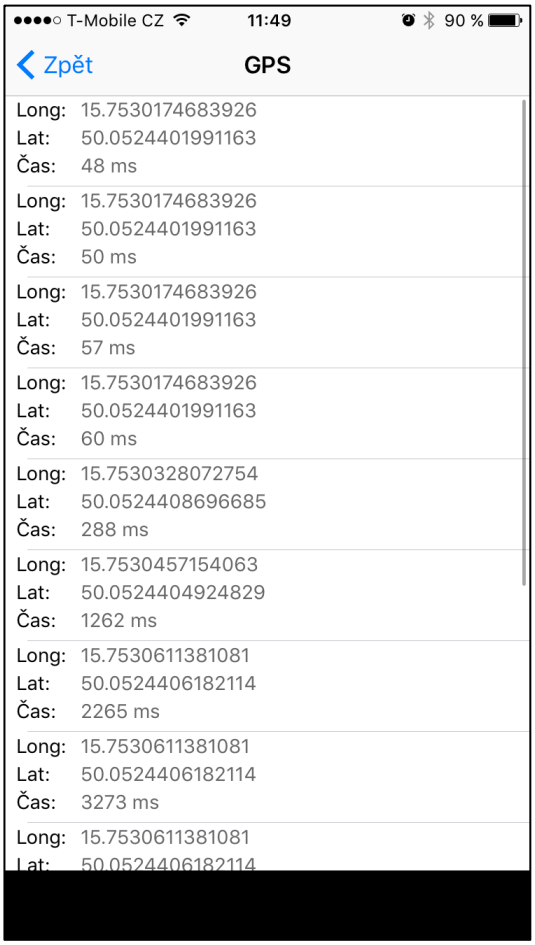
Zobrazení tabulky se seznamem hodnot získaných z dostupných BLE zařízení (***Obrázek 14***) proběhne klepnutím na řádek *Naměřené hodnoty* v sekci *BLUETOOTH SCAN*. Každý záznam obsahuje název a MAC adresu zařízení, sílu přijatého signálu a počet milisekund, který uplynul od začátku měření do doby, kdy byla hodnota získána. Data jsou seřazena vzestupně podle počtu milisekund.

Přehled přijatých zeměpisných souřadnic (***Obrázek 15***) je dostupný z hlavního seznamu hodnot, klepnutím na řádek *Naměřené hodnoty* v sekci *GEOLOKAČNÍ DATA*. Zde je opět tabulka seřazena vzestupně podle času přijetí dat. Je zde zobrazena zeměpisná délka *Long*, zeměpisná šířka *Lat* a počet milisekund uplynulý od začátku měření do přijetí daného záznamu.



Obrázek 14: Přehled – Bluetooth hodnoty

Zdroj: vlastní, printscreen z mobilní aplikace



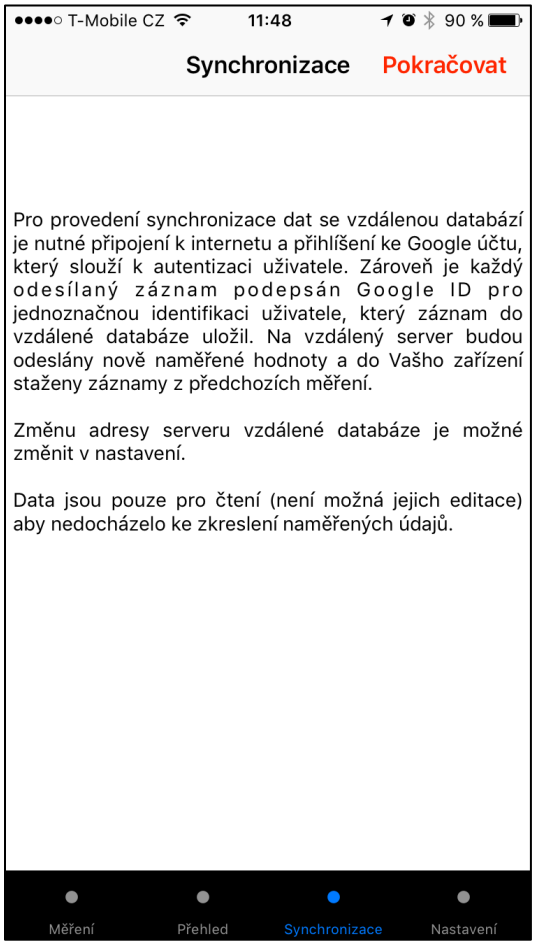
Obrázek 15: Přehled – GPS hodnoty

Zdroj: vlastní, printscreen z mobilní aplikace

3.3.3. Synchronizace

Pomocí této funkce může uživatel po předchozí autentizaci pomocí Google účtu provést synchronizaci databáze se vzdáleným serverem. Jak již bylo zmíněno, tato funkce vyžaduje připojení k internetu. V závislosti na rychlosti připojení a množství odesílaných a přijímaných dat může přenos trvat delší dobu.

Synchronizace je obousměrná, tedy z iOS zařízení jsou odesílána nově naměřená data a naopak jsou ze serveru stahovány veškeré záznamy, které zatím nejsou v lokální databázi. Synchronizovaná databáze obsahuje všechny záznamy, které jsou uložené na vzdáleném serveru. Filtrování dat probíhá až před jejich zobrazením uživateli, kterému jsou zpřístupněna pouze měření, které byly provedeny na zařízení s operačním systémem iOS. V případě požadavku na změnu chování filtru, např. aby místo operačního systému docházelo k filtrování podle Google ID, je možné parametry filtru změnit jednoduchou úpravou nastavené podmínky.



Obrázek 16: Synchronizace – popis funkce

Zdroj: vlastní, printscreen z mobilní aplikace

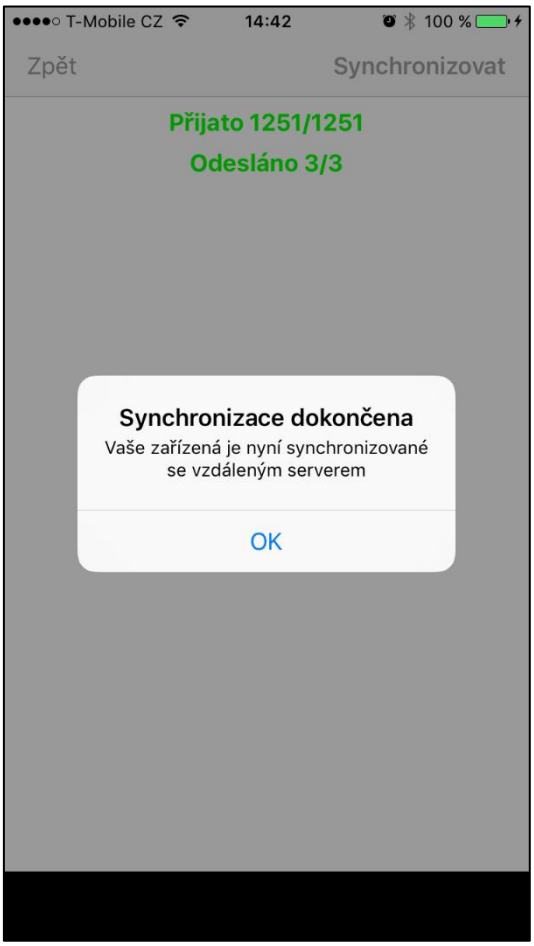
Obrázek 17: Synchronizace – login

Zdroj: vlastní, printscreen z mobilní aplikace

První obrazovka je opět úvodní obrazovka funkce (***Obrázek 16***), obsahující její popis, možnosti a omezení. Na tuto obrazovku je uživatel navigován při zvolení

Synchronizace na dolní liště. Tlačítkem *Pokračovat* dojde k přesunu na obrazovku, kde je zobrazena stránka pro přihlášení ke Google účtu. (***Obrázek 17***),

Po úspěšné autentizaci uživatele si aplikace uloží jeho Google ID a je zobrazena poslední obrazovka (***Obrázek 18***), kde je možné zahájit samotnou synchronizaci dat. Pokud je uživatel ke Google účtu již přihlášen z předchozí synchronizace a aplikace je schopna získat jeho Google ID, přihlašovací stránka není uživateli zobrazena a místo toho je uživatel přesměrován přímo na poslední obrazovku.



Obrázek 18: Synchronizace – přenos dat

Zdroj: vlastní, printscreen z mobilní aplikace

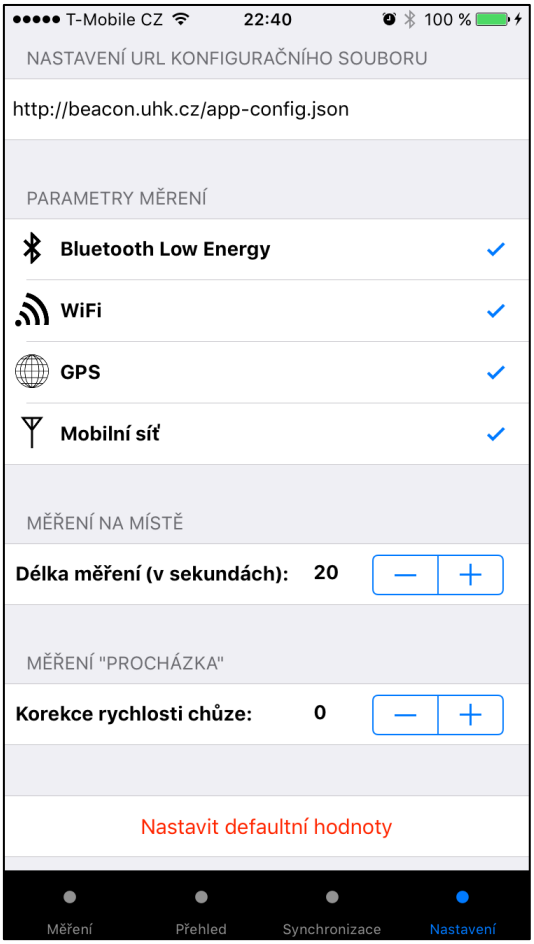
Synchronizace dat se spouští tlačítkem *Synchronizovat* vpravo nahoře. V závislosti na rychlosti internetového připojení a množství odesílaných a přijímaných dat může trvat delší dobu (řádově nižší jednotky minut) než je celý proces dokončen.

Zpravidla nejdéle trvá první synchronizace po nainstalování aplikace do zařízení, protože jsou obousměrně přenášeny všechna data. Následující synchronizace trvají výrazně kratší dobu, protože jsou přenášeny pouze nové nebo změněné záznamy.

Uživatel zde může sledovat, jak synchronizace postupuje. Je mu zobrazen počet přenesených záznamů a celkové množství dat, které se bude synchronizovat. Po úspěšném dokončení synchronizace je zobrazeno upozornění s textem „Vaše zařízení je nyní synchronizované se vzdáleným serverem“.

3.3.4. Nastavení

Na panelu *Nastavení* může uživatel změnit některé parametry měření a URL adresu, ze které je načítán konfigurační soubor.



Obrázek 19: Nastavení aplikace

Zdroj: vlastní, printscreen z mobilní aplikace

Na obrázku je znázorněna obrazovka nastavení s výchozími hodnotami, která je členěna do pěti logických celků (***Obrázek 19***). Pokud zde uživatel provede nějakou změnu, nová hodnota je uložena do trvalého uložiště aplikace.

Nastavení obsahuje pět sekcí:

- nastavení URL konfiguračního souboru
- parametry měření (společné pro *Měření na místě* a měření *Procházka*)
- parametry pro *Měření na místě*
- parametry pro měření *Procházka*
- nastavení výchozích hodnot

V tabulce 2 je uveden výčet atributů, jejichž hodnoty může uživatel editovat, jejich popis a také výchozí hodnoty.

URL adresa konfiguračního souboru je zadávána formou textu. Pokud dojde ke změně této hodnoty, aplikace znovu načte konfigurační soubor z internetu.

Pro změnu délky měření a korekci rychlosti chůze byla zvolena komponenta Stepper. Ta umožňuje vývojáři nastavit povolený interval hodnot a je uživatelsky přívětivá zejména v případě, kdy je třeba nastavit výrazně jinou hodnotu, než je ta aktuální. Uživateli stačí položit prst na tlačítko + resp. – a počkat, než dojde ke změně hodnoty na požadovanou úroveň. Čím déle je příslušné tlačítko stisknuto, tím rychleji změna hodnot probíhá.

Nastavení položek *Bluetooth Low Energy*, *WiFi*, *GPS* a *Mobilní síť* je indikováno modrým zaškrtnutím v pravé části řádku v případě, že je daná položka zahrnuta do měření. Uživatel si může zvolit libovolnou kombinaci sítí, pokud však nevybere ani jednu, není mu umožněno zahájit měření (tlačítko *Pokračovat* na úvodní obrazovce funkce *Měření* bude neaktivní).

Název	Výchozí hodnota	Popis
URL konfiguračního souboru	http://beacon.uhk.cz/app-config.json	URL adresa, ze které je získáván konfigurační soubor
Bluetooth Low Energy	True	Indikuje, zda jsou při měření sbírána data z dostupných BLE zařízení
WiFi	True	Indikuje, zda jsou při měření sbírána data z aktuálně připojené WiFi
GPS	True	Indikuje, zda jsou při měření sbírána polohová data z GPS modul
Mobilní síť	True	Indikuje, zda jsou při měření sbírána data z mobilní sítě
Délka měření	20	Celková délka měření v sekundách. Rozmezí od 5 do 120 sekund (pouze při *Měření na místě*)
Korekce rychlosti chůze	0	Zpomalení (záporné hodnoty) nebo zrychlení (kladné hodnoty) pohybu bodu na mapě. Rozmezí od -10 do +10 (pouze měření *Procházka*)
Nastavit defaultní hodnoty	-	Tlačítko pro návrat k výchozímu nastavení

Tabulka 2: Nastavení – editovatelné parametry

Zdroj: vlastní

4 Vlastní implementace a zdrojový kód

Aplikace Radio F-Prints ke svému chodu většinou používá originální frameworky poskytované společností Apple Inc. které jsou součástí vývojového prostředí Xcode, ve kterém je aplikace naprogramovaná. Výjimkou je pouze framework pro práci s Couchbase databází. Vývoj aplikace, rozvržení uživatelského rozhraní a použití externího frameworku probíhalo tak, aby výsledný produkt splňoval podmínky společnosti Apple Inc. pro publikování aplikací v online obchodu App Store, platné v době psaní této práce. (Apple, 2017b). Aplikace tak splňuje všechny základní podmínky nutné pro úspěšné absolvování schvalovacího procesu a může být do App Store umístěna.

4.1. Použité frameworky

V kontextu vývoje aplikací pro platformy iOS a Mac OS X je jako framework označována *„Knihovna tříd, funkcí, protokolů, dokumentace a hlavičkových souborů i jiných prostředků, jež vzájemně souvisejí“* (Kochan, 2010, s. 473). Tato knihovna je znovupoužitelná napříč aplikacemi. Framework navíc může obsahovat více různých verzí jedné knihovny.

Velká škála frameworků je poskytnuta přímo společností Apple Inc. a je možné je zdarma využít při vývoji aplikací. U těchto frameworků má vývojář jistotu, že obsažené knihovny byly řádně otestovány a že budou správně a efektivně fungovat na všech zařízeních, jejichž operační systém daný framework obsahuje. Navíc nastavením minimální podporované verze iOS v prostředí Xcode dojde k výběru takové verze knihovny, která je kompatibilní se všemi zařízeními, které chce vývojář podporovat.

Při vývoji aplikací je možné také použití vlastních frameworků, do kterých vývojář implementuje často používané funkce, popř. lze z internetu stáhnout řadu externích frameworků, které rozšiřují nebo zjednodušují nějaký druh často řešeného problému.

Při použití frameworků třetích stran je však na místě ostražitost, protože zde vývojář může narazit na řadu problémů a omezení, které ve svém výsledku mohou znemožnit publikování aplikace na oficiálním App Store.

Dříve než se vývojář rozhodne některý z externích frameworků použít, je nutné zjistit jakým způsobem je licencovaný a jaké jsou podmínky jeho použití. Dále by se měl zajímat také o to, zda k frameworku existuje přehledná dokumentace. Zejména na začátku vývoje, kdy programátor ještě není seznámen se strukturou knihoven a použitím funkcí daného frameworku, může prostudování dokumentace pomoci urychlit vývoj a vyhnout se chybám, jejichž náprava by později mohla být časově velmi náročná. Je také nutné zjistit jakou minimální verzi iOS daný framework podporuje a zda toto zjištění koresponduje s nejstarší verzí iOS, kterou chce podporovat vývojář

Je dobré používat pouze ověřené frameworky získané z důvěryhodných zdrojů. V opačném případě je zde riziko, že knihovny mohou obsahovat nějaké skryté funkce s nekalým účelem. Ty se např. mohou pokoušet bez vědomí vývojáře a uživatelů získávat citlivá data ze zařízení a odesílat je na vzdálený server. Tento druh infiltrace je zpravidla odhalen při testování v rámci schvalovacího procesu aplikace ve společnosti Apple Inc. Nicméně kromě znemožnění umístění aplikace do App Store může v krajním případě dojít k zablokování účtu vývojáře se všemi dopady. Navíc mohou být napadeny zařízení jak samotného vývojáře, tak i testerů aplikace.

Níže se nachází výčet a stručný popis frameworků a knihoven použitých v aplikaci Radio F-Prints.

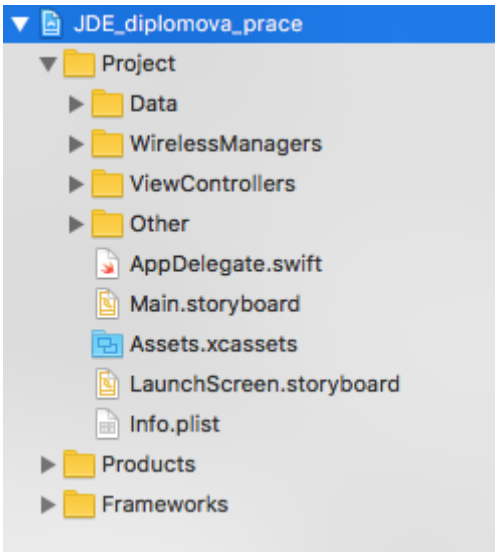
- *CFNetwork.framework*: umožňuje pracovat s WiFi, mobilní sítí a také poskytuje možnost komunikace přes HTTP, HTTPS a FTP protokoly. V aplikaci Radio F-Prints je použit pro zjištění identifikátoru připojené WiFi sítě a MAC adresy přístupového bodu. Tento framework je také interně používán CouchbaseLite frameworkem při synchronizaci dat.
- *CoreBluetooth.framework*: umožňuje komunikovat s BLE zařízeními, které jsou v dosahu. Pomocí něj je možné skenovat zařízení, posílat jim příkazy a vyžádat si získávání notifikací o změnách stavu zařízení. V aplikaci je framework využit pro získávání síly signálu RSSI.

- *CoreLocation.framework*: prostřednictvím něj lze získat aktuální polohu v podobě geografických souřadnic. Tento framework umožňuje také zjistit vzdálenost od konkrétního BLE zařízení, které pracuje se vhodným profilem. Pro potřeby aplikace je však využit pouze pro zjišťování zeměpisné šířky a délky. Pro skenování BLE je využit CoreBluetooth.framework.
- *CoreTelephony.framework*: poskytuje rozhraní pro přístup k datům souvisejícím s mobilní sítí. Aplikace využívá tento framework pro zjišťování informací o mobilní operátorovi.
- *CouchbaseLite.framework*: zprostředkovává objektový přístup ke Couchbase databázi a umožňuje synchronizaci dat.
- *libc++.tbd*: jedná se o standardní C++ knihovnu, která je interně používána CouchbaseLite frameworkem.
- *libsqlite3.tbd*: SQL lite je relační databáze, která je umístěna přímo v aplikaci a která nemusí být provozována jako samostatná služba. Tato knihovna je interně používán CouchbaseLite frameworkem.
- *libz.tbd*: knihovna obsahující funkce pro kompresi dat. Tato knihovna je interně používán CouchbaseLite frameworkem.
- *Security.framework*: umožňuje spravovat certifikáty, veřejné a privátní klíče a také zprostředkovává přístup ke klíčence (keychain), do které mohou být citlivá data ukládána. Tento framework je interně používán CouchbaseLite frameworkem při synchronizaci dat.
- *SystemConfiguration.framework*: poskytuje rozhraní pro přístup ke konfiguraci sítě. Tento framework je interně používán CouchbaseLite frameworkem při synchronizaci dat.

(Apple, 2017e-j)

4.2. Členění zdrojového kódu

Jednotlivé soubory obsahující zdrojový kód jsou pro lepší orientaci členěny do čtyř skupin, podle typu úlohy, kterou v aplikaci zastávají (**Obrázek *20***). Přitom platí, že jeden soubor = jedna třída. Rozdělení zdrojových souborů je viditelné pouze v prostředí Xcode, protože se jedná o členění projektu, nikoli o uložení do samostatných složek na disku.



Obrázek 20: Členění zdrojového kódu

Zdroj: vlastní, výstřižek z vývojového prostředí

Zdrojový kód a všechny automaticky vygenerované soubory nutné k sestavení aplikace se nacházejí v podsložce *Project*. Kromě toho v kořenové složce projektu jsou další dva automaticky vygenerované adresáře. Prvním z nich je *Products*, ve které najdeme po úspěšném sestavení projektu spustitelnou aplikaci. Druhým adresářem je *Frameworks*, ve kterém se nacházejí výše zmíněné frameworky použité v aplikaci. Dále se budeme zabývat pouze obsahem složky *Project*.

- podsložka *Data* obsahuje třídy, které přímo manipulují s daty z databáze nebo generují body při měření *Procházka*. Dále jsou zde třídy, které reprezentují data v objektové podobě. Je zde také umístěna singleton třída

sloužící k načítání a ukládání uživatelem editovatelnému nastavení aplikace a načítání konfiguračního souboru.

- podsložka *WirelessManagers* obsahuje třídy, které zapouzdřují přístup k funkcím mající na starosti získávání dat z WiFi, BLE, GPS a mobilní sítě. Je zde také soubor obsahující delegáta, kterého všechny třídy v této složce implementují z důvodu sjednocení přijímání a odesílání zpráv o přijetí nových dat. Delegát je *„Objekt, na který jiný objekt deleguje část své funkčnosti. Objekt komunikuje se svým delegátem pomocí předem deklarovaných zpráv a informuje jej tak například o svých změnách“.* (Kochan, 2010, s. 470).

- podsložka *ViewControllers* obsahuje třídy obsluhující jednotlivé obrazovky. Hlavním úkolem těchto tříd je předávat si vzájemně data v závislosti na akcích prováděných uživatelem. Pokud daná obrazovka obsahuje tabulku, tyto třídy implementují delegáty, pomocí kterých se tabulky plní daty. Zajišťují také navigaci napříč aplikací. Jednotlivé třídy jsou pojmenované podle funkcí, se kterými jsou spjaty. Jsou to *SettingsViewController* pro obrazovku nastavení, *SynchronizeViewController* pro funkci synchronizace, *ShowDataViewController* pro funkci přehled a *DataCollectionViewController* pro funkci měření. S výjimkou nastavení obsahuje název třídy číslo, podle pořadí obrazovky v rámci dané funkce. U některých tříd následuje ještě upřesňující název.

- podsložka *Other* sdružuje zejména prototypy buněk jednotlivých tabulek. Názvy těchto tříd začínají *PrototypeCell* a následuje upřesňující název, aby bylo zřejmé ve kterých tabulkách je daný prototyp buňky použitý. Dále je zde umístěný tzv. Bridging header. Jedná se o hlavičkový soubor Objective-C, který umožňuje volání Objective-C funkcí Couchbase databáze ze tříd naprogramovaných ve Swiftu. Jako poslední je zde umístěn předpis protokolu *ProtocolWirelessManager*, který implementují jednotlivé datové třídy ze složky Data. Protokol je *„Seznam metod, které musí nějaká třída implementovat, aby naplnila neboli přijala daný protokol. Protokoly umožňují standardizovat rozhraní mezi třídami.“* (Kochan, 2010, s. 473).

Jak již bylo zmíněno, ve složce *Projects* se nacházejí také automaticky vygenerované soubory nezbytné pro úspěšné sestavení aplikace.

Soubor *AppDelegate.swift* je zdrojový soubor implementující delegáta *UIApplicationDelegate*. Pomocí něj je aplikace informována o změně svého stavu, na kterou může vývojář reagovat. Změnou stavu se rozumí např. zapnutí nebo vypnutí aplikace, přesun aplikace na pozadí nebo vyvolání aplikace do popředí. Na základě těchto notifikací může být např. zastavena nějaká probíhající akce, načteno nastavení atd. Pro potřeby aplikace Radio F-Prints je využita funkce *didFinishLaunchingWithOptions* která indikuje, že aplikace byla načtena do paměti a je připravena ke spuštění. Ve chvíli kdy je tato funkce zavolána, aplikace si vyžádá povolení k monitorování stavu baterie, který je během měření přidán k naměřeným hodnotám jako podpůrný údaj.

Soubor *Main.storyboard* slouží pro návrh uživatelského rozhraní. Podrobnější informace jsou uvedeny v následující kapitole.

Assets.xcassets je speciální typ složky, ve které jsou umístěny obrázky a ikony použité v aplikaci. Použití této složky je oproti přímému odkazování na uložené obrázky a ikony vhodnější zejména proto, že podle typu podporovaných zařízení prostředí Xcode vývojáři radí, jaké rozměry ikon a obrázků je vhodné použití. Při jejich následném použití v aplikaci je automaticky vybrán obrázek nebo ikona nejvhodnější velikosti pro dané zařízení.

LaunchScreen.storyboard je stejně jako *Main.storyboard* určen pro návrh uživatelského rozhraní. V tomto případě se však jedná o tzv. loading screen, tedy obrazovku která je zobrazena uživateli, zatímco se aplikace načítá do paměti. Z tohoto důvodu je obrazovka pouze statická a uživateli není povolena žádná interakce. Často je zobrazován statický obrázek v podobě loga aplikace nebo jejího výrobce. V aplikaci Radio F-Prints není *LaunchScreen.storyboard* využit, resp. je ponechán prázdný.

Soubor *Info.plist* je konfigurační soubor obsahující údaje nutné k sestavení aplikace. Data jsou ve formátu klíč – hodnota, popř. klíč – pole hodnot. Je zde uložen např. název aplikace, podporované orientace zařízení, informace že aplikace potřebuje používat GPS pro určení polohy atd.

4.3. Uživatelské rozhraní

Jak již bylo zmíněno, údaje o rozložení uživatelského rozhraní jsou uloženy v souboru *Main.storyboard*. Jedná se o xml soubor, který dokáže vývojové prostředí Xcode po otevření převést do grafické podoby uživatelského rozhraní. Vývojář tak nemusí do souboru ručně zasahovat a může uživatelské rozhraní plně navrhnout v nástroji Interface Builder, který je součástí vývojového prostředí Xcode. Interface Builder dříve býval samostatnou aplikací a má poměrně dlouhou historii. Používat se začal již v roce 1988 a byl použit při návrhu aplikací pro NextSTEP, OpenSTEP a nyní také pro Mac OS X a iOS (Mark a LaMarche, 2010, s. 33). Do Xcode je Interface Builder plně integrován od verze 4 (Apple, 2017d).

Kromě rozložení jednotlivých komponent na obrazovkách je zde možné také vytvářet vazby mezi obrazovkami samotnými. Komponenta, která tuto funkci zajišťuje, se nazývá Storyboard Segue. Každé vizuální komponentě použité v Interface Builderu je možné přiřadit vlastní třídu, které dědí od základní třídy dané komponenty. To umožňuje vývojáři rozšířit nebo pozměnit její standardní chování. Je zde také možné definovat, kde budou odchytávány akce, jako je např. stisk tlačítka, provedení gesta atd.

Interface Builder navíc umožňuje vývojáři zobrazit náhled uživatelského rozhraní a jeho rozložení na všech podporovaných zařízeních při různých orientacích displeje. V omezené míře dokáže také navrhovat vhodné změny v rozložení tak, aby po spuštění aplikace vypadalo její uživatelské rozhraní stejně, jako návrh v Interface Builderu.

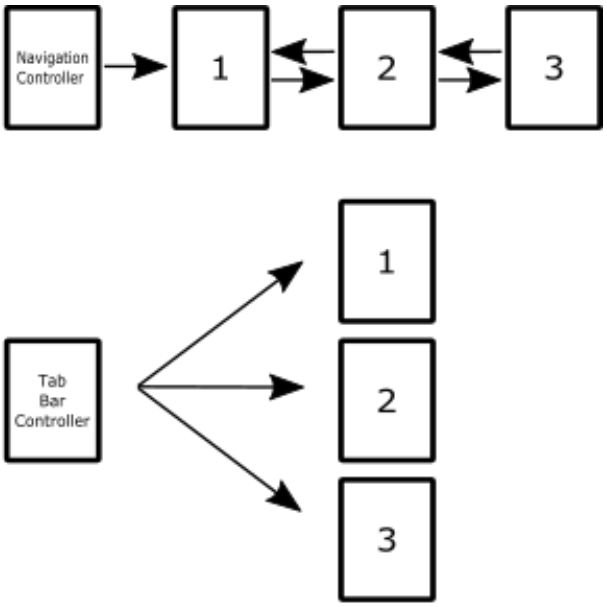
4.3.1. Komponenty pro navigaci v aplikaci

Jak již bylo zmíněno v kapitole zabývající se ovládáním aplikace, navigace uživatele je řešena kombinací komponent Navigation Controller a Tab Bar Controller.

Navigation Controller je tvořen kořenovým elementem, do kterého jsou přidány jednotlivé obrazovky. Do horní části těchto obrazovek je automaticky přidán panel, který zobrazuje název obrazovky, popř. libovolný text, tlačítko pro návrat na předchozí obrazovku a volitelně tlačítko pro spuštění další akce jako je třeba přesun

na další obrazovku, začátek měření atd. Tato komponenta si sama udržuje seznam obrazovek a pořadí, ve kterém byly navštíveny, takže navigace na předchozí obrazovku probíhá automaticky a vývojář ji nemusí manuálně implementovat.

Komponenta Tab Bar Controller má také za úkol přepínání obrazovek. Na rozdíl od Navigation Controlleru, který se používá pro vytvoření sledu obrazovek na kterých je uživatel navigován v předem stanoveném pořadí, Tab Bar Controller umožňuje uživateli přepínání mezi obrazovkami v pořadí libovolném. Rozdíl mezi těmito komponentami ilustruje následující obrázek (**Obrázek *21***).

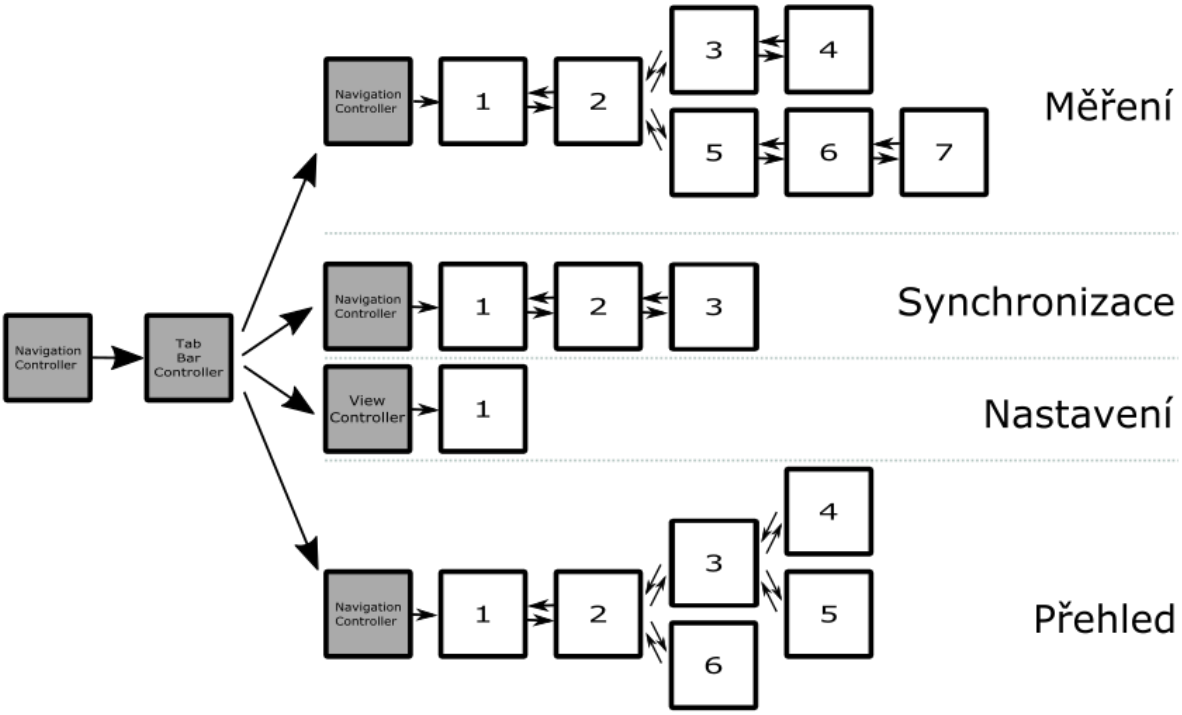


Obrázek 21: Navigation Controller a Tab Bar Controller

Zdroj: vlastní

Vhodnou kombinací těchto dvou komponent je možné vytvořit požadovanou strukturu obrazovek a docílit tak požadovaného chování aplikace. Komponent pro přechod mezi obrazovkami je k dispozici více, ale tyto dvě patří k základním a nejčastěji používaným.

Níže je znázorněna zjednodušená struktura obrazovek u jednotlivých funkcí v aplikaci Radio F-Prints (**Obrázek *22***). Čísla označují pořadí jednotlivých obrazovek v rámci dané funkce.



Obrázek 22: Hierarchie uživatelského rozhraní

Zdroj: vlastní

Šedivě podbarvená pole označují komponenty pro navigaci v aplikaci. Pole s bílým pozadím reprezentují obrazovky viditelné uživatelům. Po spuštění aplikace je uživateli zobrazena obrazovka č. 1 z funkce *Měření*. Při inicializaci aplikace je vytvořen první Navigation Controller (úplně vlevo), který zajistí vytvoření základní struktury uživatelského rozhraní, do které jsou vloženy všechny následující komponenty. Přímo v tomto Navigation Controlleru je vložena pouze komponenta Tab Bar Controller. Ta se rozděluje do čtyř větví, které reprezentují jednotlivé funkce aplikace. S výjimkou *Nastavení*, které obsahuje pouze jednu obrazovku, každá větev začíná vlastním Navigation Controllerem. Ten umožňuje navigaci mezi obrazovkami v rámci dané funkce, aniž by ovlivňoval ostatní větve nebo jimi byl sám ovlivňován. Toto řešení bylo zvoleno kvůli své přehlednosti a snadné rozšiřitelnosti, protože v případě úpravy aplikace stačí provést změnu pouze v dané větvi, přičemž ostatní funkce zůstanou nedotčeny. Níže je přehled obrazovek dle členění uvedeném na schématu (**Obrázek *22***).

- **Měření**
    - Úvodní obrazovka funkce *Měření* s jejím stručným popisem
    - Tabulka pro výběr typu měření (*Procházka* nebo *Měření na místě*)
    - Měření *Procházka* – výběr scénáře
    - Měření *Procházka* – mapa a samotné měření
    - *Měření na místě* – výběr lokace
    - *Měření na místě* – označení místa na mapě
    - *Měření na místě* – samotné měření a jeho průběh
- **Synchronizace**
    - Úvodní obrazovka funkce *Synchronizace* s jejím stručným popisem
    - Formulář pro přihlášení ke Google účtu
    - Spuštění samotné synchronizace a sledování jejího stavu
- **Nastavení**
    - Obrazovka pro uživatelskou změnu nastavení
- **Přehled**
    - Úvodní obrazovka funkce *Přehled* s jejím stručným popisem
    - Seznam provedených měření
    - Naměřené hodnoty
    - Detailní přehled hodnot získaný z BLE zařízení
    - Detailní přehled hodnot získaný z GPS
    - Mapa s vyznačeným místem, ke kterému naměřené hodnoty náleží

4.3.2. Předávání dat mezi obrazovkami

Definování přechodů mezi obrazovkami a nastavení typu animace pomocí Interface Builderu šetří čas při vývoji aplikace a pomáhá předejít chybám. Pokud je však potřeba předávat nějaká data mezi obrazovkami, např. identifikátor zvolené mapy, která má být na následující obrazovce načtena a zobrazena, je nutná úprava zdrojového kódu tříd, které reprezentují jednotlivé obrazovky.

Postup pro předávání dat je následující:

- v Interface Builderu přiřadit k požadovanému přechodu na jinou obrazovku identifikátor. Ten by měl být unikátní v rámci celé aplikace.
- ve třídě reprezentující cílové okno vytvořit vlastnosti nebo funkce, které umožní předávání dat z vnějšku třídy
- ve třídě reprezentující okno, ze kterého se přechází, zachytit notifikaci přepsáním funkce prepare for segue

Pojmenování přechodů mezi obrazovkami jednoznačným identifikátorem není povinné, je však dobrým zvykem ho používat. V rámci jedné obrazovky může být totiž přechodů nastaveno několik a to i do jednoho cílového okna. Např. pokud budeme mít na jedné obrazovce tři tlačítka pro navigací na jedno cílové okno, ale pokaždé budeme chtít předat jiná data, podle identifikátoru poznáme, o který z přechodů se jedná a budeme na něj moci patřičně zareagovat. I v případě jednoho přechodu je pojmenování vhodné z důvodu usnadnění práce a předejití chyb při případném rozvoji aplikace.

Vytvoření funkcí nebo vlastností přístupných z vnějšku třídy nevyžaduje žádnou speciální jmennou konvenci nebo použití datových typů, jak tomu je např. při použití delegátů. Je však dobré dát si pozor, aby při přepínání oken nebylo předáváno enormní množství dat, popř. aby nebyla prováděna žádná déle trvající akce, pokud to povaha okna nezbytně nevyžaduje. V opačném případě může dojít k prodlevě při přechodu mezi obrazovkami a tím ke snížení komfortu pro uživatele.

Notifikace prepare for segue jsou zasílány instancím třídy, které reprezentují obrazovku, ze které se bude přecházet jinam. Funkce je zavolána předtím, než animace proběhne. Výchozí implementace této funkce nedělá nic, její využití je tak

plně na vývojáři (Apple, 2017k). Funkce však přijímá důležité parametry nutné pro předání dat.

Ukázka zdrojového kódu (**Zdrojový kód 1**) znázorňuje princip předávání dat mezi obrazovkami. Tato konkrétní implementace se nachází ve třídě ShowDataViewController01. Funkce je zavolána při požadavku na přechod z úvodní obrazovky funkce na obrazovku obsahující tabulku se seznamem měření.

```
override func prepare(for segue: UIStoryboardSegue, sender: Any?) {

   if segue.identifier == "segueShowDataList" && collectedData != nil {
      let destinationView = segue.destination as? ShowDataViewController02
         if destinationView != nil {
            destinationView?.collectedData = collectedData!
            collectedData=nil
         }
   }
}
```

Zdrojový kód 1: Předávání dat mezi obrazovkami

Zdroj: vlastní

Funkce přijímá dva parametry, segue a sender. Parametr sender obsahuje objekt, který požadavek na přechod mezi obrazovkami vyvolal. V tomto případě se jedná o tlačítko Pokračovat na horní liště aplikace. Tento parametr však není v této situaci použit.

V implementaci funkce se však pracuje s parametrem segue, který obsahuje důležitá data. Nejprve je z jeho vlastnosti identifier získán identifikátor přechodu a porovnán s očekávaným názvem. Zároveň je provedena kontrola, zda jsou k dispozici nějaká data k předání. Poté je z vlastnosti destination získána instanci třídy, reprezentující cílovou obrazovku. Jedná se o stejnou třídu, ve které jsou vytvořené funkce nebo vlastnosti pro předávání dat z vnějšku. Vlastnost destination je typu UIViewController ze které dědí i použitá třída. Z tohoto důvodu je provedena

kontrola datového typu, resp. je proveden pokus o přetypování. Pokud byla vlastnost destination očekávaného datového typu, bylo přetypování úspěšné a je možné získanou instanci třídy naplnit daty. Pokud by byla funkce zavolána jiným přechodem mezi obrazovkami, tedy identifikátor nebo datový typ cílového okna by neodpovídal očekávaným hodnotám, dojde ke změně obrazovky bez předání dat. Ve zvolené hierarchii uživatelského rozhraní k tomu však dojít nemůže.

```
override func shouldPerformSegue(withIdentifier identifier: String,
sender: Any?) -> Bool {

   if identifier == "segueShowDataList" {

      collectedData = DataHolder.importDatabase()

      if collectedData!.count == 0 {

        let alert = UIAlertController(title: "Žádná data k zobrazení",
        message: "Nejsou k dispozici žádná    data k zobrazení. Nejprve
        synchronizujte databázi nebo proveďte měření", preferredStyle:
        UIAlertControllerStyle.alert)

        alert.addAction(UIAlertAction(title: "OK", style:
        UIAlertActionStyle.default, handler: nil))

        self.present(alert, animated: true, completion: nil)

        return false

        }

     }

   return true

}
```

Zdrojový kód 2: Kontrola dat před přechodem na jinou obrazovku

Zdroj: vlastní

K přechodu na další obrazovku nedojde ani v případě, kdy aplikace nemá k dispozici žádná data k zobrazení. Tato situace může nastat, pokud je aplikace nově

nainstalována do zařízení a ještě neproběhla synchronizace dat ani měření. V tomto případě není žádoucí, aby byla uživateli zobrazena prázdná tabulka, se kterou by nemohl pokračovat v zamýšlené činnosti. Místo toho dojde k zastavení přechodu na další obrazovku a uživateli je zobrazena hláška o absenci dat. Funkce prepare for segue sice neumožňuje přerušit přechod na jinou obrazovku, ale za tímto účelem je k dispozici funkce shouldPerformSegue withIdentifier. Příklad použití této funkce je v ukázce (**Zdrojový kód 2**).

Tato funkce přijímá dva parametry. Název přechodu identifier a stejně jako v případě funkce prepare for segue parametr sender. Funkce ale navíc obsahuje výstupní parametr typu Bool, pomocí kterého se určuje, zda se má přechod na jinou obrazovku provést (hodnota výstupního parametru bude true), nebo zda má být tento požadavek stornován (hodnota bude false). V tomto konkrétním případě je nejprve provedena kontrola identifikátoru, protože dostupnost dat má smysl kontrolovat pouze při přechodu na obrazovku, která tyto data potřebuje. Dále je provedeno načtení dat z databáze následované kontrolou počtu přijatých záznamů. Pokud je zjištěno, že získaná kolekce neobsahuje žádná data, požadavek pro přechod na další obrazovku je stornován a uživateli je zobrazeno upozornění.

4.3.3. Mapové podklady a označování polohy

Jak již bylo zmíněno, mapové podklady jsou načítány ze vzdáleného serveru. Každá mapa je webová stránka načítaná pomocí HTTP protokolu, která je uživateli zobrazovaná pomocí komponenty UIWebView. Zobrazení mapy, nastavení hranic pro výběr, označení vybraného místa červeným bodem, získání souřadnic a přibližování/oddalování mapy je implementováno přímo ve stránce s mapou. Aplikace tedy pouze těchto funkcí využívá.

Seznam Javascript funkcí a html elementů očekávaných aplikací:

- *GetCrosshairLogicX():* získání souřadnice X aktuálně vybraného umístění
- *GetCrosshairLogicY():* získání souřadnice Y aktuálně vybraného umístění

- *scrollToLogicXY(x, y)*: přesun na souřadnice x, y a jejich označení červeným bodem
- element *button* s atributem id nastaveným na hodnotu zoom-in: tlačítko pro přiblížení mapy
- element *button* s atributem id nastaveným na hodnotu zoom-out: tlačítko pro oddálení mapy

Funkce getCrosshairLogicX *a* getCrosshairLogicY jsou použity při výběru lokace před začátkem měření. Aktuálně vybrané souřadnice jsou zobrazeny uživateli v horní části obrazovky uprostřed. Souřadnice jsou přepočítány vždy po dokončení scrollováni mapy.

Funkce scrollToLogicXY je používána při zobrazování dat z předchozích měření a při zobrazování očekávané pozice uživatele v rámci měření *Procházka*. Funkci jsou jako vstupní parametry předány souřadnice X a Y. Jejím zavoláním dojde k označení požadovaných souřadnic červeným bodem a vycentrováním tohoto místa na obrazovce. Pokud se v náhledu naměřených hodnot nebo při měření *Procházka* pokusí uživatel změnit umístění bodu na mapě, je po dokončení scrollování mapy funkce scrollToLogicXY volána znovu. Toto chování je zde záměrné proto, aby uživatel měl jistotu, že označené souřadnice skutečně odpovídají místu měření a nemohlo dojít nechtěným posunutím mapy k jejich změně.

Tlačítka s atributem id zoom-in a zoom-out jsou očekávána jak při sběru dat, tak při zobrazování předchozích měření. Tyto tlačítka jsou po načtení mapy zobrazena v pravé dolní části obrazovky. Jsou označena symbolem + resp. – a slouží k přiblížení resp. oddálení mapy. Z důvodu úspory místa na displeji zařízení a zlepšení komfortu práce s aplikací jsou tyto tlačítka skryta a jejich funkce nahrazena gestem *Pinch*, tedy rozevřením resp. sevřením dvou prstů, které je běžně používané u zařízeních s dotykovým displejem.

Přibližování resp. oddalování mapy je realizováno odchytáváním začátku a konce gesta *Pinch* a zavolání funkce click tlačítek zoom-in resp. zoom-out pokud se úroveň přiblížení změní alespoň o minimální nastavenou hodnotu. Ke správnému fungování bylo nutné zvolit několik konstant, jejichž kombinace hodnot byla nastavena při vývoji a testování tak, aby zoomování bylo co nejhladší a uživatelsky přívětivé.

Nejprve byly zvoleny konstanty určující hranice maximálního přiblížení a oddálení. Pokud uvažujeme, že načtená mapa, která nebyla přiblížena ani oddálena má úroveň přiblížení 1.0 pak konstanta pro maximální přiblížení má hodnotu 1.5 a pro maximální oddálení 0.5. Vzhledem k tomu, že po skončení gesta není interně udržována informace o celkové úrovni přiblížení, je použita pomocná proměnná, která umožňuje tuto informaci mezi gesty předávat. Pokud by předávána nebyla, maximální hranice přiblížení a oddálení by fungovala pouze v rámci jednoho gesta. Při opakovaném používání gesta by však úroveň přiblížení a oddálení nebyla žádným způsobem omezena. Další konstantou je citlivost gesta. Testováním byla zvolena hodnota 0.025, při které bylo použití gesta nejplynulejší. Pokud absolutní hodnota rozdílu úrovně naposledy provedeného přiblížení a úrovně přiblížení v rámci gesta není větší než tato konstanta, přiblížení mapy se nemění. Pro větší přehlednost a lepší pochopení toho, jak je tato funkcionalita implementována, je níže uveden zdrojový kód obou klíčových funkcí.

```
func gestureRecognizer(_ gestureRecognizer: UIGestureRecognizer,
shouldRequireFailureOf otherGestureRecognizer: UIGestureRecognizer) -> Bool
{
   if let pinchGesture = gestureRecognizer as? UIPinchGestureRecognizer {
      pinchGesture.scale = lastScaleValue
   }
   return false
}
```

Zdrojový kód 3: Začátek gesta Pinch

Zdroj: vlastní

Funkce gestureRecognizer, jejíž implementace je v ukázce (**Zdrojový kód 3**), je zavolána při začátku gesta. Vzhledem k tomu že tuto funkci kromě gesta *Pinch* mohou volat i gesta ostatní, je provedena kontrola, zda je parametr gestureRecognizer očekávaného typu tím, že je proveden pokus o jeho přetypování.

Pokud je tento pokus úspěšný, je gestu nastavena úroveň přiblížení převzatá z předchozího gesta. Tím je zajištěno, že nemůže dojít k prolomení hranic maximální úrovně přiblížení a oddálení ani opakovaným použitím gesta. Pokud bychom se chtěli tomuto předávání hodnot vyhnout, řešením by bylo přesunout kontrolu spodní a horní hranice přiblížení přímo do mapy a tuto hranici hlídat pomocí Javascriptu. V současné verzi map (březen 2017) je omezena pouze hranice oddálení, nikoli však přiblížení.

```
@IBAction func gestureDidUse(_ sender: UIPinchGestureRecognizer) {


   if sender.scale < minScale {sender.scale = minScale}
   else if sender.scale > maxScale {sender.scale = maxScale}


   if abs(lastScaleValue - sender.scale) < 0.025 {return}


   if sender.velocity < 0 && sender.scale > minScale {
      mapWebView.stringByEvaluatingJavaScript(from:
      "document.getElementById('zoom-out').click();") // zoom-out
    }
    else if sender.velocity > 0 && sender.scale < maxScale {
      mapWebView.stringByEvaluatingJavaScript(from:
      "document.getElementById('zoom-in').click();") // zoom-in
    }
    lastScaleValue = sender.scale
}
```

Zdrojový kód 4: Provádění gesta Pinch

Zdroj: vlastní

Funkce gestureDidUse, jež je v ukázce (**Zdrojový kód 4**), je volána během provádění gesta. Z parametru sender je možné získat potřebné informace. Vlastnost scale

obsahuje velikost úrovně přiblížení a vlastnost velocity informaci o tom, zda se jedná o přiblížení nebo oddálení. Přitom platí, že pokud je hodnota vlastnosti velocity menší než nula, jedná se o oddálení mapy. V opačném případě jde o přiblížení. Z povahy gesta nemůže tato proměnná nabývat hodnoty 0.

4.4. Data a přístup k databázi

Pro ukládání dat v Couchbase databázi je použit formát JSON[6]. To dává databázi značnou variabilitu, protože každý záznam (v terminologii Couchbase databáze se jedná o dokument), může mít jiné atributy (Couchbase, 2017b). Pro zpracování dat mobilní aplikací je však vhodnější objektová reprezentace těchto dat. Z tohoto důvodu aplikace obsahuje třídu DataHolder, která se stará o převod JSON do objektové podoby a naopak, z instance třídy DataHolder umí vygenerovat JSON pro uložení do lokální databáze, ve které je jedno měření reprezentováno jedním dokumentem.

DataHolder obsahuje řadu vlastností, které odpovídají jednotlivým atributům dokumentu z databáze. Přitom platí, že jedna instance třídy DataHolder odpovídá jednomu dokumentu, tedy jednomu záznamu o provedeném měření. Více záznamů je pak reprezentováno kolekcí instancí třídy DataHolder.

Jako již bylo zmíněno, třída obsahuje vlastnosti pro uložení naměřených hodnot. Pro všechny hodnoty mimo dat naměřených z bezdrátových zařízení a dokumentu z Couchbase databáze, který daná instance třídy reprezentuje, jsou použity základní datové typy. Jedná se např. o čas začátku a konce měření, identifikátor zvolené mapy a vybrané souřadnice, verze systému, ID iOS zařízení atd. Pro uložení hodnot, které byly získány v průběhu měření je zde vytvořena vlastnost collectedData. Jedná se o kolekci instancí tříd, které implementují protokol ProtocolWirelessManager. Ten předepisuje třídám, jaké vlastnosti a funkce musejí implementovat. Toto řešení bylo zvoleno z důvodu jednotného přístupu k datům, např. při zobrazení v tabulce. Platí, že jedna položka v kolekci collectedData reprezentuje jedno bezdrátové zařízení, které bylo při měření zjištěno.

[6] JavaScript Object Notation

4.4.1. Třídy reprezentující fyzická zařízení

Jak již bylo zmíněno v předchozí kapitole, jedná se o třídy implementující protokol ProtocolWirelessManager. Díky němu je umožněn jednotný přístup k základním údajům, jako je třeba název zařízení nebo jeho identifikátor bez ohledu na typ zařízení. Níže je uveden seznam tříd implementující tento protokol a stručný popis jejich použití:

- *DataItemBLE*: jedna instance třídy DataItemBLE odpovídá jednomu zjištěnému Bluetooth Low Energy zařízení. V kolekci může být těchto zařízení více a každá instance třídy obsahuje kolekci hodnot získaných v průběhu měření.
- *DataItemGPS*: pokud došlo během měření k získání údajů o poloze z GPS modulu, obsahuje kolekce collectedData jednu instanci třídy DataItemGPS. Ta má uvnitř kolekci hodnot, které jsou během měření průběžně získávány z GPS modulu.
- *DataItemBTS*: pokud byly získávány údaje z mobilní sítě do které bylo v době měření zařízení připojeno, obsahuje kolekce collectedData jednu instanci třídy DataItemBTS. Jedná se o statická data, neměnná v průběhu měření.
- *DataItemWifi*: pokud byla získávána data u WiFi sítě do které bylo v době měření zařízení připojeno, obsahuje kolekce collectedData jednu instanci třídy DataItemWiFi. Jedná se o statická data, neměnná v průběhu měření.

Kolekce collectedData tedy může obsahovat 0 až neomezený počet[7] instancí třídy DataItemBLE a 0 nebo 1 instanci pro každou z ostatních uvedených tříd. Tento způsob implementace byl zvolen pro svou snadnou rozšiřitelnost. V případě že v budoucnu vyvstane požadavek na přidání nového typu zařízení, není potřeba přidávat nové vlastnosti a dělat úpravy napříč aplikací. Rozšíření se provede vytvořením nové třídy implementující protokol ProtocolWirelessManager, jejíž instance budou přidávány do kolekce collectedData.

[7] Počet záznamů je limitován dostupnou operační pamětí zařízení

4.4.2. Řazení záznamů

Při prezentaci dat uživateli je nutné nastavit logiku řazení záznamů tak, aby zobrazené pořadí bylo přehledné a pro uživatele lehce pochopitelné. Z tohoto důvodu bylo zvolené sestupné řazení podle data a času začátku měření.

```
/// Chování operátoru < (je menší než).
public static func <(first: DataHolder, second: DataHolder) -> Bool {

   if first.collectStartTime == nil {return false}
   if second.collectStartTime == nil {return true}
   return first.collectStartTime! < second.collectStartTime!
}


/// Chování operátoru > (je větší než)
public static func >(first: DataHolder, second: DataHolder) -> Bool {

   if first.collectStartTime == nil {return false}
   if second.collectStartTime == nil {return true}
   return first.collectStartTime! > second.collectStartTime!
}
```

Zdrojový kód 5: Implementace sorteru

Zdroj: vlastní

Aby byla logika řazení implementována pouze na jednom místě a také aby nebylo nutné kvůli řazení procházet kolekce instancí třídy DataHolder sekvenčně, implementuje třída DataHolder protokol Comparable (**Zdrojový kód 5**). Díky implementaci tohoto protokolu je přímo v těle třídy definována logika porovnávání záznamů pomocí operátorů *je menší než* a *je větší než*. V praxi se jedná se o dvě statické funkce, jednu pro každý operátor, s předepsanou jmennou konvencí, které na svém vstupu přijímají dvě instance třídy DataHolder a výstupem je Bool hodnota, která podle typu použité funkce indikuje, zda je první parametr větší, resp. menší než parametr druhý.

Kromě porovnávání dvou instancí tříd implementace protokolu Comparable umožňuje použít řazení kolekce dat. Toho je využito při načítání dat z databáze, kdy je kolekce záznamů před odesláním seřazena sestupně (**Zdrojový kód 6**).

```
return collectedData.sorted(by: { $0 > $1 })
```

Zdrojový kód 6: Řazení kolekce

Zdroj: vlastní

4.4.3. Převod dat z JSON do objektové podoby a naopak

Třída DataHolder obsahuje funkce pro převedení hodnot zadaných v instanci třídy do formátu JSON, který je možné přímo uložit do databáze a naopak, umožňuje na základě dokumentu z databáze vytvořit novou instanci třídy a naplnit jí daty.

Pro export do formátu JSON je implementována instanční funkce exportToJson. Tato funkce na svém vstupu nepřijímá žádné parametry, data jsou získána z metod a vlastností instance třídy, na které je tato funkce zavolána. Výstupem jsou data ve formátu požadovaném databází. V programovacím jazyce Swift je formát JSON reprezentován kolekcí dvojic klíč-hodnota.

Pro získání instance třídy DataHolder, která je naplněna daty z databáze, slouží statická funkce importDocument. Tato funkce na svém vstupu přijímá jeden dokument z CouchBase databáze. Výstupním parametrem je pak nová instance třídy DataHolder, která obsahuje data ze vstupního dokumentu.

Pro získání všech[8] naměřených hodnot z databáze je určena statická funkce importDatabase. Funkce nemá žádné vstupní parametry, výstupem je kolekce instancí třídy DataHolder. Funkce získá z Couchbase databáze všechny dokumenty reprezentující provedená měření na platformě iOS. Tyto dokumenty jsou sekvenčně procházeny a předávány funkci importDocument, která je popsána v předchozím odstavci. Instance třídy DataHolder, které jsou navráceny funkcí importDocument,

[8] Aplikace pracuje pouze se záznamy, které byly pořízeny na zařízení s operačním systémem iOS. Data naměřená na jiných platformách jsou ignorována z důvodu odlišností ve formátu dat.

jsou vkládány do kolekce, která je výstupním parametrem funkce importDatabase. Tato kolekce je seřazena vzestupně podle data a času začátku měření.

4.4.4. Podepisování odesílaných dat

Všechny záznamy odesílané do vzdálené databáze jsou před provedením synchronizace podepsány Google ID uživatele. V praxi to je realizováno přiřazením hodnoty Google ID k atributu couchbase_sync_gateway_id v Couchbase databázi. Vzhledem k tomu že přihlášení uživatele je vyžadováno pouze před provedením synchronizace a nikoli před začátkem měření, jsou záznamy o měření podepisovány až před provedením synchronizace. Nejedná se tedy o Google ID uživatele který provedl měření, ale od ID uživatele který spustil synchronizaci dat, při které byl daný záznam poprvé vložen do databáze na vzdáleném serveru.

Pro podepsání dat je ve třídě DataHolder implementována statická funkce signData, která na svém vstupu přijímá Google ID jako text. Následně jsou získány všechny dokumenty z lokální Couchbase databáze a jsou hledány záznamy, které byly pořízeny na daném zařízení (kontroluje se ID zařízení) a zároveň daný záznam není podepsán (hodnota atributu couchbase_sync_gateway_id je nastavena na znak ‘‘-‘‘). U všech záznamů vyhovující oběma podmínkám následně dojde k nastavení hodnoty atributu couchbase_sync_gateway_id na hodnotu Google ID, která byla obdržena jako vstupní parametr.

4.5. Sběr radiových fingerprintů

Soubory se zdrojovým kódem pro sběr radiových dat jsou uložené ve složce WirelessManagers. Jsou zde třídy ManagerBLE, ManagerGPS, ManagerBTS a ManagerWiFi. Všechny tyto třídy používají delegáta WirelessManagerDelegate, pomocí kterého informují ostatní objekty, které tohoto delegáta přijímají, o připojení nového zařízení, popř. o přijetí nové hodnoty.

WirelessManagerDelegate předepisuje tři funkce: deviceDidConnect, bleDataReceived gpsDataReceived. Funkci deviceDidConnect používají všechny čtyři zmíněné třídy pro oznámení o získání dat z nového zařízení. Tato funkce má

jeden parametr a tím je instance třídy implementující protokol ProtocolWirelessManager. Tato instance třídy představuje jedno fyzické zařízení. Jak již bylo zmíněno v kapitole zabývající se popisem tříd reprezentující fyzická zařízení, může se jednat o instance tříd DataItemBLE, DataItemGPS, DataItemBTS a DataItemWIFI.

Druhou funkcí, kterou tento protokol předepisuje, je bleDataReceived. Tuto funkci využívá třída ManagerBLE ve chvíli, kdy obdrží z BLE zařízení novou hodnotu síly signálu RSSI. Parametry této funkce jsou UUID a získaná hodnota signálu. UUID je jednoznačný identifikátor, pomocí kterého je daný BLE zařízení nalezeno v seznamu připojených zařízení a je k němu přidána nová celočíselná hodnota RSSI.

Poslední funkcí je gpsDataReceived. Tato funkce je používána třídou ManagerGPS, když tato třída obdrží informaci o změně polohy. Princip fungování je stejný jako v předchozím případě, podle UUID je nalezeno zařízení v seznamu a následně je k němu přidána nová hodnota aktuální polohy.

Třída ManagerWIFI a ManagerBTS slouží k získání dat z WiFi sítě resp. mobilní sítě, ke které je v době měření iOS zařízení připojené. Vzhledem k omezením platformy iOS se aplikace omezuje na získání několika statických hodnot, které se v průběhu měření nemění. Po vytvoření nové instance těchto tříd dojde k vygenerování interního UUID, které slouží pouze pro identifikaci zařízení v kolekci collectedData, která je obsažena ve třídě DataHolder. Data jsou pak získána pomocí delegátu zavoláním metody start.

Třída ManagerGPS slouží k získání geografické polohy iOS zařízení a její průběžné aktualizaci během měření. Po vytvoření nové instance třídy dojde k vygenerování nového UUID, které slouží jako identifikátor v kolekci collectedData ve třídě DataHolder. Po zavolání metody start dojde k vytvoření nové instance třídy CLLocationManager a nastavení hodnot atributům, které určují přesnost zaměření a četnost notifikací při pohybu. Pro potřeby této aplikace byla zvoleno nastavení zaměřené na přesnost a co nejčastější získávání notifikací (Apple, 2017a). Toto nastavení by při dlouhodobém použité mělo značně negativní dopad na výdrž baterie, ale vzhledem k tomu, že délka měření je řádově v desítkách sekund a smyslem aplikace je poskytnout co nejpřesnější data, je tento fakt zanedbán.

Když je nastavení provedeno, je zaslán požadavek na odběr dat z GPS modulu. Nejprve je provedena kontrola, zda uživatel souhlasí s tím, aby aplikace získala jeho polohu. Pokud je kontrola úspěšná, je pomocí delegáta WirelessManagerDelegate zaslána notifikace o novém zařízení. Poté je při každé změně polohy, o které informuje instance třídy CLLocationManager, odeslána přes téhož delegáta notifikace o přijetí nových hodnot. Sběr dat je ukončen zavoláním metody stop.

Třída ManagerBLE slouží ke sběru dat z BLE zařízení. Zavoláním metody start dojde k vytvoření nové instance třídy CBCentralManager, která zajišťuje samotné zachytávání dat vysílané BLE zařízeními. Nejprve je provedena kontrola, zda má iOS zařízení zapnuté Bluetooth. Pokud je kontrola úspěšná, začne vyhledávání dostupných BLE zařízení. Pokud je nějaké zařízení nalezeno, je obdržena notifikace didDiscover. Jedním z parametrů této notifikace je i parametr datového typu CBPeriheral, který reprezentuje nalezené zařízení. Vzhledem k tomu, že tento parametr má slabou (weak) referenci, je nutné jeho uložení do interní kolekce, aby s ním bylo možné později pracovat. Pokud by k uložené nedošlo, BLE zařízení by se opakovaně odpojovalo.

Po obdržení notifikace didDiscover je proveden pokus o spojení se zařízením. Pokud je úspěšný, je obdržena notifikace didConnect ve které je zaslán požadavek na získání hodnoty síly signálu RSSI. Hodnota je obdržena pomocí notifikace didReadRSSI kde je přečtena a společně s UUID odeslána pomocí delegáta WirelessManagerDelegate dále. Zároveň je znovu zaslán požadavek na získání hodnoty RSSI. V případě že se zařízení odpojí, je obdržena notifikace didDisconnectPeripheral, kde je proveden pokus o nové spojení. Pojmy spojení a odpojení jsou na tomto místě používány výhradně v kontextu používání třídy CBCentralManager. Navázání spojení mezi iOS zařízením a BLE zařízením jako takové, není v aplikaci Radio F-Prints používáno.

4.6. Konfigurační soubory

Aplikace ke své konfiguraci používá konfigurační soubory načítané z internetu. Jedná se o dva soubory ve formátu JSON, které obsahují data nutná pro zahájení měření. První soubor obsahuje seznam mapových podkladů a URL adresy využívaných služeb (**Zdrojový kód 7**). Druhý soubor obsahuje testovací scénáře (**Zdrojový kód 8**).

Atributy konfiguračního souboru aplikace (Zdrojový kód 7)

- *gatewayURL*: URL adresa služby Couchbase databáze, která je využívaná při synchronizaci dat
- *authURL*: URL adresa přihlašovací stránky ke Google účtu
- *scenariosURL*: URL adresa druhého konfiguračního souboru obsahující seznam scénářů (jeho popis je níže)

- maps: pole mapových podkladů. Každá položka v poli obsahuje klíče:
    - *id*: jednoznačný identifikátor mapy
    - *location*: název lokace
    - *description*: popis lokace (např. ulice, podlaží atd.)
    - *mapURL*: URL adresa mapy

```
{
   "gatewayURL": "http://beacon.uhk.cz/beacongw",
   "authURL": "http://beacon.uhk.cz/auth",
   "scenariosURL" : "http://beacon.uhk.cz/app-scenario.json",
   "maps": [
      {
         "id": "J1NP",
         "location": "Univerzita Hradec Králové",
         "description": "Hradecká 1249/6 - Budova J 1NP",
         "mapURL": "http://beacon.uhk.cz/fimnav-webview/?map=J1NP"
      },
      {
         "id": "J2NP",
         "location": "Univerzita Hradec Králové",
         "description": "Hradecká 1249/6 - Budova J 1NP",
         "mapURL": "http://beacon.uhk.cz/fimnav-webview/?map=J2NP"
      }
   ]
}
```

Zdrojový kód 7: Konfigurační soubor aplikace

Zdroj: vlastní

Velikost pole maps není omezena a může být libovolně měněna. ID mapy musí být unikátní, protože tato hodnota je přidávána k provedeným měřením, z důvodu jednoznačné identifikace lokace, kde bylo měření prováděno. Pokud není tento soubor z nějakého důvodu dostupný, nelze provádět ani jeden typ měření, synchronizaci dat a při prohlížení dříve naměřených hodnot nebude dostupná mapa.

Atributy konfiguračního souboru testovacích scénářů (Zdrojový kód 8)

Tento soubor obsahuje jeden kořenový prvek scenarios, který je definovaný jako pole dalších prvků. Ty obsahují následující klíče:

- *scenariosID*: jednoznačný identifikátor testovacího scénáře
- *name*: název scénáře. Hodnota přiřazená k tomuto klíči se následně uživateli zobrazí v seznamu scénářů
- *mapID*: identifikátor mapy lokace, ve které má být testování prováděno. Tento identifikátor by měl korespondovat s některým ID mapy, která jsou uvedena v prvním konfiguračním souboru.
- *points*: jedná se o pole bodů, které jsou použity při navigování uživatele během měření typu *Procházka*. Každý bod obsahuje klíče X a Y. Počet bodů není omezen a mohou se opakovat. Při zobrazování cesty uživateli jsou jednotlivé body procházeny v pořadí, v jakém jsou zadány v tomto poli. Měření končí dosažením posledního bodu.

```
{
   "scenarios": [
      {
         "scenarioID": "scenarioJ1NP1",
         "name": "FIM UHK - 1 NP (okruh)",
         "mapID": "J1NP",
         "points": [
            {"X": 1000, "Y": 1863},
            {"X": 363, "Y": 1863},
            {"X": 363, "Y": 701},
            {"X": 1534, "Y": 701},
            {"X": 1534, "Y": 1881},
            {"X": 1028, "Y": 1881}
         ]
      },
      {
         "scenarioID": "scenarioJ2NP1",
         "name": "FIM UHK - 2 NP (okruh)",
         "mapID": "J2NP",
         "points": [
            {"X": 1996, "Y": 713},
            {"X": 1996, "Y": 2196},
            {"X": 1856, "Y": 2354},
            {"X": 1608, "Y": 2354},
            {"X": 1608, "Y": 2963},
            {"X": 865, "Y": 2956},
            {"X": 865, "Y": 2352},
            {"X": 495, "Y": 2331},
            {"X": 495, "Y": 825},
            {"X": 1958, "Y": 825}
         ]
      }
   ]
}
```

Zdrojový kód 8: Konfigurační soubor scénářů

Zdroj: vlastní

Pokud tento konfigurační soubor nebude dostupný, nebude možné provádět měření typu *Procházka*. Ostatní funkce aplikace budou však dostupné bez omezení.

Toto řešení nastavení aplikace, kdy jsou načítány konfigurační soubory online, bylo zvoleno kvůli své variabilitě. Pokud dojde k přesunu databáze nebo mapových podkladů na jiný server, stačí udělat úpravu konfiguračního souboru, aniž by došlo ke zvýšeným nárokům na uživatele. Současně jde provádět změny u mapových podkladů a testovacích scénářů, kdy se provedené úpravy promítnou prakticky okamžitě u všech uživatelů.

5 Testování aplikace, sběr a vyhodnocení dat

5.1. Testování funkcí a uživatelského rozhraní

Testování aplikace probíhalo průběžně během vývoje, kdy byly testovány jednotlivé funkce a poté po jejím dokončení, kdy byla aplikace testována jako celek.

Testování uživatelské rozhraní probíhalo na simulátoru, který je součástí vývojového prostředí Xcode (Apple, 2017l) a na fyzickém zařízeni, na kterém byl později prováděn také sběr dat. Pro testování byly použity simulátory imitující zařízení iPhone 6S, iPhone 6S Plus, iPhone 7, iPhone 7 Plus, iPad Air 2, iPad Mini 4 a iPad Pro (obě velikosti) s iOS 10.1 a 10.2. Fyzické zařízení, na kterém bylo prováděno testování, byl iPhone 6S s iOS 10.2 a 10.3

Testování na simulátoru bylo prováděno zejména z důvodu možnosti nasimulovat kombinaci různých verzí iOS a typů zařízení. Díky tomu bylo možné zjistit, zda je vzhled aplikace, její chování a rozložení uživatelského rozhraní na různých velikostech displeje takové, jaké se očekává. Při testování bylo zjištěno několik drobných chyb v zobrazení některých prvků uživatelského rozhraní, jako např. nesprávné ukotvení tlačítka nebo popisku. Tyto chyby se většinou projevily při změně orientace zařízení a byly na základě výsledků testování opraveny.

Na fyzickém zařízení byla testována uživatelská přívětivost a plynulost chodu aplikace, protože na simulátoru není možné otestovat, zda rozložení jednotlivých prvků uživatelského rozhraní neomezuje uživatele v používání aplikace a zda aplikace správně reaguje na gesta. Výpočetně náročnější operace je také nutné testovat na fyzickém zařízení, protože mobilní zařízení má mnohem omezenější prostředky (RAM, procesor) než simulátor. Ten využívá prostředků počítače, na kterém je spuštěn, přičemž velikost RAM nebo výkon procesoru není omezován. Pro porovnání iPhone 6S má 2 GB RAM a 1.84 GHz procesor, ale iMac, na kterém byl provozován simulátor, má k dispozici 24 GB RAM a čtyřjádrový procesor s taktem 3,2 GHZ.

Testování funkcí probíhalo pouze na fyzickém zařízení a to jak z důvodu omezeného výkonu mobilního zařízení, jak bylo popsáno v předchozím odstavci, tak z důvodu

velice obtížné až nemožné simulace sběru dat z Bluetooth, WiFi a polohových senzorů na simulátoru. Testování sběru dat probíhalo průběžně s využitím zapůjčených beaconů a poté přímo v budově Fakulty informatiky a managementu Univerzity Hradec Králové, kde byla testována zejména schopnost aplikace zpracovávat data z většího množství BLE zařízení. Díky testování v terénu byl objeven zásadní problém při sběru dat z většího množství beaconů. Analýzou crash logu[9] v Xcode bylo zjištěno, že tento problém byl způsoben několikanásobným přístupem ke sdíleným prostředkům z více vláken současně. Řešení spočívalo v synchronizování vláken, která ke sdíleným prostředkům přistupují. Tento problém se dříve při testování během vývoje, kdy byla sbírána data pouze ze dvou beaconů, neprojevil.

5.2. Sběr a vyhodnocení dat

Sběr dat probíhal ve 3. podlaží budovy Fakulty informatiky a managementu. Pro měření dat byly využity funkce *Procházka* i *Měření na místě*. Vyhodnocení naměřených dat probíhalo pomocí software *Radio localization*, který byl na Fakultě informatiky a managementu Univerzity Hradec Králové za tímto účelem vytvořen. Tento software obsahuje seznam beaconů na budově FIM a umí zpracovat datový soubor Couchbase databáze. Zdrojový kód verze aplikace, která byla pro vyhodnocení dat použita, je dostupný na https://github.com/pavkriz/radio-localization-eval/tree/dedek. Aplikace *Radio localization* se také nachází na přiloženém CD a to ve dvou verzích. Jednou s filtry nastavenými na vyhodnocení měření *Procházka* (měření se uskutečnilo 5. 4. 2017) a jednou s filtry nastavenými na vyhodnocení *Měření na místě* (měření proběhlo 10. 4. 2017).

Tato aplikace měří přesnost pomocí Leave-one-out cross-validation, tedy křížové validace, kde je vždy jedno měření označeno jako testovací, zbytek vzorku jsou pak data učící. Tento postup je opakovaně aplikován pro každý bod v kolekci naměřených hodnot. Aplikace se u testovacích vzorku snaží odhadnout pozici, na které bylo prováděno měření. Spočítaný výsledek je poté porovnán se skutečností a výstupem z aplikace je medián vzdálenosti odhadnutých hodnot od skutečnosti a

[9] Soubor, který obsahuje informace o pádu aplikace

maximální hodnota této vzdálenosti. Aplikace provádí výpočet zvlášť pro data získaná z WiFi, Bluetooth Low Energy a jejich kombinaci.

U měření *Procházka* byl použit scénář *FIM UHK – 3 NP (okruh)*. Jedná se o scénář, jehož trasa je naplánována jako okruh po chodbě 3. podlaží. Tento scénář byl spuštěn pětkrát a během těchto pěti okruhů bylo získáno celkem 264 jednotlivých měření.

Vyhodnocení naměřených dat z měření *Procházka* je uvedeno v tabulce 3. Medián a maximální odchylka jsou vzdálenosti mezi skutečnými souřadnicemi a souřadnicemi odhadnutých algoritmem aplikace *Radio localization*. Vzdálenost je uvedena v metrech.

Typ zařízení	Medián	Maximální odchylka
WiFi	5,49375 m	16,85802 m
BLE	2,33524 m	19,01 m
WiFi + BLE	2,72415 m	19,01 m

Tabulka 3: Vyhodnocení naměřených hodnot (procházka)

Zdroj: aplikace Radio localization

Typ zařízení	Medián	Maximální odchylka
WiFi	7,53215 m	32,77209 m
BLE	2,20485 m	28,18678 m
WiFi + BLE	1,86174 m	25,60978 m

Tabulka 4: vyhodnocení naměřených hodnot (měření na místě)

Zdroj: aplikace Radio localization

Měření na místě bylo provedeno také ve 3. podlaží Fakulty informatiky a managementu. Měření se uskutečnilo ve dvou vzájemně kolmých chodbách po celé jejich délce a šířce. Každé měření trvalo 20 sekund a rozestup mezi jednotlivými místy, kde bylo měření prováděno, byl 1 metr. Celkem bylo provedeno 117 měření. Jejich vyhodnocení aplikací *Radio localization* je uvedeno v tabulce 4.

Jak je z tabulky patrné, kombinace WiFi + BLE je vyhodnocena jako přesnější, než při měření *Procházka* (tabulka 3). To může být zapříčiněno tím, že pokud uživatel stojí na jednom místě, je více času na získání dat z beaconů a připojení k přístupovým bodům WiFi. Na rozdíl od měření *Procházka* se také nestane, že jsou naměřená data připsána k trochu jiné poloze v důsledku dvousekundového intervalu.

Vyšší medián u WiFi , který byl zjištěn u obou měření, může být částečně způsoben omezením platformy iOS, kdy není možné získat seznam přístupových bodů v okolí a je dostupný pouze přístupový bod, ke kterému je zařízení aktuálně připojeno. Pro vyhodnocení přesnosti WiFi tak byl k dispozici mnohem menší počet naměřených dat než k BLE.

6 Závěr

Cílem této práce byl návrh aplikace pro sběr radiových dat a implementace vlastního řešení, které je schopen používat běžný uživatel. Výstupem je plně funkční aplikace Radio F-Prints pro zařízení s operačním systémem iOS, která tento sběr dat umožňuje. Aplikace splňuje podmínky společnosti Apple Inc. pro publikování v App Store (Apple, 2017b) a může tedy být prostřednictvím tohoto obchodu distribuována koncovým uživatelům.

Při zkušebním sběru dat, který byl proveden na 3. podlaží budovy Fakulty informatiky a managementu Univerzity Hradec Králové na zařízení iPhone 6S, bylo nejpřesnějších výsledků dosaženo při použití funkce *Měření na místě* a zkombinování dat získaných z WiFi a BLE zařízení. Při této konfiguraci byl medián chyby spočítaný aplikací *Radio localization* nejmenší a to 1,86174 m.

Aplikace nabízí prostor pro zlepšení a to zejména u funkce procházka, kde by mohlo být teoreticky možné provádět během měření s pomocí akcelerometru korekci rychlosti chůze, protože ta je v současné době konstantní. Implementace tohoto vylepšení by si však vyžádala rozsáhlejší testování, do kterého by bylo nutné zahrnout všechny typy iOS zařízení na kterých může být aplikace provozována a také větší počet uživatelů. Údaje z jednotlivých typů zařízení se většinou trochu liší a navíc záleží i na způsobu chůze uživatele a uchopení zařízení v ruce.

Pokud by mapové podklady obsahovaly metadata, ve kterých by byly zahrnuty informace o stěnách, dveřích, délce chodeb atd. mohlo by dojít k výraznému zjednodušení návrhu testovacích scénářů. Do něj by pak nemusela být zahrnuta celá trasa, ale stačilo by zadat seznam bodů, které má uživatel projít. Aplikace by pak sama pomocí vhodného algoritmu navrhla trasu, která by byla nabídnuta uživateli.

Pokud by bylo v plánu přidání dalších mapových podkladů ve zcela jiných lokalitách, bylo by možné upravit konfigurační soubor tak, aby jednotlivé mapy obsahovaly GPS souřadnice dané lokality. Uživateli by se pak nemusely načítat všechny mapové podklady, ale pouze lokace nacházející se v jeho okolí. Stejným způsobem by bylo možné filtrovat i testovací scénáře.

Seznam použité literatury

Apple: *CLLocationManager* [online]
Dostupné z: https://developer.apple.com/reference/corelocation/cllocationmanager
[přístup 11. 3. 2017a]

Apple: *App Store Review Guidelines* [online]
Dostupné z: https://developer.apple.com/app-store/review/guidelines/
[přístup 11. 3. 2017b]

Apple: *iOS Human Interface Guidelines* [online]
Dostupné z: https://developer.apple.com/ios/human-interface-guidelines/overview/design-principles/
[přístup 11. 3. 2017c]

Apple: *Xcode 4 Release Notes* [online]
Dostupné z: https://developer.apple.com/library/content/documentation/Xcode/Conceptual/RN-Xcode-Archive/Chapters/xc4_release_notes.html#//apple_ref/doc/uid/TP40016994-CH3-SW2
[přístup 11. 3. 2017d]

Apple: *CFNetwork Framework* [online]
Dostupné z: https://developer.apple.com/library/content/documentation/Miscellaneous/Conceptual/iPhoneOSTechOverview/CoreServicesLayer/CoreServicesLayer.html#//apple_ref/doc/uid/TP40007898-CH10-SW14
[přístup 11. 3. 2017e]

Apple: *CoreBluetooth Framework* [online]
Dostupné z: https://developer.apple.com/library/content/documentation/Miscellaneous/Conceptual/iPhoneOSTechOverview/CoreOSLayer/CoreOSLayer.html#//apple_ref/doc/uid/TP40007898-CH11-SW6
[přístup 11. 3. 2017f]

Apple: *CoreLocation Framework* [online]
Dostupné z: https://developer.apple.com/library/content/documentation/Miscellaneous/Conceptual/iPhoneOSTechOverview/CoreServicesLayer/CoreServicesLayer.html#//apple_ref/doc/uid/TP40007898-CH10-SW3
[přístup 11. 3. 2017g]

Apple: *CoreTelephony Framework* [online]
Dostupné z: https://developer.apple.com/library/content/documentation/Miscellaneous/Conceptual/iPhoneOSTechOverview/CoreServicesLayer/CoreServicesLayer.html#//apple_ref/doc/uid/TP40007898-CH10-SW18
[přístup 11. 3. 2017h]

Apple: *Security Framework* [online]
Dostupné z: https://developer.apple.com/library/content/documentation/Miscellaneous/Conceptual/iPhoneOSTechOverview/CoreOSLayer/CoreOSLayer.html#//apple_ref/doc/uid/TP40007898-CH11-SW3
[přístup 11. 3. 2017i]

Apple: *SystemConfiguration Framework* [online]
Dostupné z: https://developer.apple.com/library/content/documentation/Miscellaneous/Conceptual/iPhoneOSTechOverview/CoreServicesLayer/CoreServicesLayer.html#//apple_ref/doc/uid/TP40007898-CH10-SW21
[přístup 11. 3. 2017j]

Apple: *prepareForSegue:sender* [online]
Dostupné z: https://developer.apple.com/reference/uikit/uiviewcontroller/1621490-prepareforsegue
[přístup 11. 3. 2017k]

Apple: *About Simulator* [online]
Dostupné z: https://developer.apple.com/library/content/documentation/IDEs/Conceptual/iOS_Simulator_Guide/Introduction/Introduction.html#//apple_ref/doc/uid/TP40012848-CH1-SW1
[přístup 28. 3. 2017l]

Bluetooth SIG**:** *Our History* [online]
Dostupné z: https://www.bluetooth.com/about-us/our-history
[přístup 11. 3. 2017a]

Bluetooth SIG**:** *Adopted specifications* [online]
Dostupné z: https://www.bluetooth.com/specifications/adopted-specifications
[přístup 11. 3. 2017b]

Couchbase: *Authentication* [online]
Dostupné z: https://developer.couchbase.com/documentation/mobile/current/guides/authentication/index.html
[přístup. 9. 3. 2017a]

Couchbase: *Document* [online]
Dostupné z: https://developer.couchbase.com/documentation/mobile/current/guides/couchbase-lite/native-api/document/index.html
[přístup. 9. 3. 2017b]

Estimote Beacon: *Need to compare our products?* [online]
Dostupné z: http://estimote.com/?gclid=Cj0KEQjw2LjGBRDYm9jj5JSxiJcBEiQAwKWAC3vceTXmsAcFtXRnCsRtpLPKef8SrawdjIdtvBW5CVEaAkMc8P8HAQ#get-beacons
[přístup. 9. 3. 2017]

iPhone Hacks: *iOS 9.3.3 Jailbreak FAQ: All Your Questions Answered* [online]
Dostupné z: http://www.iphonehacks.com/2016/08/ios-9-3-3-jailbreak-faq.html
[přístup. 11. 3. 2017]

KOCHAN, Stephen G. *Objective-C 2.0: výukový kurz programování pro Mac OS X a iPhone.* Brno: Computer Press, 2010.

MARK, Dave a Jeff LAMARCHE. *IPhone SDK: průvodce vývojem aplikací pro iPhone a iPod touch.* Brno: Computer Press, 2010.

Nordic Semiconductor**:** *A short history of Bluetooth* [online]
Dostupné z: https://www.nordicsemi.com/eng/News/ULP-Wireless-Update/A-short-history-of-Bluetooth
[přístup 10. 3. 2017]

PC World**:** *Základy technologie Bluetooth: původ a rozsah funkcí* [online]
Dostupné z: http://pcworld.cz/hardware/Zaklady-technologie-Bluetooth-puvod-a-rozsah-funkci-6635
[přístup 10. 3. 2017]

Seznam obrázků

Seznam tabulek

Seznam ukázek zdrojových kódu

Příloha 1: Obsah přiloženého CD

Níže je uveden seznam zip archivů, které se nacházejí na přiloženém CD a popis jejich obsahu.

- **ConfigFiles.zip**: 2 konfigurační soubory aplikace ve formátu JSON
- **CouchbaseDump.zip**: datový soubor Couchbase databáze obsahující naměřené hodnoty
- **RadioFprints.zip**: zdrojové kódy aplikace Radio F-Prints pro iOS (projekt pro Xcode)
- **RadioLocalization_mereni_na_miste.zip**: zdrojové kódy aplikace Radio Localization s parametry pro měření na místě (projekt pro Eclipse)
- **RadioLocalization_prochazka.zip**: zdrojové kódy aplikace Radio Localization s parametry pro měření Procházka (projekt pro Eclipse)

Univerzita Hradec Králové
Fakulta informatiky a managementu
Akademický rok: 2016/2017

Studijní program: Aplikovaná informatika
Forma: Kombinovaná
Obor/komb.: Aplikovaná informatika (ai2-k)

Podklad pro zadání DIPLOMOVÉ práce studenta

PŘEDKLÁDÁ:	ADRESA	OSOBNÍ ČÍSLO
Bc. Dědek Jan	Josefa Bublíka 367, Pardubice - Trnová	I1500702

TÉMA ČESKY:

Sběr rádiových fingerprintů na platformě iOS

TÉMA ANGLICKY:

Radio fingerprint acquisition on iOS platform

VEDOUCÍ PRÁCE:

Ing. Pavel Kříž, Ph.D. - KIKM

ZÁSADY PRO VYPRACOVÁNÍ:

Cílem práce je praktické využití otisků dostupných bezdrátových sítí pro indoor lokalizaci. Práce se zaměřuje na technologie WiFi a Bluetooth 4.0 Low Energy.
Součástí práce bude vlastní implementace sběru bezdrátových dat manuálně opatřených lokalizační značkou (souřadnicemi) pro Apple zařízení s operačním systémem iOS 10.0 nebo vyšším a jejich uložení na vzdálený server.

Osnova:
1. Úvod
2. Přehled technologií
3. Dostupná API v oper. systému iOS
4. Návrh a implementace aplikace pro sběr dat
5. Testování a výsledky
6. Závěr

SEZNAM DOPORUČENÉ LITERATURY:

https://developer.apple.com/bluetooth/
http://estimote.github.io/iOS-SDK/
https://developer.apple.com/library/content/documentation/Swift/Conceptual/Swift_Programming_Language/
Craig Gilchrist: Learning iBeacon
Matthew S. Gast: Building Applications with iBeacon, Proximity and Location Services with Bluetooth Low Energy

Podpis studenta: ..... **Datum:** 12.10.16

Podpis vedoucího práce: ..... **Datum:** 12.10.16