

UNIVERZITA PARDUBICE
FAKULTA ELEKTROTECHNIKY A INFORMATIKY

BAKALÁŘSKÁ PRÁCE

2015

Filip Němeček

Univerzita Pardubice
Fakulta elektrotechniky a informatiky

Návrh a realizace mobilní aplikace pro MHD využívající geolokaci v
operačním systému Windows Phone

Filip Němeček

Bakalářská práce

2015

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení: **Filip Němeček**
Osobní číslo: **I13184**
Studijní program: **B2646 Informační technologie**
Studijní obor: **Informační technologie**
Název tématu: **Návrh a realizace mobilní aplikace pro MHD využívající geolokaci v operačním systému Windows Phone**
Zadávající katedra: **Katedra informačních technologií**

Z á s a d y p r o v y p r a c o v á n í :

Cílem bakalářské práce bude v teoretické části popsat existující možnosti geolokace a analyzovat jejich využitelnost v mobilních aplikacích. Součástí práce bude i přehled lokačních a bezdrátových technologií a srovnání jejich parametrů, výhod a nevýhod.

Druhým cílem v praktické části bakalářské práce bude navrhnout a realizovat mobilní aplikaci pro platformu Windows Phone (verze 8.1 a vyšší), která bude s využitím geolokace (GPS a WiFi) zobrazovat aktuální odjezdy MHD (ve zvoleném městě) pro zastávku nacházející se nejbližše uživateli aplikace. Nabídne mu i možnost odeslat předvyplněnou SMS jízdenku. Umožní také plánování trasy na základě zadání cílové destinace a času. Hlavní funkcionalita bude dostupná bez nutnosti připojení k internetu. Pokud se podaří získat přístup k datům od dopravního podniku, aplikace s připojením k internetu zobrazí také aktuální zpoždění jednotlivých linek. Aplikace umožní využít rovněž rozšířenou realitu pro zobrazení polohy zastávek skrze hledáček fotoaparátu. K implementaci bude využit jazyk C# a XAML pro tvorbu uživatelského rozhraní.

Rozsah grafických prací:

Rozsah pracovní zprávy: 40

Forma zpracování bakalářské práce: tištěná

Seznam odborné literatury:

* SKEET, Jon. C# in depth. Third editions. Shelter Island, NY: Manning, 2014, xxx, 582 pages. ISBN 161729134x.

* WAGNER, Bill. Effective C#: 50 specific ways to improve your C#. Boston: Addison-Wesley, c2005, xviii, 307 p. ISBN 0321245660.

* UNGER, Russ a Carolyn CHANDLER. A project guide to UX design: for user experience designers in the field or in the making. Berkeley: New Riders, c2009, xix, 267 s. Voices that matter. ISBN 9780321607379.

* LIDWELL, William, Kritina HOLDEN a Jill BUTLER. Universal principles of design: 100 ways to enhance usability, influence perception, increase appeal, make better, design decisions, and teach through design. Gloucester: Rockport Publishers, c2003, 216 s. ISBN 1592530079.

Vedoucí bakalářské práce:

Ing. Jiří Kysela

Katedra informačních technologií

Datum zadání bakalářské práce: 31. října 2015

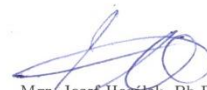
Termín odevzdání bakalářské práce: 13. května 2016



prof. Ing. Simeon Karamazov, Dr.
děkan



L.S.



Mgr. Josef Horátek, Ph.D.
vedoucí katedry

V Pardubicích dne 31. března 2016

Prohlášení autora

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

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

Souhlasím s prezenčním zpřístupněním své práce v Univerzitní knihovně.

V Pardubicích dne 10. 5. 2016

Filip Němeček

Poděkování

Rád bych poděkoval svému vedoucímu bakalářské práce panu Ing. Jiřímu Kyselovi za možnost zpracovat mnou vybrané téma a také za vstřícný přístup při zpracování této práce.

ANOTACE

Práce se zabývá návrhem a implementací Windows Phone mobilní aplikace pro městskou hromadnou dopravu v Pardubicích. Využívá prostředků geolokace pro prezentaci dat, ukazuje možnosti využití rozšířené reality a plánování spojů.

KLÍČOVÁ SLOVA

Windows Phone, geolokace, rozšířená realita, C#

TITLE

Design and implementation of a mobile application for public transport with geolocation for Windows Phone operating system

ANNOTATION

Thesis shows the design and implementation of a Windows Phone mobile application for public transport in the city of Pardubice. Application uses means of geolocation to present meaningful data, demonstrates the possibility of augmented reality and route planning through the city.

KEYWORDS

Windows Phone, geolocation, augmented reality, C#

OBSAH

0	Úvod	13
1	bezdrátové technologie	15
1.1	Celulární radiová síť	15
1.1.1	GSM	15
1.1.2	Tabulka rychlostí jednotlivých technologií	16
1.2	Wi-Fi	16
1.2.1	Pásmo a omezení	17
1.2.2	Standardy a rychlosti	17
1.2.3	Méně známé a specializované standardy	18
1.3	Bluetooth	19
1.3.1	Stručná historie	19
1.3.2	Verze Bluetooth a jejich stručné charakteristiky	20
1.4	WiMAX	21
1.4.1	Stručná historie	21
1.4.2	Princip fungování	21
1.5	ZigBee	22
1.6	Z-Wave	22
2	Geolokace	24
2.1	Úvod do geolokace	24
2.2	Nejčastější technologie pro využití geolokace	25
2.2.1	GPS	25
2.2.2	Wi-Fi	25
2.2.3	Mobilní síť	25
2.2.4	IP Adresa	25
2.3	Platforma Windows Phone a geolokace	26
2.3.1	Použití geolokace ve Windows Phone	26

2.3.2	Geofencing ve Windows Phone	27
3	Návrh a implementace řešení	28
3.1	Získání potřebných dat o hromadné dopravě	28
3.2	Návrh fungování aplikace	29
3.3	Design	29
3.4	Programové vybavení a technologie	30
3.4.1	Visual Studio	30
3.4.2	ReSharper	31
3.5	Datový model a uložení dat.....	31
3.5.1	Úvod do SQLite.....	31
3.5.2	Použití SQLite v aplikaci	32
3.5.3	Uložení zastávek pomocí statické třídy.....	33
3.6	Hlavní stránka aplikace	33
3.6.1	Implementace rozhraní INotifyPropertyChanged	34
3.6.2	Pokročilé možnosti implementace INotifyPropertyChanged.....	34
3.6.3	Získání polohy zařízení a zobrazení odjezdů	35
3.6.4	Použití systémové třídy Geolocator	35
3.6.5	Nalezení vyhovujících zastávek a odjezdů.....	36
3.6.6	Filtrování nalezených spojení.....	36
3.6.7	Zobrazení nalezených výsledků v uživatelském rozhraní.....	37
3.6.8	Navigace z hlavní stránky aplikace	37
3.6.9	Použití navigace	37
3.7	Stránka pro nákup SMS jízdenky.....	38
3.7.1	Programové posílání SMS ve Windows Phone.....	38
3.8	Stránka pro zobrazení zastávek na mapě s rozšířenou realitou	39
3.8.1	Popis rozšířené reality	39
3.8.2	Rozšířená realita a Windows Phone	39

3.8.3	Implementace GART v projektu	40
3.8.4	Spuštění GART služeb	40
3.8.5	Použití Motion API	41
3.8.6	Zpracování hodnot od Motion API	42
3.9	Stránka pro plánování trasy	42
3.9.1	Algoritmus pro hledání spojení	43
4	Ukázka hotové aplikace	44
5	Závěr	46
6	Použitá literatura	47

SEZNAM ILUSTRACÍ A TABULEK

Obrázek 1 - Ukázka vhodné implementace rozhraní <code>INotifyPropertyChanged</code>	34
Obrázek 2 - Ukázka využití operátoru <code>nameof</code>	35
Obrázek 3 - Ukázka použití <code>NavigationService</code>	37
Obrázek 4 - Příklad markeru pro iOS aplikaci HoloWatch [21]	39
Obrázek 5 - Ukázka přidání jmenného prostoru XAML	40
Obrázek 6 - Ukázka rozšířené reality a nákupu SMS jízdenek	44
Obrázek 7 - Ukázka úvodní stránky se zobrazením spojů a plánování trasy	45
Tabulka 1 - Rychlosti mobilních sítí [13]	16

SEZNAM ZKRATEK A ZNAČEK

GSM	Global System for Mobile communications
GPRS	General Packet Radio Service
EDGE	Enhanced Data rates for GSM Evolution
HSPA+	Evolved High Speed Packet Access
LTE	Long Term Evolution
IEEE	Institute of Electrical and Electronics Engineers
SIG.	Special Interest Group
USA	Spojené státy americké
GPS	Global Positioning System
IP	Internet Protocol
SDK	Software Development Kit
PDF	Portable Document Format
HTML	HyperText Markup Language
XAML	Extensible Application Markup Language
LINQ	Language Integrated Query
SMS	Short Message Service
GART	Geo Augmented Reality Toolkit
ARM	Advanced RISC Machine
DLL	Dynamic Link Library
API	Application Programming Interface
UI	User Interface
LE	Low Energy

0 ÚVOD

Chytrý telefon dnes využívají stovky milionů lidí [26]. Kromě původních funkcí mobilních telefonů slouží ty chytré jako fotoaparát, kompaktní prohlížeč webu a díky nespočtu mobilních aplikací jsou možnosti využití prakticky nekonečné. Spousta uživatelů se na svůj chytrý telefon spoléhá také při cestování a slouží jim k hledání vlakových, autobusových spojů či informací o městské hromadné dopravě (viz například počet stažení mobilní aplikace Jízdní řády IDOS [27]).

Cílem v teoretické části práce je, popsat dostupné možnosti určení polohy zařízení (tzv. geolokace) a následně analyzovat jejich využitelnost v mobilních aplikacích, speciálně potom v systému Windows Phone 8.1, pro který bude aplikace vytvořena. Součástí teoretické části bude také přehled a srovnání bezdrátových technologií.

Cílem praktické části je návrh a vytvoření mobilní aplikace vyhledávající spoje pro MHD v Pardubicích. Důraz bude kladen na intuitivní ovládání, rychlé vyhledání informací a fungování bez připojení k internetu. Tradiční postup pro nalezení nejbližších odjezdů může probíhat následovně: Uživatel otevře prohlížeč webu na svém chytrém telefonu, počká na načtení webové stránky pro vyhledávání spojů MHD. Nyní musí zadat počáteční a cílovou zastávku, kliknout na tlačítko pro hledání a čekat. V lepším případě mu počáteční zastávku vyplní webová aplikace sama pomocí určení polohy zařízení. Podobně fungují také specializované mobilní aplikace pro vyhledávání informací o městské hromadné dopravě.

Aplikace bude mít za cíl celý výše popsany postup uživateli značně zjednodušit. V klasickém scénáři bude nutné pouze aplikaci spustit a nestarat se o nic dalšího. Aplikace si zjistí polohu zařízení a z databáze všech zastávek najde tu nejbližší. Následně aplikace sama získá z databáze nejbližší odjezdy jednotlivých linek MHD a ty v přehledném rozhraní zobrazí uživateli. Uživateli tak stačilo, aplikaci spustit a pár vteřin počkat. Navíc není nutné připojení k internetu, protože jsou všechna data dostupná off-line.

Automatické zobrazení nejbližších odjezdů nebude představovat jedinou funkcionalitu aplikace. Uživateli umožní také naplánovat trasu. Celý proces bude opět velmi jednoduchý. Uživatel zvolí požadovanou cílovou zastávku a čas, do kterého chce na dané místo dorazit. Aplikace potom zobrazí několik nejlepších spojů s časem odjezdu a očekávaným časem příjezdu na vybranou cílovou zastávku. Vše bude opět fungovat bez připojení k internetu. Uživatel se tak nemusí obávat snížení funkcionality kvůli špatnému mobilnímu připojení či dalším problémům.

Kromě výše zmíněné primární funkcionality nabídne aplikace také demonstraci využití augmentované neboli rozšířené reality. Uživateli zobrazí mapu města a viditelné zastávky. Pokud uživatel uchopí mobil do svislé podoby, tak aplikace zobrazí hledáček fotoaparátu a umístění nejbližších zastávek městské hromadné dopravy. Tato funkcionality může dobře posloužit v případě, že uživatel neví, kde přesně se některá ze zastávek nachází.

Pro zvýšení komfortu nabídne aplikace snadné zakoupení SMS jízdenky pro pardubickou městskou hromadnou dopravu. V přehledném menu si uživatel vybere požadovanou jízdenku a tu následně jednoduše odešle.

1 BEZDRÁTOVÉ TECHNOLOGIE

1.1 Celulární radiová síť

Jedná se o radiovou telekomunikační síť. Komunikaci zajišťují základnové stanice, které pokrývají části území označované jako buňky. Každá buňka má minimálně jednu základnovou stanici, ve většině případů jich je ale více. Velikost buněk, která se odvíjí od rozmístění a počtu základnových stanic určuje lokalita. Městská zástavba vyžaduje mnohem větší počet buněk než volná prostranství, kde se skoro nikdo nepohybuje.

Klíčovou charakteristikou této sítě je možnost opětovného použití stejné frekvence v různých buňkách. Díky tomu lze zvýšit pokrytí a rovněž kapacitu. Pro rozlišení musí sousedící buňky používat různé frekvence, vzdálenější ale mohou stejné frekvence použít bez potíží.

1.1.1 GSM

Dnes nejrozšířenější standard pro mobilní telefony. Název standardu pochází z francouzské pracovní skupiny Groupe Spécial Mobile, která ho v roce 1982 navrhla. První generace byla analogová a umožňovala přenášet pouze hovor.

Druhá generace označovaná jako 2G přinesla přechod na digitální technologii. Digitalizace mimo jiné přinesla vyšší zvukovou kvalitu hovorů a znatelně ztlížila jejich odposlech. K dalšímu velkému rozvoji došlo u přechodné generace označované jako 2.5G. Díky technologii GPRS již bylo možné přenášet jakýkoliv druh dat a uživatelé tak dostali přístup k internetu. Nejrychlejší teoretická rychlost o GPRS dosahuje 52 Kbit/s. Zlepšení v tomto směru přinesla technologie EDGE. Hlavním cílem bylo zrychlení datových přenosů. Maximální teoretická rychlost je 384 Kbit/s, v praxi je ovšem nižší.

Třetí generace mobilní sítě představuje zatím tu nejkomplicovanější. Vývoj začal už v 80. letech minulého století a trval zhruba 15 let. Ačkoliv se jedná o přímého nástupce pro EDGE, tak funguje na zcela nových frekvencích a nasazení vyžadovalo velké změny technické infrastruktury. Cílem této generace bylo především nabídnutí dostatečných rychlostí pro streamování videa a další datově náročné činnosti. Rychlosti nejprve začaly nad hranicí 2 Mbit/s a postupně se díky vývoji technologie dostaly až ke hranici 7,2 Mbit/s v rámci specifikace HSPA.

Mezi třetí a čtvrtou generací se objevila přechodná, označována v zahraničí také jako „Pre-4G“. Zahrnuje technologie HSPA+ s rychlostí okolo 56 Mbit/s a LTE s rychlostmi dosahující teoreticky na hranici 100 Mbit/s. Mobilní operátoři často tyto technologie promovali v reklamách jako sítě 4G generace, ačkoliv se o plnohodnotnou čtvrtou generaci nejedná.

Aktuální čtvrtá generace mobilních sítí je označována pojmem LTE Advanced. Znatelného zvýšení rychlosti je dosaženo pomocí nových čipů, které musí zvládat současně až 8 příchozích spojení a poloviční počet odchozích. Díky těmto a dalším vylepšením i ze strany síťové infrastruktury nabízí tato technologie rychlosti stahování až 1 Gbit/s.

1.1.2 Tabulka rychlostí jednotlivých technologií

Technologie	Rychlost stahování	Rychlost odesílání
GPRS	114 Kbit/s	20 Kbit/s
EDGE	384 Kbit/s	60 Kbit/s
W-CDMA (3G)	2 Mbit/s	153 Kbit/s
HSPA (7.2)	7,2 Mbit/s	2 Mbit/s
HSPA+	56 Mbit/s	22 Mbit/s
LTE	100 Mbit/s	50 Mbit/s
LTE Advanced	1 Gbit/s	500 Mbit/s

Tabulka 1 - Rychlosti mobilních sítí [13]

Z pohledu této mobilní aplikace je postačující připojení EDGE pro načtení mapy města. Aplikace byla testována v emulátoru se zapnutým omezením sítě na rychlost EDGE a načtení mapy města proběhlo v postačujícím čase (2 až 4 sekundy). Vzhledem k pokrytí Pardubic sítěmi LTE a smyslu využití aplikace pouze ve městě se nepředpokládá, že by mohl nastat problém s rychlostí připojení a fungování aplikace.

1.2 Wi-Fi

Jedna z nejznámějších technologií pro bezdrátové poskytování internetového připojení počítačům, chytrým telefonům a dalším typům zařízení. Historie Wi-Fi technologie sahá až do roku 1971, kdy došlo k propojení zařízení na Havajských ostrovech pomocí systému ALOHAnet, který vyvinula tamní univerzita. Formálně lze označit rok 1999 jako rok vzniku

tohoto bezdrátového standardu. V tomto roce vznikla organizace Wi-Fi Alliance a oficiální označení Wi-Fi.

Technologie Wi-Fi využívá frekvenčního pásma, které nespadá pod licenci a není nutno vlastnit oprávnění pro jeho používání. Nejčastěji používá pásmo 2,4 GHz, v posledních letech se ale začíná objevovat také 5 GHz a to hlavně z důvodu velkého vytížení původního pásma. Kolizi Wi-Fi sítí částečně řeší dostupné kanály, jejich počet, ale především v městské zástavbě, nestačí pro vyřešení všech kolizí.

1.2.1 Pásma a omezení

Obecně je pásmo 5 GHz vhodnější do otevřených prostor a pro zařízení s kratší vzdáleností od vysílače. V tomto případě poskytne lepší přenosové rychlosti a šance na kolizi s dalšími sítěmi je nižší. Naopak pásmo 2,4 GHz se hodí pro vzdálenější zařízení a v případě, že je mezi vysílačem a přijímačem velké množství překážek.

Omezení pásem představuje další možný problém při využívání Wi-Fi sítí. Rozsah 2,4 GHz znamená v praxi prostor od 2,412 GHz do 2,484 GHz. V tomto prostoru je možné získat 14 nezávislých kanálů, které jsou od sebe odstupňovány kroky o velikosti 5 MHz. Výjimku zde představuje poslední kanál, který je vzdálen krokem 12 MHz. Ovšem každá síť si obsadí šířku pásma 20 MHz v okolí používaného kanálu. Pokud tak má dojít ke stavu, že se žádné kanály neruší, mohou současně běžet pouze 4. Jmenovitě to jsou kanály 1, 5, 9 a 13. Při domácím nastavení Wi-Fi se vyplatí použít mobilní aplikaci pro analýzu nejbližších sítí a najít nejméně vytížený kanál.

Novější pásmo 5 GHz nabízí oproti 2,4 GHz o poznání větší rozsah. Začíná na 5,180 GHz a končí až na 5,700 GHz. Kanály jsou v tomto pásmu vzdálené 20 MHz kroky, každé vysílací zařízení tak využívá vlastní a nedochází ke kolizím s ostatními. Vyšší pásmo s sebou ovšem přináší jiná omezení. Konkrétně v Evropě je dostupných 19 kanálů, navíc prvních 8 je určeno pouze pro použití v interiérech. Jejich vysílací výkon je omezen na 200 mW. Zbýlých 11 kanálů je možné provozovat s vysílacím výkonem do 1 W.

1.2.2 Standardy a rychlosti

Wi-Fi v současné době existuje v několika standardech. Následující sekce popisuje převážně ty, které je možné skutečně potkat. První standard 802.11 byl vydán v roce 1997

a představoval úplný počátek technologie. Rychlosti se tak pohybovaly velmi nízko, konkrétně mezi 1 – 2 Mbit/s, tedy maximálně asi 128 KB/s.

Krátce po zveřejnění prvního standardu byl v roce 1999 poprvé vydán nástupce v podobě 802.11a. Operoval v pásmu 5,8 GHz, aby se šlo vyhnout přeplněnému pásmu 2,4 GHz, přičemž maximální teoretická rychlost je 54 Mbit/s. Což v praxi znamená okolo 3 – 4 MB/s a ve většině případů okamžité načtení například obrázku.

Standard 802.11b se vrátil do pásma 2,4 GHz a zařízení, která ho implementovala, se objevila na trhu od roku 2000. Oproti původnímu 802.11 přinesl podstatné zrychlení s maximální propustností až 11 Mbit/s. Tento standard zároveň znamenal výrazný pokles cen zařízení a krátce poté se z 802.11b stal hlavní standard.

Červenec roku 2003 přinesl do Wi-Fi světa nový standard 802.11g. Přenosové rychlosti se vyhouply na úroveň 802.11a, ovšem za použití pásma 2,4 GHz. Nový standard opět potkal velký zájem a rychle se rozšířil, především z důvodu vyšší rychlosti. Velká část zařízení od roku 2003 podporovala všechny 3 předchozí standardy (a, b a g).

Rok 2009 byl ve znamení nového standardu 802.11n, který je velmi používaný ještě v současné době. Přinesl použití více antén pro zrychlení přenosu a také možnost operovat jak v pásmu 2,4 GHz, tak 5 GHz. Jako nové technologie a možnosti přinesl vylepšenou bezpečnost či agregaci rámců, která umožnila poslat jedním přenosem dva a více rámců a tím zvýšit propustnost. Rychlosti 802.11n se pohybují od 54 Mbit/s až po 600 Mbit/s. Záleží na kvalitě i počtu antén a terénu.

802.11ac představuje zatím poslední standard v hlavní vývojové větvi. Práce na něm probíhaly v letech 2010 až 2013, schválen byl následně v lednu roku 2014. Hlavním cílem je přinést ještě větší rychlost v pásmu 5 GHz. Teoreticky je možné s 802.11ac dosáhnout při použití 5 GHz pásma až na rychlost 2166 Mbit/s, což znamená přenosové rychlosti 270 MB/s, které stačí na veškeré on-line aktivity a jsou rychlejší než přístup k běžnému pevnému disku.

1.2.3 Méně známé a specializované standardy

Kromě běžných Wi-Fi standardů existuje celá řada méně známých a speciálně zaměřených. Například 802.11af označován jako „Super Wi-Fi“ nebo také „White-Fi“. Schválen byl v únoru roku 2014. Jeho hlavní specialitou je využití nevyužitých pásem pro televizní vysílání. Vysílače pro tento standard používají GPS pro určení své polohy a následně

internet, aby mohly zjistit informace o příslušných omezeních v dané lokalitě. Vysílání potom probíhá na těchto frekvencích. 802.11af funguje v pásmu 54 až 790 MHz. Nižší pásmo oproti běžným 2,4 GHz znamená větší dosah a signál lépe prochází stěnami či dalšími překážkami.

1.3 Bluetooth

Rovněž velmi známá a populární bezdrátová technologie. V porovnání s Wi-Fi se nezaměřuje na internetové připojení, ale slouží hlavně k výměně dat mezi zařízeními. Typicky se může jednat o chytrý telefon a příslušenství v podobě handsfree sluchátek, chytrých náramků či hodinek a dalšího bezdrátového příslušenství. U počítačů se Bluetooth využívá často pro bezdrátové klávesnice a myši.

1.3.1 Stručná historie

Historie technologie sahá do roku 1989. Technický ředitel společnosti Ericsson Mobile Dr. Nils Rydbeck a Dr. Johan Ullman měli za cíl vyvinout bezdrátovou technologii pro sluchátka. Později se objevil název Bluetooth. Oficiálně se jako rok vzniku technologie Bluetooth uvádí 1994.

Samotný název a jeho původ dokáže překvapit, nemá totiž nic společného se světem technologií. Za název je zodpovědná král Harald Bluetooth z 10. století, který sjednotil dánské kmeny pod jedno království a podle legend mezi ně ještě přinesl křesťanství. Návrh názvu poskytl autorům technologie Jim Kardach v roce 1997, v této době zrovna četl knihu *The Long Ships*, ve které vystupuje právě král Harald Bluetooth a Vikingové.

Bluetooth operuje na pásmu 2,4 GHz, konkrétně na frekvencích od 2,402 GHz do 2,480 GHz. Pásmo se tak může překrývat s Wi-Fi signály a docházet k dalšímu rušení. Kanálů je k dispozici 79, pro každý 1 MHz. Využívá se tak celé dostupné spektrum. Toto rozdělení ovšem neplatí pro standard Bluetooth Low Energy (LE), který nabízí 40 kanálů rozdělených po 2 MHz.

Nové specifikace vydává skupina Bluetooth SIG, zkratka znamená Special Interest Group. Do této skupiny patří například společnosti Intel, Ericsson, Lenovo či jako nově příchozí Apple.

1.3.2 Verze Bluetooth a jejich stručné charakteristiky

První verze 1.0 Bluetooth nabízela velmi nízké přenosové rychlosti s maximální teoretickou hodnotou 1 Mbit/s. V dnešní době již tato zařízení prakticky není možné potkat. První verze byla ještě před příchodem druhé rozšířena verzí 1.1. Přidala možnost nezabezpečených kanálů, indikaci síly signálu a také opravy některých specifikačních problémů první verze. Později navázala verze 1.2. Připojení a nacházení Bluetooth zařízení bylo rychlejší, stejně jako praktické přenosové rychlosti, které dosahovaly až na 721 Kbit/s.

Rok 2004 znamenal pro Bluetooth dlouze očekávanou druhou verzi. Přelomovou novinku představovalo jednoduché párování zařízení. Uživatel pouze zvolil, že chce zařízení spárovat a ta se sama našla a připojila. Pro počítačové periferie typu klávesnice či myš znamenala nová verze zvýšení výdrže na baterii, až pětinasobně. Teoretická rychlost verze 2.0 je 3 Mbit/s, přičemž ta praktická potom 2,1 Mbit/s. V červenci roku 2007 dorazila verze 2.1. Hlavním funkcionalitu představovalo Secure Simple Pairing (SSP). Další zjednodušení při párování zařízení a zaměřilo se hlavně na zvýšení bezpečnosti.

Třetí verze technologie označovaná jako 3.0 + HS přinesla rychlý přenos souborů. Teoretická rychlost přenosu dosahuje až na 24 Mbit/s. Rozšíření HS není povinné, právě ono se však stará o vysoké přenosové rychlosti. Zařízení s touto podporou pro poslání například velkého souboru může využít Wi-Fi antény a díky tomu dosáhnout vyšší rychlosti. Ovšem zařízení s touto technologií nejsou rozšířená.

S verzí 4.0 technologie přišla specializovaná odnož Low Energy označovaná také zkratkou LE nebo názvem Smart. Jak plyne z názvu, hlavní motivací při vývoji tohoto rozšířeného standardu byl velký důraz na nízkou spotřebu. Bluetooth LE lze také chápat jako reakci na nástup chytrých náramků a hodinek (v angličtině „wearables“), které potřebují být připojené v mnoha případech neustále.

Bluetooth LE není zpětně kompatibilní, musí ho pro fungování podporovat obě zařízení. Technologie používá softwarový model Generic Attribute Profile, označovaný zkráceně jako GATT. Komunikace je založena na modelu klient-server. Klient inicializuje GATT požadavky a přijímá odpovědi od serveru. Jako charakteristiky jsou označována data, která mezi zařízeními putují. Hodnota může být například stav baterie chytrých hodinek či zaznamenaný počet kroků.

Verze 4.1 vydaná koncem roku 2013 představuje evoluci 4.0. Vylepšení jsou hlavně pro vývojáře. Jako jedním z klíčových bodů je koexistence s LTE v mobilních telefonech. U starších verzí mohlo docházet k rušení. Tradičně byla zvýšena přenosová rychlost a výrobci dostali více možností, jak se vyhnout rušení signálu. Velkou novinkou je možnost, aby zařízení bylo spojeno s dalším a zároveň sloužilo jako hub pro ostatní klienty.

Aktuální verze 4.2 přišla asi rok po předchozí verzi 4.1. Skupina Bluetooth SIG ji vydala jako reakci na IoT (internet věcí, angl.: internet of things). Důraz je kladen na vyšší bezpečnost a ještě nižší spotřebu v režimu Bluetooth LE. Díky novému profilu se mohou koncová Bluetooth 4.2 zařízení připojit přímo k internetu a nepotřebují k tomu chytrý telefon nebo další podobné zařízení jako prostředníka.

1.4 WiMAX

Bezdrátová technologie stále ve vývoji. Jejím cílem je poskytnout rychlý internet na velké vzdálenosti. Zjednodušeně se tak dá říci, že WiMAX by mohl být pro internet něco, jako mobilní telefonní síť pro pevnou linku. Principem fungování je technologie dost podobná Wi-Fi, ovšem určením míří na větší vzdálenosti, vyšší rychlosti a více uživatelů.

1.4.1 Stručná historie

Zkratka v angličtině znamená Worldwide Interoperability for Microwave Access a technologie je definovaná v normě IEEE 802.16. Počátky tohoto standardu sice začínají v roce 1998, většina jeho zpracování a definování ovšem proběhla v letech 2000 až 2003. K publikaci první verze došlo roku 2002. První verze definovala technologii pro pásma 10 – 66 GHz. V dubnu 2003 došlo k publikaci druhé verze standardu označované jako 802.16a. Ta definovala frekvence v rozsahu 2 až 11 GHz.

1.4.2 Princip fungování

Signál pro zařízení by měl podle plánů zajišťovat speciální vysílač, podobně jako u mobilní sítě. Předpokládaný teoretický dosah hovoří až o pokrytí 8000 čtverečních kilometrů. Podobně jako u dalších vysílačů samozřejmě závisí na charakteristice terénu a počtu uživatelů. Klient by potom byl přímo notebook, chytrý telefon či jiná zařízení. Naprostá většina současných zařízení ovšem WiMAX nepodporuje, z počátku by tak uživatelé museli spoléhat na dostupné adaptéry či rozšiřující karty.

WiMAX vysílače by mohly být připojeny k internetu tradičními způsoby, jako jsou optické kabely či pomocí dalšího WiMAX vysílače. Podmínkou v tomto případě je přímá viditelnost mezi takto propojenými vysílači. Spojení mezi vysílači by potom probíhalo na frekvenci 66 GHz, což znamená mnohem méně rušení a širokou propustnost. Pro klientská zařízení by vysílače používaly frekvence v rozmezí 2 GHz až 11 GHz. Nižším frekvencím tolik nevádí překážky a není nutné udržovat přímou viditelnost. Tímto pokrytím by vysílač dokázal obsloužit zhruba 65 čtverečních kilometrů, což znamená radius asi 8 kilometrů.

1.5 ZigBee

Specifikace založena na IEEE 802.15.4 sloužící jako levná a jednoduchá alternativa k Wi-Fi a Bluetooth. Vznikla v roce 1998 a v roce 2003 došlo k její standardizaci. Tato síť funguje skrze malá rádia s velmi nízkým odběrem elektrické energie. ZigBee může posílat data až na vzdálenost 100 metrů, efektivní fungování závisí na charakteru terénu a kvalitě hardwarového vybavení.

Nejčastěji se používá v průmyslových aplikacích, kde je hlavní požadavek malá energetická náročnost a není třeba posílat velké množství dat. Typicky se jedná o sběr dat z celé řady senzorů.

Použitá frekvence závisí na lokalitě využití této technologie. V Evropě se jedná o pásmo 868 MHz, zatímco v USA a Austrálii to je 915 MHz. Čína používá 784 MHz, ve zbytku světa potom není problém nalézt ani frekvence 2,4 GHz. Maximální rychlost se odvíjí od pásma. Například u 868 MHz je to maximálně 20 Kb/s. Při použití 2,4 GHz pásma lze dosáhnout až na teoretickou rychlost 250 Kb/s.

1.6 Z-Wave

Bezdrátový protokol navržen jak alternativa k výše popsanému ZigBee. Jeho primární využití a důvod vzniku představuje automatizace domácnosti. Cílem Z-Wave je poskytnout jednoduchou a spolehlivou metodu, jak ovládat domácí systémy. Mezi ně může patřit osvětlení, garážová vrata, žaluzie či třeba ovládání termostatu a podobných částí „chytré domácnosti“. Z-Wave systém může být rovněž ovládán přes internet. Je ovšem zapotřebí speciálního centrálního zařízení.

Návrh pro použití k ovládání domácího systému se odráží na maximálních přenosových rychlostech. V současné době je možné dosáhnout propustnosti 40 Kb/s (starší Z-Wave čipy zvládaly maximálně 9 Kb/s). Maximální dosah je zhruba 30 metrů a v Evropě Z-Wave operuje na frekvenci 868.42 MHz.

2 GEOLOKACE

2.1 Úvod do geolokace

Jako geolokace se označuje metoda sloužící pro určení geografické pozice vybraného objektu. Může jít o počítač, mobilní telefon či podobná zařízení. V mnoha případech není nutné znát přesnou polohu a často stačí pouze přibližná. Například určení konkrétního města, kde se daný uživatel lokalizovaného zařízení nachází nebo může postačovat určení státu.

Výstupem metody jsou nejčastěji zeměpisné souřadnice. Slouží k jednotnému označení polohy. Zeměpisná šířka je určena jako uhlová vzdálenost od rovníku, zeměpisná délka potom označuje úhlovou vzdálenost od nultého poledníku. Nultý poledník prochází londýnským obvodem Greenwich.

Získané údaje lze následně využít pro přizpůsobení obsahu, cílení reklamy a dalších činností, u kterých významně pomáhá znalost polohy. Určení, ze kterého státu počítač pochází, může posloužit k nastavení patřičného jazyka webové stránky, případně nastavení použité měny v případě e-shopu.

Určení polohy přesněji může sloužit u služeb operujících v daném městě. Mobilní či webová aplikace tak může uživateli nabídnout relevantní obsah vzhledem k jeho aktuální pozici. Například automaticky vyplnit zastávku odjezdu při hledání vlakových spojení. Stejně tak může zjištění polohy sloužit ke geografickému omezení digitálního obsahu (filmy, hudba), na který se v každém státě vztahují jiné zákony.

Geolokaci lze využít také při odhalování finančních podvodů na internetu. Pokud je během krátkého časového horizontu využita platební karta na nákup suší v japonské metropoli a nového počítače v New Yorku, pro finanční instituci jde o velmi silný náznak, že mohlo dojít ke kompromitaci dané karty.

Zaznamenání polohy může být užitečné i u běžné věci, jako je pořizování fotografií. Dnešní chytré telefony a také digitální fotoaparáty umožňují přidat k fotkám také místo pořízení. Uživatel tak později vidí, kde daný snímek pořídil, případně může využít dostupné aplikace pro vizualizaci na mapě.

2.2 Nejčastější technologie pro využití geolokace

2.2.1 GPS

Neboli Global Positioning System (česky označován jako Globální polohovací systém). Jedná se o zdaleka nejpřesnější metodu, zároveň ale také o tu nejpomalejší. Využívá systém 24 satelitů a dokáže tak poskytnout polohu s vyšší přesností než 1 metr. V současné době mají GPS prakticky všechny chytré telefony a pro přesné určení polohy se tak jedná o nejlepší možnost.

2.2.2 Wi-Fi

Polohu lze „odhadnout“ pomocí známých Wi-Fi sítí v dosahu zařízení. Ve spojení se známou polohou statických Wi-Fi sítí dokáže rychle poskytnout přibližnou polohu zařízení. Výhoda může spočívat ve využití v interiérech, kde nemusí slabé GPS přijímače mobilních telefonů postačovat.

2.2.3 Mobilní síť

U zařízení s připojením k mobilní síti lze zjistit přibližnou polohu tímto způsobem. Každé zařízení se pravidelně hlásí nejbližším vysílačům a pomocí techniky triangulace založené na síle signálu lze dané zařízení lokalizovat. Chytré telefony často kombinují tento způsob s GPS. Přesná poloha je díky tomu nalezena rychleji a prakticky okamžitě je k dispozici alespoň přibližná.

2.2.4 IP Adresa

Zdaleka nejméně přesná metoda pro určení polohy. Používá se hlavně v případě, že lepší nejsou k dispozici. Polohu se snaží zjistit podle přidružení dané IP adresy k danému poskytovateli internetového připojení. Výsledkem je tak často nejbližší velké město, které poskytovatel internetu využívá jako uzel. Avšak použitím připojení proxy může dojít k ještě větším nepřesnostem a poloha se může odchýlit i stovky kilometrů.

2.3 Platforma Windows Phone a geolokace

Vývoj mobilních aplikací pro platformu Windows Phone se odvíjí od zvolené verze. V době psaní práce byla aktuální verze systému 8.1, kterou společnost Microsoft nahradila dřívější verzí 8.0.

Pro Windows Phone 8.1 lze aplikace programovat ve dvou odlišných technologiích. Windows Phone 8.1 Silverlight představuje pokračování technologie založené na verzích systému 8.0 a předchozích. Naopak Windows Phone 8.1 Runtime je odnoží technologie pro Windows 8 aplikace, které je možné nabízet v oficiálním obchodě Windows Store.

Teoretická analýza a praktická část této práce se věnuje Windows Phone 8.1 Silverlight.

2.3.1 Použití geolokace ve Windows Phone

Získávání polohy je při použití možností platformy Windows Phone 8.1 Silverlight přímočaré a jednoduché. Bohužel v jednoduchosti se ukrývá také poměrně velké omezení. Programátor nemůže podrobněji měnit chování systému pro geolokaci. Interně systém spoléhá na GPS data a vypomáhá si pomocí připojení k mobilní síti. Ve druhém případě je získání přesné pozice znatelně rychlejší.

Všechny relevantní třídy SDK jsou zahrnuty ve jmenném prostoru `Windows.Devices.Geolocation`. Pro získání polohy slouží dostupná třída `Geocator`. Po vytvoření její instance je možné použít asynchronní metodu `GetGeopositionAsync`. Jako parametry tato metoda přijímá datový typ `TimeSpan` pro určení toho, jak stará mohou být poziční data. Druhý datový typ je opět `TimeSpan`, v tomto případě určuje, jak dlouho má metoda čekat na získání dat o pozici. Instanci třídy `Geocator` je možné nastavit požadovanou přesnost v metrech. Nastavení této hodnoty může mít vliv na rychlost získání pozičních souřadnic. Hlavně v případě, že není požadována vysoká míra přesnosti.

Třída `Geocator` programátorovi navíc poskytuje možnost, jak sledovat případné změny pozice. Pro tento účel je k dispozici událost `PositionChanged`, která je vyvolána vždy, když se změní pozice. Programátor může chování této události nastavovat pomocí vlastností třídy `Geocator`. Jedná se o vlastnost `MovementThreshold`, která v metrech určuje, o kolik se musí změnit pozice, aby došlo k vyvolání této události. Společně s ní existuje vlastnost `ReportInterval` v milisekundách, pomocí ní je možné specifikovat, jak často má docházet ke

kontrole pozice. Nižší hodnota znamená vyšší přesnost a rychlejší reakci na změnu polohy, zároveň ale také může vést k podstatnému zvýšení energetické náročnosti aplikace.

Výstupem výše zmíněné metody je instance třídy `GeoCoordinate`. Z ní je možné přistupovat k samotným získaným souřadnicím.

2.3.2 Geofencing ve Windows Phone

K využití geolokace pomocí systémové třídy `Geolocator` se pojí možnosti využití technologie označované jako geofencing. Nejčastěji je tato možnost využívána na pozadí, když není daná aplikace aktivně používána uživatelem. Jako příklad může posloužit aplikace na připomínky, která využívá znalost polohy zařízení. Uživatel si tak například může nastavit, aby mu jeho chytrý telefon připomněl nakoupit, když se bude zrovna nacházet poblíž relevantního obchodu.

Platforma Windows Phone nabízí možnosti pro geofencing od verze 8.1. Systémové třídy pro tuto funkcionalitu se nachází ve jmenném prostoru `Windows.Devices.Geolocation.Geofencing`. Dvě nejdůležitější jsou `Geofence` a `GeofenceMonitor`. Třída `Geofence` slouží ke specifikaci míst, na která chce být programátor upozorněn pomocí třídy `GeofenceMonitor`.

Geofencing nelze využít při programování Windows Phone 8.1 Silverlight aplikací. Programátor musí sáhnout po Windows Phone 8.1 Runtime a využít připravenou komponentu Windows Runtime Component pro běh na pozadí a následnou implementaci těchto možností.

3 NÁVRH A IMPLEMENTACE ŘEŠENÍ

3.1 Získání potřebných dat o hromadné dopravě

Pro off-line fungování aplikace bylo zapotřebí získat podrobné informace o odjezdech jednotlivých spojů ze zastávek městské hromadné dopravy. V minulosti bylo možné získat tato data ve formátu .xls z webových stránek idos.cz [19]. Společnost CHAPS s.r.o., která data poskytuje, ovšem změnila na konci roku 2015 podmínky a nyní nabízí data pouze ve formátu PDF, který není vhodný pro strojové zpracování.

Jako zdroj dat tak nakonec posloužily oficiální webové stránky Dopravního podniku města Pardubic a.s. V sekci Zastávkové jízdní řády jsou k dispozici jízdní řády pro všechny linky fungující ve městě. Pro jejich získání byla vytvořena jednoduchá C# aplikace pro stažení a zpracování dat. Výstupem byly textové soubory se všemi dostupnými spojeními pro každou linku a zastávku. Vyskytl se problém, protože jízdní řády neobsahují čas příjezdu na poslední zastávku linky, který je v aplikaci potřeba pro část funkcionality. Bylo tak nutné data následně upravit. Pro každou linku tak byla pomocí webových stránek Idos.cz zjištěna časová vzdálenost předposlední a poslední zastávky a doplněna mezi výsledná vstupní data pro aplikaci.

Pro aplikaci bylo dále třeba připravit seznam všech zastávek městské hromadné dopravy v Pardubicích a přilehlých obcích. Po delší analýze, jak tato data získat (název zastávky a geografické souřadnice), byl nakonec zvolen manuální postup s využitím Mapy.cz. Nejdříve byl vytvořen seznam všech zastávek z již dostupných dat z předchozího kroku a následovalo ruční vyhledávání v mapových podkladech. Celý úkol byl nakonec hotový rychleji, než kolik zabrala předchozí analýza možného automatizovaného řešení. Ze situace plyne ponaučení, že ne vždy se vyplatí zkoušet vše automatizovat.

Vstupní data pro kompletní fungování aplikace tedy tvoří seznam všech zastávek městské hromadné dopravy včetně jejich geografických souřadnic. Uspořádané seznamy posloupnosti názvů zastávek pro každou linku a jednotlivá spojení. U každého spojení je evidováno identifikační číslo zastávky, čas odjezdu, jaké dny jezdí (pracovní, víkendy a státní svátky či letní prázdniny) a jako poslední také směr daného spoje.

Celkově nakonec nebylo shánění potřebných dat tak těžké, jak se z počátku zdálo. Ovšem v případě přípravy podobné aplikace pro všechna města České republiky, kde funguje

městská hromadná doprava, by se tento postup nepochybně značně protáhl a bylo by lepší získat přístup k centralizovaným datům, pokud taková data vůbec existují.

3.2 Návrh fungování aplikace

Na základě prvotní analýzy byly formulovány základní požadavky na výslednou mobilní aplikaci.

- Získat údaje o poloze.
- Umožnit vyhledávání z uložených spojení.
- Implementovat rozšířenou realitu pro zobrazení zastávek.
- Umožnit plánování trasy po zadání času a cílové zastávky.
- Většina funkcionality musí být dostupná bez připojení k internetu.

Základní fungování a ovládání aplikace bylo rozděleno na čtyři části. Zobrazení odjezdů od nejbližší zastávky, plánování trasy, zobrazení mapy města se zastávkami a možnost zakoupení SMS jízdenky.

Pro větší přehlednost pro koncového uživatele a také oddělení pro přehlednější implementaci bylo rozhodnuto, že každá z výše zmíněných funkcionalit bude mít v aplikaci vlastní stránku. Zobrazení odjezdů spojí z nejbližších zastávek přitom bude představovat hlavní část rozhraní aplikace a tato stránka bude zobrazena vždy po spuštění. Rozhodnutí vychází z očekávání, že právě tato funkcionality bude koncovými uživateli nejčastěji využívána.

Hlavní stránka s odjezdy bude zároveň sloužit k navigaci na ostatní stránky v aplikaci. Zároveň se na ní bude vždy uživatel vracet, pokud na jiné stránce zvolí možnost zpět.

3.3 Design

Cílem designu je, co nejvíce se přiblížit stylu platformy, pro kterou aplikace vzniká, tedy Windows Phone ve verzi 8.1. Mobilní systém Windows Phone již od své první verze klade důraz na maximální přehlednost a zvýraznění obsahu nad ostatními designovými prvky.

Aplikace z tohoto důvodu bude používat stejné komponenty uživatelského rozhraní, jaké je možné vidět při používání tohoto systému. Rovněž zachová typ písma a barevné

schéma. V případě Windows Phone je dominantní barvou černá a obsah v podobě textu či ovládacích prvků je uživateli zobrazen v bílé. Případně je možné v nastavení systému tyto dvě barvy prohodit.

Důležitým designovým prvkem systému Windows Phone je tematická barva, kterou si uživatel může zvolit v nastavení. Barva se následně promítne napříč uživatelským rozhraním systému, především potom na úvodní obrazovce Start, kde jsou touto barvou v základu vykresleny dlaždice (tyto prvky používá Windows Phone místo ikoněk z ostatních systémů). Aplikace tak bude používat uživatelem nastavenou tematickou barvu pro osvěžení uživatelského rozhraní a navození dojmu, že se prakticky jedná o část softwaru telefonu.

3.4 Programové vybavení a technologie

Aplikace byla vytvořena pomocí programovacího jazyka C# v aktuální šesté verzi a technologie .NET Framework ve verzi 4.6. Jazyk C# nabízí programátorovi komfort moderního objektově orientovaného jazyka s důrazem na typovou strukturu. Tyto vlastnosti umožňují, aby kompilátor pomohl odhalit drobné chyby vznikající překlapy. Windows Phone aplikace je možné psát také v jazycích Visual Basic, C++ či pomocí kombinace webových technologií HTML5 a Javascript. Jedná se ale spíše o okrajové možnosti a C# představuje nejčastější volbu. Zvolení nejpoblárnějšího jazyka pro tento typ aplikací znamená snadnější hledání on-line rad pro řešení problémů a usnadňuje tak vývoj celé aplikace.

Pro tvorbu uživatelského rozhraní používají Windows Phone aplikace značkovací jazyk XAML. Vychází z obecného jazyka XML a dovoluje deklarativní tvorbu uživatelského rozhraní, včetně nastavení vlastností jednotlivých komponent. Výhodou použití C# a XAML v jednom projektu je automatické oddělení uživatelského rozhraní od samotné funkcionality a následné zpřehlednění vývoje. Uživatelské rozhraní jde ovšem tvořit také skrze jazyk C#, tato možnost se ale moc často nepoužívá. Je však nutností při dynamickém tvoření uživatelského rozhraní za běhu aplikace.

3.4.1 Visual Studio

Aplikace byla vytvořena pomocí vývojářského prostředí Visual Studio v aktuální verzi 15 a v edici Community Edition. Tato edice je zdarma i pro komerční účely a představuje komplexní řešení, jak vyvíjet na platformě .NET (nabízí ale rovněž podporu pro Python a další jazyky).

Visual Studio poskytuje skvělé možnosti pro vývoj Windows Phone aplikací. Nabízí rychlé emulátory s několika velikostmi displeje založených na virtualizační technologii Hyper-V. Integrovaný designer pro XAML v reálném čase vykresluje uživatelské rozhraní a programátor tak nemusí při každé drobné změně zapínat aplikaci, aby viděl nové změny. Součástí Visual Studia je mezi mnoha nástroji také klient pro Visual Studio Team Services. Jedná se o službu pro verzování projektů, která byla při vytváření této mobilní aplikace rovněž využívána.

3.4.2 ReSharper

Jako vítaný pomocník při vývoji se osvědčilo rozšíření ReSharper pro Visual Studio od firmy JetBrains. ReSharper nabízí komplexní možnosti při refaktorování kódu, jako je přejmenování názvů objektů, extrakce bazové třídy či rozhraní a celkově usnadňuje nejběžnější úkony. Rozšíření navíc hlídá dodržování syntaktických konvencí jazyka C#, jako je název metod začínající velkým písmem, podtržítka u privátních referencí či psaní statických instancí velkým prvním písmenem. Umí také zjednodušovat podmínkové výrazy, volání metod technologie LINQ a celou řadu dalších nápomocných funkcí. Používán byl v dostupné studentské licenci.

3.5 Datový model a uložení dat

Z předchozí analýzy vyplynulo, že aplikace musí mít přístup k seznamu jednotlivých zastávek pardubické městské dopravy, jednotlivým spojením a také posloupnosti zastávek pro jednotlivé linky. Na základě charakteru dat bylo rozhodnuto o využití databáze pro jednotlivá spojení hromadné dopravy a zároveň použití statické kolekce v paměti pro jednotlivé zastávky. Zastávek je relativně malý počet (187) a navíc se nemění.

Výběr databáze pro off-line použití je v případě mobilního zařízení znatelně omezen v porovnání s webovými nebo desktopovými aplikacemi. Po provedení analýzy bylo rozhodnuto o použití populární databáze SQLite.

3.5.1 Úvod do SQLite

SQLite představuje relační databázový systém. Obsažen je v malé knihovně napsané v jazyku C a vyvíjí ho Richard Hipp společně s týmem. Šířen je pod licencí public domain, což znamená, že autor své dílo uvolnil k dalšímu použití bez jakýchkoliv nároků na další

ochranu díla. Tento databázový systém vyniká snadným použitím, nevyžaduje konfiguraci a přitom nabízí bohaté možnosti. Autoři SQLite se na oficiálním webu chlubí [20], že jejich výtvar používají takoví giganti, jako jsou Adobe, Apple, Microsoft, Google, Facebook a celá řada dalších velkých jmen. Časté využití je právě uvnitř mobilních aplikací.

Pro použití SQLite ve Windows Phone 8.1 projektu je třeba použít rozšiřující balíček nazvaný SQLite for Windows Phone 8.1. Je zdarma a dostupný ke stažení přímo z oficiální galerie doplňků pro Visual Studio [28]. Instaluje se přímo do Visual Studia, ne do projektu, jak je zvykem u ostatních balíčků. Výhoda spočívá v tom, že ho lze snadno použít znovu v dalším projektu. Po instalaci je ovšem nutno provést dodatečnou konfiguraci a rozšíření přidat ještě do konkrétního projektu.

Jednotlivá spojení jsou v aplikaci modelována třídou Connection. Ta obsahuje identifikační číslo, identifikaci zastávky, hodinu a minutu odjezdu, typ spojení, číslo linky a konečně směr.

3.5.2 Použití SQLite v aplikaci

Použití třídy Connection s SQLite databází je velice jednoduché. V první řadě je třeba definovat třídu, která bude sloužit jako rozhraní pro databázi a zároveň jako model jednotlivých tabulek. Nutné je dědit ze systémové třídy DataContext a jejímu konstruktoru poskytnout cestu k databázi (kterou při prvním spuštění sám vytvoří). Jednotlivé tabulky jsou potom v této třídě definovány jako Table. Takže v případě spojení je to Table<Connection>. Na základě toho potom SQLite při inicializaci vytvoří tabulky. Přístup do tabulky je realizován technologií LINQ, která umožňuje snadnou práci s databází bez explicitního psaní SQL příkazů. Nabízí například metodu Where, které je možné předat lambda výraz pro filtrování, metodu OrderBy pro seřazení výsledků a spoustu dalších.

Vytvoření sloupců probíhá na základě atributů tříd, které mají modelovat entity v tabulkách databáze. Vlastnosti stačí označit atributem Column a SQLite pro takto označenou vlastnost vytvoří sloupec v databázi datového typu, který nejlépe odpovídá datovému typu z jazyka C#. Případně lze přidat další konfiguraci, jako například automatické generování identifikačního čísla. Data spojení jsou načtena do databáze při prvním spuštění aplikace z textových souborů uložených v projektu ve složce InitData/Connections. Textové soubory jsou rozděleny podle čísla linek, které je rovněž uloženo do databáze pro každý řádek se spojením.

3.5.3 Uložení zastávek pomocí statické třídy

Jednotlivé zastávky byly implementovány jako statická instance třídy `Dictionary<int, Station>`. Jako klíč slouží identifikační číslo zastávky a hodnota je instance třídy `Station`. Třída pro modelování zastávky obsahuje název, číslo a geografické souřadnice. Kód pro třídu `Stations`, která zastávky obsahuje, byl vygenerován automaticky z dříve získaných dat. Pro usnadnění práce při plánování trasy obsahuje třída `Stations` ještě druhé statické `Dictionary`. Slouží k převodu názvu zastávky na identifikační číslo. Z názvu zastávky se tak lze pomocí obou `Dictionary` dostat na kompletní instanci téže zastávky.

Jednotlivé posloupnosti zastávek pro každou linku a směr jsou načítány za běhu aplikace podle potřeby. K jejich načtení a ukládání do cache ve formě instance kolekce `Dictionary` zajišťuje třída `StationChainsCache`. Od této třídy je možné pomocí metody `GetChainForLineNumberAsync` vyžádat posloupnost názvů zastávek podle zadaného čísla linky a směru. Pokud už byly jednou tyto zastávky načtené, tak se pouze vrátí uložená instance třídy `List<string>`, v opačném případě dojde k načtení zastávek z textového souboru pro každou linku. Textové soubory jsou uloženy v projektu ve složce `InitData/Stations`. Pro plánování tras umožňuje tato třída načíst všechny zastávky dopředu pomocí metody `PreloadAllStationsAsync`. Jakmile dojde k načtení všech, je rovněž spuštěna událost `AllStationsPreloaded`, takže klient může na dostupná data dále reagovat.

3.6 Hlavní stránka aplikace

Na hlavní stránce aplikace zobrazuje uživateli odjezdy městské hromadné dopravy ze zastávky, která se nachází nejbližší. K zobrazení jednotlivých položek je použita kontrolka `LongListSelector`, dostupná z balíčku `Windows Phone Toolkit`, který vydává přímo Microsoft jako rozšíření pro vývojáře aplikací pro tuto platformu.

O možnosti navigace se stará komponenta `AppBar`, kterou používá také přímo operační systém. Uživatel má tak k dispozici známý prvek, pomocí kterého se může v aplikaci pohybovat. Kromě tlačítka na obnovení zobrazených spojení obsahuje tlačítka pro přechod na další stránky, které budou popsány v následujících sekcích.

Pro propojení uživatelského rozhraní definovaného v jazyku XAML s programovou C# implementací je využívám návrhový vzor MVVM (Model-View-ViewModel). Funkcionalitu hlavní stránky zajišťuje přidružená třída `MainVm`. Data jsou zobrazovány

pomocí techniky data binding. Pro její fungování jsou v XAML definovány vlastnosti, které si má rozhraní načíst z instance třídy nastavené jako DataContext. Rozhraní tedy zobrazuje data určených vlastností instance třídy MainVm.

3.6.1 Implementace rozhraní INotifyPropertyChanged

Rozhraní INotifyPropertyChanged představuje velmi důležitou součást při používání data binding. Ve výchozím stavu totiž data binding neřeší aktualizaci dat, která zobrazuje. Pokud tedy dojde ke změně dat v určitém view modelu, tak není tato změna provedena v uživatelském rozhraní. V projektu je rozhraní implementováno abstraktní třídou PropertyChangedBase, aby ho bylo možné snadno používat se všemi view model třídami.

```
public abstract class PropertyChangedBase : INotifyPropertyChanged
{
    public event PropertyChangedEventHandler PropertyChanged;

    protected void OnPropertyChanged([CallerMemberName] string name = null)
    {
        PropertyChangedEventHandler handler = PropertyChanged;
        handler?.Invoke(this, new PropertyChangedEventArgs(name));
    }
}
```

Obrázek 1 - Ukázka vhodné implementace rozhraní INotifyPropertyChanged

Kromě standardních konstrukcí stojí za povšimnutí atribut CallerMemberName před parametrem metody OnPropertyChanged, která upozorňuje na změnu dat a potřebu změnit uživatelské rozhraní. Potřeba je totiž předat, jaká vlastnost daného view modelu byla změněna. Díky atributu název vlastnosti doplní runtime jazyka C# a programátor tak může metodu volat bez předání parametru.

3.6.2 Pokročilé možnosti implementace INotifyPropertyChanged

V souvislosti s INotifyPropertyChanged se vyplatí myslet na nový operátor C# 6.0 a to konkrétně nameof. Vrací textový název daného objektu. V minulosti bylo třeba specifikovat názvy pomocí konstant typu string přímo v kódu, pokud ale došlo ke změně názvu dané vlastnosti, konstanta většinou zůstala nezměněná a data binding následně nefungoval korektně. Explicitní použití parametru se používá v případě, kdy některé vlastnosti dané třídy závisí na jiných a je nutné je aktualizovat současně.

```

...public bool IsPlanningVisible
...{
...    get { return _isPlanningVisible; }
...    set
...    {
...        _isPlanningVisible = value;
...        OnPropertyChanged();
...        OnPropertyChanged(nameof(IsResultsVisible));
...    }
...}

```

Obrázek 2 - Ukázka využití operátoru nameof

3.6.3 Získání polohy zařízení a zobrazení odjezdů

K poloze zařízení je přístupováno skrze vytvořené rozhraní `IGeoServiceProvider`. Pro tvorbu vlastního rozhraní bylo rozhodnuto z důvodu testování aplikace v domácích podmínkách, což znamená hlavně jinou geografickou polohu. Rozhraní je velmi jednoduché. Definuje asynchronní metodu `GetCurrentLocationAsync`, která vrací modelovou třídu `GeoLocation`, v ní je obsažena zeměpisná šířka a délka. Druhou a zároveň poslední metodou tohoto rozhraní je `SetMaxCacheAge`. Nastavuje hodnotu, jak stará mohou být získaná data o poloze, aniž by bylo nutné provádět jejich aktualizaci.

Výše popsané rozhraní využívá v projektu třída `GeoLocatorWrapper`, která obsahuje implementaci systémové třídy `GeoLocator`. Druhou třídou implementující stejné rozhraní je `FakeGeoServiceProvider`. Slouží pro testovací a vývojové účely aplikace a vždy vrací stejnou polohu pro Masarykovo náměstí v Pardubicích.

3.6.4 Použití systémové třídy `Geolocator`

Třída `Geolocator` byla popsána v teoretické části práce. Její využití v aplikaci je velmi přímočaré. K vytvoření instance dochází v konstruktoru třídy `GeoLocatorWrapper`, která zároveň nastavuje instanci třídy `Geolocator` některé vlastnosti. Jmenovitě to je přesnost označovaná jako `DesiredAccuracyInMeters`. Hodnota byla zvolena 50 metrů, protože aplikace pro nalezení nejbližší zastávky, vzhledem k jejich rozmístění, nepotřebuje znát co možná nejpřesnější polohu. Díky tomuto nastavení rovněž samotné hledání polohy může proběhnout rychleji.

Pro získání polohy od třídy `Geolocator` je nutné využít asynchronní metodu `GetGeopositionAsync`. Parametry jsou maximální stáří dříve získané polohy, která ještě vyhovuje a nastavení času `timeout`, po kterém dojde k přerušení geolokace. Jako maximální

stáří polohy byl určen čas 1 minuta. Předpokládá se, že během této doby průměrný uživatel nezmění svou polohu tak významně, aby takto získaný údaj nevyhovoval potřebám aplikace.

3.6.5 Nalezení vyhovujících zastávek a odjezdů

Po úspěšném získání souřadnic zařízení je zavolána metoda `FindClosestStopAsync`. Na vstupu přijímá instanci třídy `GeoLocation`. Z dostupných zastávek ve statické třídě `Stations` pomocí technologie LINQ To Objects vypočítá pro každou zastávku vzdálenost od zařízení a vrátí kolekci anonymních objektů, které obsahují vždy zastávku a vzdálenost, pro které platí, že je vzdálenost menší než konstanta 300 metrů. Pokud nejsou nalezeny žádné zastávky, tak se aplikace uživatele zeptá, jestli si přeje rozšířit vyhledávání na okruh 700 metrů. Pokud ani po této změně aplikace nenalezne žádné zastávky, tak na tuto skutečnost uživatele upozorní. V budoucnu by nebyl problém dát uživateli možnost, aby si vybral, jak vzdálené zastávky chce zobrazit.

Z nalezených zastávek vybere tu nejméně vzdálenou a pro ni začne vyhledávat spojení pro aktuální čas a typ dne. Zbylé zastávky, pokud nějaké existují, jsou uloženy do neaktivní nabídky, kterou může uživatel vyvolat a vybrat si jako startovací zastávku jinou, než kterou automaticky zvolila aplikace podle nejkratší vzdálenosti.

Nejbližší nalezená zastávky je předána metodě `UpdateUIForClosestStationAsync`. Metoda nastaví tuto zastávku jako výchozí, zobrazí uživateli její název v uživatelském rozhraní a také zavolá další metodu `UpdateNextConnectionsAsync` pro nalezení odjezdů. Tato metoda získá aktuální čas skrze rozhraní `ITimeProvider`. Následně ještě zjistí vyžadovaný typ spojení skrze rozhraní `IConnectionTypeProvider`. Spojení jsou totiž rozdělena na ty, která jezdí všední dny, víkendy a svátky a třetí skupinu představují letní prázdniny. U obou výše zmíněných rozhraní jsou využívány také testovací implementace, které vždy vrací stejné údaje.

3.6.6 Filtrování nalezených spojení

Po získání potřebných údajů získá metoda z databáze pomocí LINQ spojení, u kterých vyhovuje identifikační číslo zastávky a jejich typ. Následně jsou filtrována pomocí dodatečné metody `GoesInNextXMinutes`, která porovná čas spojení s aktuálním časem, jestli je menší než zadaných X minut formou parametru. Ve výchozím stavu je nastaveno 15 minut.

3.6.7 Zobrazení nalezených výsledků v uživatelském rozhraní

Jako poslední akci metoda každému vyhovujícímu spojení nastaví textově všechny následující zastávky, které získá pomocí výše popsané třídy StationChainsCache. Názvy všech zastávek jsou spojeny pomocí systémové metody string.Join, která mezi ně přidá znak „->“ a vytvoří z nich jeden řetězec textu. Výsledek je potom zobrazen uživateli, aby měl přehled, na jaké zastávky se může s jakým spojem dostat. Zobrazení zastávek probíhá velmi jednoduše pomocí instance kolekce ObservableCollection, které jsou nalezená spojení předána a data binding se postará o jejich zobrazení uživateli v uživatelském rozhraní.

3.6.8 Navigace z hlavní stránky aplikace

Jak již bylo zmíněno v sekci o designu aplikace, pro hlavní navigaci slouží systémová komponenta ApplicationBar. ApplicationBar je na hlavní stránce aplikace implementován skrze jazyk XAML. Každé tlačítko má definovaný popisek, ikonku a událost, která se stane po jeho stisknutí.

Tlačítko obnovit je jediné, které neslouží pro navigaci na další stránky aplikace. Uživateli umožňuje manuálně znovu zahájit proces vyhledání lokace zařízení, zobrazení nejbližší zastávky a konečně nalezené odjezdů pro daný čas.

Zbýlá trojice tlačítek umožňuje navigovat v aplikaci na stránku s mapou města, kde je implementovaná rozšířená realita, stránku se zakoupením SMS jízdenek a poslední tlačítko slouží k navigaci na stránku obsahující plánování trasy.

3.6.9 Použití navigace

Navigování na jinou stránku je ve Windows Phone poměrně přímočaré. Připravena je metoda Navigate třídy NavigationService, která je dostupná na každé stránce. Jako parametr vyžaduje Uri instanci, skládá se z cesty k XAML souboru dané stránky a určení, jestli se jedná o relativní nebo absolutní zdroj. V případě aplikace jsou stránky vždy určeny jako relativní.

```
..private void MapAppBar_Click(object sender, EventArgs e)
..{
.....NavigationService.Navigate(new Uri("/Pages/MapPage.xaml", UriKind.Relative));
..}
```

Obrázek 3 - Ukázka použití NavigationService

3.7 Stránka pro nákup SMS jízdenky

Posílání SMS je ve Windows Phone velmi jednoduché. Microsoft při návrhu operačního systému dbal na jeho bezpečnost a vývojáři tak nedostali žádnou možnost, jak poslat SMS bez toho, aniž by o tom věděl uživatel zařízení. Podobně funguje také populární systém iOS od Apple.

Místo toho aplikace požádá systém o poslání SMS (či provedení hovoru) a ten dle instrukcí zobrazí uživateli standardní dialog pro posílání SMS. Pokud uživatel poslání SMS zprávy nepotvrdí, tak není v programátorových silách, aby SMS odeslal.

Na stránce pro objednání SMS jízdenky jsou tak tři velká tlačítka, pro všechny možnosti, jak SMS jízdenku objednat. Dopravní podnik města Pardubic nabízí buď jízdenku na 45 minut (o víkendu a svátky 60 minut) za 25 Kč nebo na celý den za 65 Kč. Jiné možnosti nejsou. Třetí možnost je v aplikaci pro zaslání duplikátu již zakoupené jízdenky, pokud by si ji například uživatel omylem smazal ze zařízení.

3.7.1 Programové posílání SMS ve Windows Phone

Jak již bylo zmíněno výše, programátor musí v aplikaci požádat systém, aby poslání SMS vyřešil za něj a uživatel poslání SMS musí potvrdit. K tomuto účelu je připravena systémová třída `SmsComposeTask`.

V aplikaci tuto třídu používá view model `SmsTicketVm`, který je přidružen ke stránce s výběrem SMS jízdenek. Pro vytvoření požadavku na zaslání SMS stačí vytvořit instanci třídy `SmsComposeTask`, které je v aplikaci předáno číslo a text zprávy.

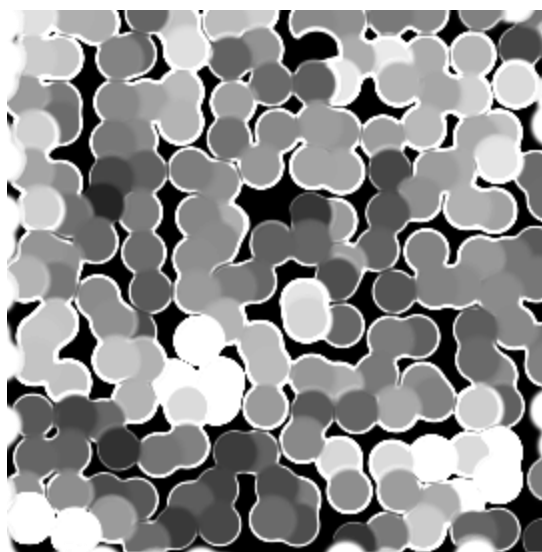
Aktivace této instance pomocí její metody `Show` způsobí opuštění aplikace a zapnutí systémové aplikace pro posílání a přijímání SMS, kde je předvyplněno poslání SMS jízdenky. Pokud uživatel souhlasí, stiskne odeslat a systém sám SMS pošle.

3.8 Stránka pro zobrazení zastávek na mapě s rozšířenou realitou

3.8.1 Popis rozšířené reality

Rozšířená či také augmentovaná realita, je atraktivní způsob, jak do fyzického světa přenést virtuální prvky. V první řadě je vyžadováno, aby se uživatel takového řešení díval na skutečný svět skrze speciální vybavení. Nejčastěji to může být hledáček fotoaparátu mobilního zařízení, vznikají ale také specializované headsety, jmenovitě známý Microsoft HoloLens.

Aby zobrazení virtuálních předmětů ve fyzickém světě fungovalo přesvědčivě, musí zařízení znát svoji polohu, mít senzory pohybu a znát polohu, kam vykreslit virtuální předmět. U mobilních aplikací se tak často využívá speciální obrázek v odstínech šedé, označovaný jako „marker“. Díky němu aplikace ví pozici a může zjistit také vzdálenost zařízení od této značky.



Obrázek 4 - Příklad markeru pro iOS aplikaci HoloWatch [21]

Rozšířená realita má před sebou ještě dlouhý vývoj, již nyní ale nabízí užitečná uplatnění. Například v mapových aplikacích pro zobrazení bodů zájmu, či v automobilech na čelních sklech pro zobrazení informací.

3.8.2 Rozšířená realita a Windows Phone

Protože je implementace rozšířené reality velmi náročný proces, bylo rozhodnuto o využití volně dostupné a propracované knihovny GART – Geo Augmented Reality Toolkit. Jedná se o nejvíce vyvinutou a podporovanou knihovnu pro Windows Phone a také

Windows 8, která primárně pracuje s geolokací a zobrazováním objektů na mapové komponentě.

Autoři knihovny GART poskytují komplexní dokumentaci, kde je možné najít všechny důležité informace potřebné pro implementaci ve vlastní aplikaci. Ke stažení je rovněž dvojice ukázkových projektů, které právě tuto knihovnu využívají.

3.8.3 Implementace GART v projektu

Po stažení balíčku je třeba přidat DLL knihovnu do projektu. Existuje verze pro ARM procesory (mobilní zařízení) a x86 (počítače, tablety a Windows Phone emulátor). Následně je třeba přidat knihovnu mezi jmenné prostory pro XAML.

```
....  
....xmlns:controls="clr-namespace:GART.Controls;assembly=GART.WP8"  
....
```

Obrázek 5 - Ukázka přidání jmenného prostoru XAML

Dále už je možné přejít k samotnému definování jednotlivých GART komponent na stránce aplikace. Tou hlavní je komponenta ARDisplay. Potřeba je nastavit dvojici vlastností a sice MovementThreshold, která udává počet metrů pro aktualizaci polohových služeb (v projektu byla zvolena hodnota 10) a AttitudeRefreshRate pro pohybové senzory telefonu (zde byla po krátké analýze zvolena hodnota 50).

Uvnitř XAML značky pro ARDisplay je třeba definovat ještě několik potomků pro kompletní funkcionalitu. Komponenta VideoPreview dodává snímky z kamery, které mohou být zobrazeny. OverheadMap slouží k zobrazení mapy města, jako mapové podklady používá GART knihovna interně ty od Bing. WorldView slouží jako vrstva pro zobrazování virtuálních předmětů na mapě a hledáčku kamery. Poslední je HeadingIndicator, ten uživateli ukazuje jeho směr při pohybu.

Jakmile jsou všechny výše zmíněné komponenty definovány, v XAML už není třeba dalších úprav, aby GART knihovna mohla fungovat.

3.8.4 Spuštění GART služeb

Windows Phone nabízí programátorovi možnost přetížení metod, které jsou volány při navigování na stránku či navigování z ní pryč. V projektu je tak využíváno přetížení metody OnNavigatedTo, která je zavolána vždy po navigování na danou stránku.

V této metodě dochází ke spuštění všech služeb potřebných pro fungování GART knihovny. Výše zmíněná komponenta ARDisplay obsahuje metodu StartServices, která se o vše postará. Rovněž při navigování na stránku dochází k naplnění kolekce ARItems komponenty ARDisplay. Jedná se o objekty, které jsou vykreslovány na mapě a v hledáčku fotoaparátu. Pro každý je nutné specifikovat geografické souřadnice.

O poskytnutí těchto objektů se stará view model MapPageVm. Pro všechny zastávky z třídy Stations vytvoří kolekci objektů typu ARItem. Jediným požadavkem na tento objekt je již zmíněná geolokace a vlastnost Content typu objekt. Jako Content aplikace nastavuje komponentu StackPanel obsahující dvojici komponent typu TextBlock. První zobrazuje název zastávky a druhá její přibližnou vzdálenost. Pro větší přehlednost je pozadí komponenty StackPanel šedé a bílý text je díky tomu více čitelný, než kdyby byl zasazen přímo do hledáčku fotoaparátu.

Pro vypnutí služeb GART knihovny byla přetížena metoda OnNavigatedFrom, která je volána vždy, když uživatel opustí danou stránku. Zde pouze stačí zavolat StopServices na komponentě ARDisplay.

3.8.5 Použití Motion API

Stránka se zobrazením mapy a rozšířené reality navíc využívá možností Motion API. Díky němu může programátor prakticky v reálném čase snímat informace o pohybu zařízení. Jak je nakloněno, jestli se hýbe a další informace.

Na této stránce je Motion API použito k automatickému skrytí komponenty mapy a zobrazení hledáčku fotoaparátu, pokud uživatel zvedne zařízení do vzpřímené polohy a je tak jasné, že nemá v plánu koukat na mapu.

Pohybové rozhraní se používá pomocí instance třídy Motion. Ta je vytvořena již ve zmíněném přetížení metody OnNavigatedTo. Kromě vytvoření instance je nastavena vlastnost TimeBetweenUpdates pro určení, jak často si programátor přeje dostávat informace o aktuální poloze zařízení. V projektu byla zvolena hodnota 200 milisekund. Druhou částí je přihlášení se k získávání dat při jejich změně skrze událost CurrentValueChanged.

3.8.6 Zpracování hodnot od Motion API

Uvnitř těla metody `_motion_CurrentValueChanged` určené na reagování na změnu dat je využita metoda `BeginInvoke` systémové třídy `Dispatcher`, která provede dané volání na hlavním (UI) vláknu aplikace.

Účelem sledování pohybu je změna viditelnosti komponenty `OverheadMap` skrze nastavení vlastnosti `Visibility`. Z tohoto důvodu je nutné využít služeb třídy `Dispatcher` pro provedení této změny týkající se uživatelského rozhraní.

Při každé události `CurrentValueChanged` je provedena metoda `MotionDataChanged` zavolaná skrze `Dispatcher`. Z předaných dat skrze instanci struktury `MotionReading` je získán náklon zařízení označovaný jako `Pitch` a převeden na stupně.

Pokud je hodnota ve stupních mezi 45 a 125 stupni, tak je podmínka splněna a dojde k nastavení vlastnosti `Visibility` na komponentě `OverheadMap` na `Collapsed`, čímž je schována a je vidět vstup z kamery zařízení. Výše zmíněné stupně byly zjištěny empiricky pomocí snímání hodnot `Pitch` a sledování náklonu testovacího zařízení. V momentě, kdy dojde k vrácení zařízení do původní polohy, již podmínka neplatí a komponenta `OverheadMap` s mapou je opět zobrazena.

3.9 Stránka pro plánování trasy

Poslední z hlavních stránek aplikace slouží pro plánování trasy. Základem uživatelského rozhraní je kontrolka na výběr zastávky. Jedná se o tlačítko, které otevírá nabídku s komponentou `LongListSelector` a po výběru je nabídka opět zavřena. Výběr času je realizován skrze kontrolku `TimePicker` z již zmíněné knihovny `Windows Phone Toolkit`. Uživateli nabízí snadné zvolení času a programátor nemusí řešit validaci dat zadaných uživatelem.

Protože algoritmus pro hledání trasy využívá ve velkém posloupnosti zastávek jednotlivých linek, je ihned po navigování na tuto stránku spuštěna metoda pro jejich předběžné načtení třídy `StationChainsCache`, která byla popsána výše. Uživatelské rozhraní je zablokované skrze nastavení vlastností `IsEnabled` jednotlivým kontrolkám do doby, než si aplikace na pozadí načte vše potřebné. Čekání je z pohledu uživatele velmi krátké.

Po spuštění hledání spojení od uživatele je zavolána metoda `FindConnectionsAsync`. Jejím úkolem je najít vyhovující spojení a ta následně zobrazit v komponentě `LongListSelector`, navíc ještě skryje prvky uživatelského rozhraní určené pro hledání spojení.

3.9.1 Algoritmus pro hledání spojení

Výše zmíněná metoda interně používá metodu `FindConnectionsInternalAsync` pro větší přehlednost. V této metodě je implementován algoritmus na hledání nejvýhodnějších spojení. Nejdříve si vyfiltruje všechny linky včetně směrů, které obsluhují vybranou cílovou zastávku a také startovní, kde se uživatel právě nachází.

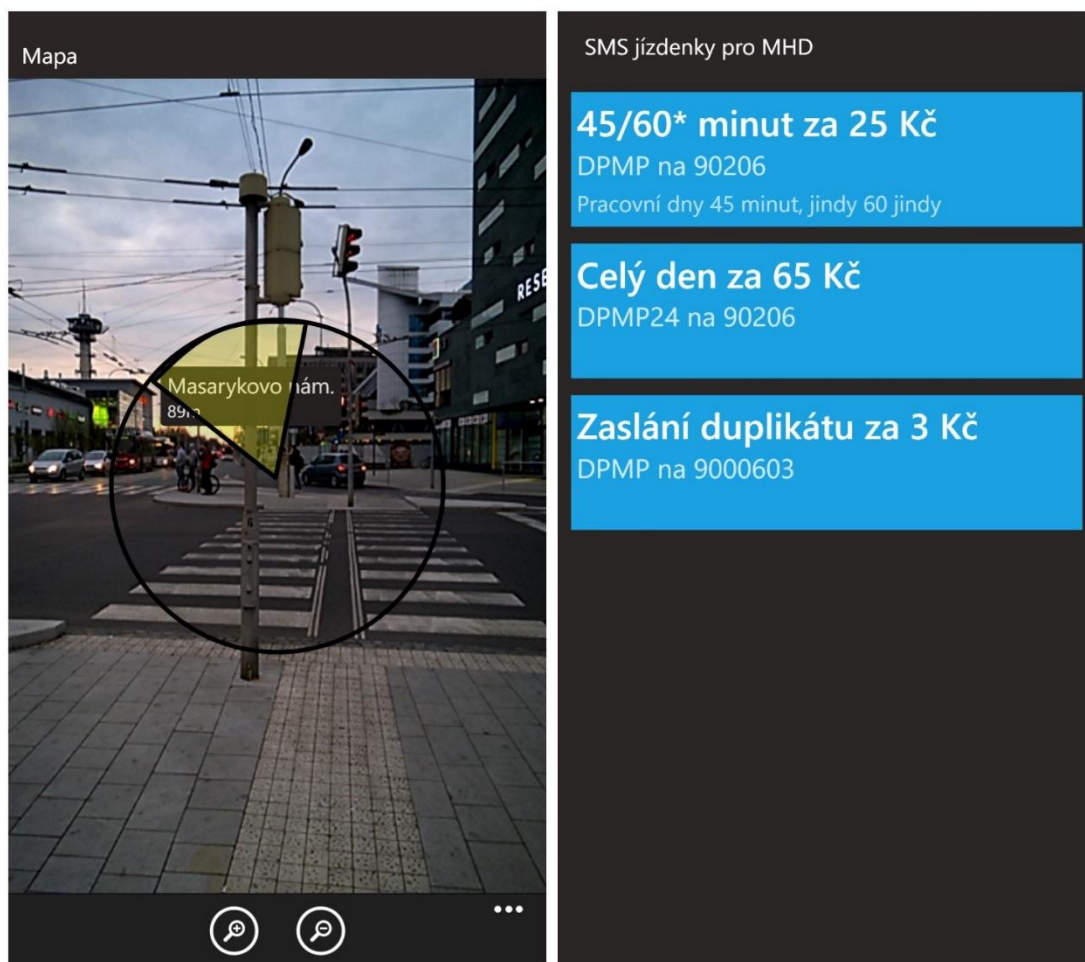
Následně jsou pro každý tento identifikátor (číslo linky a směr) získány z databáze spojení, která jsou navíc filtrována podle typu spojení (popsaného v předešlých kapitolách) a čísla linky. Spojení jsou seřazena sestupně podle času odjezdu.

Algoritmus následně cyklicky prochází získaná spojení. Jakmile narazí na spojení z cílové zastávky, které má nižší čas, než je požadovaný čas příjezdu uživatelem, spojení uloží do nové instance modelové třídy `ConnectionWithArrival` a označí nález pomocí booleovské proměnné `terminalStationFound`. Jakmile je tato proměnná přepnuta, algoritmus začne ve stejném cyklu hledat startovní zastávku. Ve chvíli, kdy ji najde (díky seřazení je to vždy ten nejbližší odjezd), přidá informace do dříve vytvořené instance třídy `ConnectionWithArrival` a začne hledat nová (dřívější) spojení.

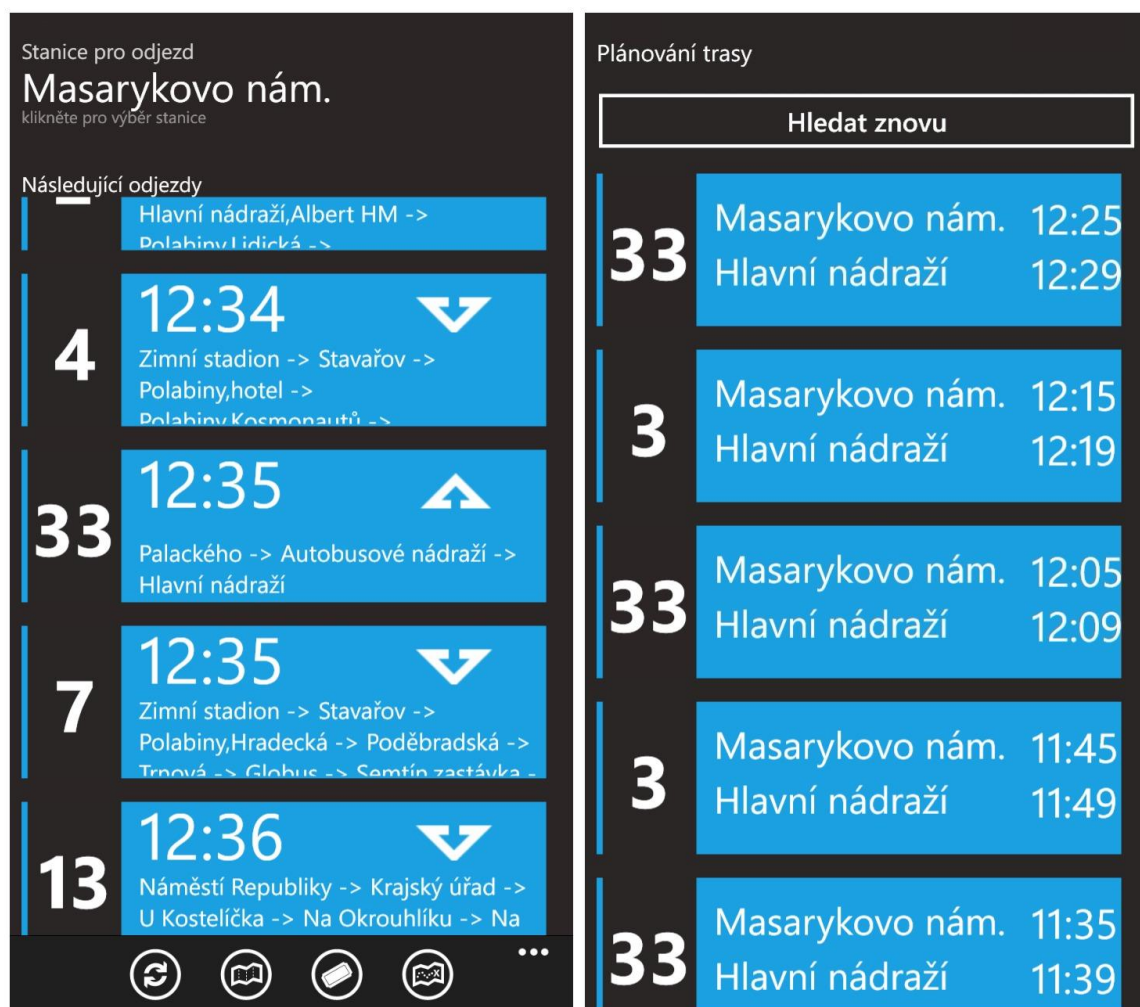
Na konci průběhu algoritmu jsou ještě nalezená spojení vyfiltrována na ta, která mají odjezd v rozmezí jedné hodiny od aktuálního času. Data binding se následně postará o zobrazení těchto spojení v uživatelském rozhraní uživateli.

Doba vyhledávání spojení je zhruba 2 sekundy při testování na nejfrekventovanějších zastávkách. V první verzi algoritmu byla více než 10 sekund, nakonec se ale čas podařilo optimalizovat na velmi příjemné maximální 2 sekundy.

4 UKÁZKA HOTOVÉ APLIKACE



Obrázek 6 - Ukázka rozšířené reality a nákupu SMS jízdenek



Obrázek 7 - Ukázka úvodní stránky se zobrazením spojů a plánování trasy

5 ZÁVĚR

V rámci bakalářské práce byla v praktické části provedena analýza požadavků na mobilní aplikaci. Následně byla získána podkladová data z několika zdrojů a vytvořena mobilní aplikace pro městskou hromadnou dopravu v Pardubicích pro operační systém Windows Phone 8.1. Výsledná aplikace umožňuje velmi snadno zjistit aktuální odjezdy spojů z nejbližší zastávky, která je uživateli nalezena automaticky skrze zjištění polohy zařízení. Jako další funkcionalitu aplikace demonstruje využití rozšířené reality pro snadnější využívání aplikace. Společně s mapou dokáže uživatele velmi snadno a netradičně navigovat na zastávky městské hromadné dopravy.

Podle definovaných cílů je rovněž umožněno plánování trasy. Uživateli stačí, aby si vybral cílovou zastávku a čas, kdy nejpozději chce být na dané cílové zastávce. Následně je uživateli prezentováno několik nejlépe vyhovujících spojů pro zadaná data. Navíc není potřeba připojení k internetu. Pro větší komfort lze přímo z aplikace velmi snadno objednat všechny dostupné SMS jízdenky pro cestování městskou dopravou v Pardubicích. Přístup k datům o zpoždění se nepodařilo získat a z tohoto důvodu nebylo možné tento cíl splnit.

Teoretická část práce popisuje možnosti geolokace zařízení a to v obecném smyslu a také přímo pro mobilní platformu Windows Phone. Dále byly popsány nejznámější bezdrátové sítě s detailnějším zaměřením na mobilní síť, Wi-Fi a Bluetooth.

Jako další rozšíření výsledné práce by mohlo být přidání dalších velkých měst. Úprava aplikace by v tomto směru pravděpodobně nevyžadovala velký zásah či rozšíření. Uživateli by dále mohlo být nabídnuto více možností při vyhledávání spojení, ukládání častých/oblíbených zastávek či zobrazení více informací v režimu rozšířené reality. Pro použití v Praze či Brně by se určitě hodila lokalizace do angličtiny.

6 POUŽITÁ LITERATURA

- [1] SKEET, Jon. C# in depth. Third editions. Shelter Island, NY: Manning, 2014, xxx, 582 pages. ISBN 161729134x.
- [2] WAGNER, Bill. Effective C#: 50 specific ways to improve your C#. Boston: Addison-Wesley, c2005, xviii, 307 p. ISBN 0321245660.
- [3] UNGER, Russ a Carolyn CHANDLER. A project guide to UX design: for user experience designers in the field or in the making. Berkeley: New Riders, c2009, xix, 267 s. Voices that matter. ISBN 9780321607379.
- [4] LIDWELL, William, Kritina HOLDEN a Jill BUTLER. Universal principles of design: 100 ways to enhance usability, influence perception, increase appeal, make better, design decisions, and teach through design. Gloucester: Rockport Publishers, c2003, 216 s. ISBN 1592530079.
- [5] Differences in Windows Phone 8.1 feature behavior for Windows Phone Silverlight 8.1 apps. Microsoft Developer Network [online]. 2016 [cit. 2016-04-09]. Dostupné z: [https://msdn.microsoft.com/en-us/library/windows/apps/dn655124\(v=vs.105\).aspx](https://msdn.microsoft.com/en-us/library/windows/apps/dn655124(v=vs.105).aspx)
- [6] INSIDE THE 3G MOBILE REVOLUTION. GSM History: History of GSM, Mobile Networks, Vintage Mobiles [online]. 2016 [cit. 2016-04-09]. Dostupné z: http://www.gsmhistory.com/inside_the_3g_revolution/
- [7] What is Bluetooth Technology? Bluetooth SIG [online]. 2016 [cit. 2016-04-09]. Dostupné z: <https://www.bluetooth.com/what-is-bluetooth-technology>
- [8] A short history of Bluetooth. NORDIC SEMICONDUCTOR [online]. 2014 [cit. 2016-04-09]. Dostupné z: <https://www.nordicsemi.com/eng/News/ULP-Wireless-Update/A-short-history-of-Bluetooth>
- [9] A brief history of Wi-Fi. The Economist [online]. 2004 [cit. 2016-04-09]. Dostupné z: <http://www.economist.com/node/2724397>
- [10] Wi-Fi - IEEE Standards. Tutorials Point [online]. 2016 [cit. 2016-04-09]. Dostupné z: http://www.tutorialspoint.com/wi-fi/wifi_ieee_standards.htm
- [11] Všechno, co byste měli vědět o Wi-Fi. Živě.cz [online]. 2012 [cit. 2016-04-09]. Dostupné z: <http://www.zive.cz/clanky/vsechno-co-byste-meli-vedet-o-wi-fi/omezena-pasma-a-standardy-anteny/sc-3-a-162796-ch-80485/default.aspx>
- [12] What is LTE Advanced? AndroidAuthority [online]. 2016 [cit. 2016-04-09]. Dostupné z: <http://www.androidauthority.com/lte-advanced-176714/>

- [13] EDGE, WiMAX, 3G, 4G: what's the difference? ICT Pulse Consulting Limited [online]. 2011 [cit. 2016-04-09]. Dostupné z: <http://www.ict-pulse.com/2011/07/edge-wimax-3g-4g-what%E2%80%99s-the-difference/>
- [14] Windows Phone 8.1 for Developers: Geolocation and Geofencing. Jayway [online]. 2014 [cit. 2016-04-10]. Dostupné z: <http://www.jayway.com/2014/04/22/windows-phone-8-1-for-developers-geolocation-and-geofencing/>
- [15] Geolocation 101: How It Works, the Apps, and Your Privacy. PCWorld [online]. 2010 [cit. 2016-04-10]. Dostupné z: <http://www.pcworld.com/article/192803/geolo.html>
- [16] Implementing augmented reality using GART. Igor ralic [online]. 2012 [cit. 2016-04-10]. Dostupné z: <http://igrali.com/2012/05/24/implementing-augmented-reality-using-gart/>
- [17] Augmented reality with GART: what's changed on Windows Phone 8. Igor ralic [online]. 2013 [cit. 2016-04-10]. Dostupné z: <http://igrali.com/2013/12/13/augmented-reality-gart-whats-changed-windows-phone-8/>
- [18] Evolution Of Mobile Technology: A Brief History of 1G, 2G, 3G and 4G Mobile Phones. Bright Hub [online]. 2011 [cit. 2016-04-10]. Dostupné z: <http://www.brighthub.com/mobile/emerging-platforms/articles/30965.aspx>
- [19] Vývěsné jízdní řády: Městská hromadná doprava Pardubice. IDOS: Jízdní řády [online]. 2016 [cit. 2016-04-11]. Dostupné z: <http://portal.idos.cz/IDS/Search.aspx?param=pacz>
- [20] Well-Known Users of SQLite. SQLite Homepage [online]. 2016 [cit. 2016-04-11]. Dostupné z: <https://www.sqlite.org/famous.html>
- [21] HoloWatch target. HoloWatch [online]. 2016 [cit. 2016-04-11]. Dostupné z: <https://holowatch.wordpress.com/about/>
- [22] What is ZigBee? Telegesis [online]. 2016 [cit. 2016-04-11]. Dostupné z: <http://www.telegesis.com/about-us/zigbee-overview/>
- [23] FREQUENTLY ASKED QUESTIONS. Z-WAVE [online]. 2016 [cit. 2016-04-11]. Dostupné z: <http://www.z-wave.com/faq>
- [24] What's The Difference Between ZigBee And Z-Wave? ElectronicDesign.com [online]. 2012 [cit. 2016-04-11]. Dostupné z: <http://electronicdesign.com/communications/what-s-difference-between-zigbee-and-z-wave>
- [25] SMS jízdenka, aplikace Sejf. Dopravní podnik města Pardubic a.s. [online]. 2016 [cit. 2016-04-11]. Dostupné z: <http://www.dpmp.cz/sms-jizdenka/>

- [26] Number of smartphone users worldwide from 2014 to 2019 (in millions). Statista [online]. 2016 [cit. 2016-04-17]. Dostupné z: <http://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/>
- [27] Jízdní řády IDOS. Google Play [online]. [cit. 2016-04-17]. Dostupné z: <https://play.google.com/store/apps/details?id=com.circlegate.tt.transit.android>
- [28] SQLite for Windows Phone 8.1. Visual Studio Gallery [online]. 2016 [cit. 2016-04-17]. Dostupné z: <https://visualstudiogallery.msdn.microsoft.com/5d97faf6-39e3-4048-a0bc-adde2af75d1b>