VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

BAKALÁŘSKÁ PRÁCE

Brno, 2017 Michal Bojtoš

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ
DEPARTMENT OF TELECOMMUNICATIONS

APLIKACE PRO URČENÍ POLOHY MOBILNÍCH ZAŘÍZENÍ
APPLICATION FOR LOCATION OF MOBILE DEVICES

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **Michal Bojtoš**
AUTHOR

VEDOUCÍ PRÁCE — **doc. Ing. Dan Komosný, Ph.D.**
SUPERVISOR

BRNO 2017

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní obor **Teleinformatika**
Ústav telekomunikací

Student: Michal Bojtoš ***ID:*** 173615
Ročník: 3 ***Akademický rok:*** 2016/17

NÁZEV TÉMATU:

Aplikace pro určení polohy mobilních zařízení

POKYNY PRO VYPRACOVÁNÍ:

Vytvořte aplikaci pro mobilní telefon, která bude zobrazovat tyto údaje: veřejnou IP adresu, privátní IP adresu a polohu ve formě souřadnic. Polohu telefonu získejte pomocí vestavěného GPS zařízení a zobrazte ji na mapě. Získané údaje odesílejte na FTP/SSH server a pomocí emailu. Proveďte srovnání s informacemi v geolokační databázi (například MaxMind nebo IP2Location) a zobrazte chybu v kilometrech. Aplikaci sestavte v programovacím jazyce Python pro OS Android.

DOPORUČENÁ LITERATURA:

[1] PUŽMANOVÁ, R. TCP/IP v kostce. 2. vyd. Kopp, 2009. 620 s. ISBN: 978-80-7232-388-3.

[2] PILGRIM, M. Ponořme se do Python(u) 3. CZ.NIC, 2010. 435 s. ISBN: 978-80-904248-2-1.

Termín zadání: 1.2.2017 ***Termín odevzdání:*** 8.6.2017

Vedoucí práce: doc. Ing. Dan Komosný, Ph.D.
Konzultant:

doc. Ing. Jiří Mišurec, CSc.
předseda oborové rady

UPOZORNĚNÍ:
Autor bakalářské práce nesmí při vytváření bakalářské práce porušit autorská práva třetích osob, zejména nesmí zasahovat nedovoleným způsobem do cizích autorských práv osobnostních a musí si být plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č.40/2009 Sb.

ABSTRAKT

Bakalářská práce se zabývá vývojem aplikace pro určení polohy mobilních zařízení. Hlavním cílem práce bylo vytvoření aplikace, kterou lze určit polohu ve tvaru GPS souřadnic zařízení. Vstupem zařízení je privátní a veřejná IP adresa. Toho bylo dosaženo prostřednictvím stanovení polohy telefonu integrovaným GPS přijímačem, spárováním a následným přenosem výstupních dat na emailovou schránku uživatele, FTP a SFTP server. Součástí práce je srovnání polohy přístroje se získanými daty z webových aplikací zaměřujících se na geolokační databáze. S těmito daty dále aplikace pracuje a poskytuje výsledky ve formě textového a grafického zobrazení. K vytvoření aplikace bylo využito programovacího jazyku Python pro OS Android.

KLÍČOVÁ SLOVA

Android OS, GPS, IP adresa, mobilní aplikace, Python

ABSTRACT

The bachelor thesis deals with the development of application for positioning of mobile devices. The main goal of the thesis was to create a specific application that can be used to determine positions in the form of GPS coordinates of the device. Device input is private and public IP address. This was achieved by identifying the location of the phone with an integrated GPS receiver, pairing and then transferring output data to the user's mailbox, FTP and SFTP server. Part of the thesis is to compare the location of the device with the data obtained from web applications focused on the geolocation databases. With these data, the application works and provides the results in the form of text and graphical views. Python for Android OS was used to create the application.

KEYWORDS

Android OS, GPS, IP address, mobile application, Python

BOJTOŠ, Michal *Aplikace pro určení polohy mobilních zařízení*: bakalářská práce. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2016. 59 s. Vedoucí práce byl doc. Ing. Dan Komosný, Ph.D.

Vysázeno pomocí balíčku `thesis` verze 2.61; `http://latex.feec.vutbr.cz`

PROHLÁŠENÍ

Prohlašuji, že svou bakalářskou práci na téma „Aplikace pro určení polohy mobilních zařízení“ jsem vypracoval(a) samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor(ka) uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této bakalářské práce jsem neporušil(a) autorská práva třetích osob, zejména jsem nezasáhl(a) nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom(a) následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno ....

podpis autora(-ky)

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu bakalářské práce panu doc. Ing. Danovi Komosnému, Ph.D. za odborné vedení, konzultace, trpělivost a podnětné návrhy k práci.

Brno ....

podpis autora(-ky)

Faculty of Electrical Engineering
and Communication
Brno University of Technology
Purkynova 118, CZ-61200 Brno
Czech Republic
http://www.six.feec.vutbr.cz

PODĚKOVÁNÍ

Výzkum popsaný v této bakalářské práci byl realizován v laboratořích podpořených z projektu SIX; registrační číslo CZ.1.05/2.1.00/03.0072, operační program Výzkum a vývoj pro inovace.

Brno . . . . . .

. . . . . .
podpis autora(-ky)

EVROPSKÁ UNIE
EVROPSKÝ FOND PRO REGIONÁLNÍ ROZVOJ
INVESTICE DO VAŠÍ BUDOUCNOSTI

OP Výzkum a vývoj
pro inovace

OBSAH

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

SEZNAM VÝPISŮ

ÚVOD

V každodenním životě může nastat situace, kdy uživatel potřebuje vyhledat určitou službu lokalizovanou do jeho nejbližšího okolí. U takového případu lze považovat za vhodný nástroj geolokační databázi, díky které použitý webový vyhledávač dokáže zobrazit nejadekvátnější hledaný výsledek na základě veřejné IP adresy zařízení. Tato problematika úzce souvisí s párováním GPS souřadnic s veřejnou IP adresou.

Geolokační databáze obecně nevykazují vysoký stupeň přesnosti při určení polohy, nicméně pro určení polohy nepotřebujeme GPS přijímač. Jednotlivé výstupy v těchto databázích mohou být tedy stanoveny na základě dat poskytnutých jinými zařízeními, které komunikují v rámci stejného rozsahu sítě. Upřesnění polohy lze dosáhnout tak, že každé zařízení, ať už s vědomím či nevědomím uživatele, bude průběžně získaná data odesílat do geolokační databáze. Tímto principem lze dosáhnout stavu, kdy konkrétní zařízení nevyužívající GPS technologie může získat údaj o své poloze z již dříve uložených lokalizačních údajů.

Tématem bakalářské práce s názvem „Aplikace pro určení polohy mobilních zařízení“, jejímž hlavním cílem je vypracování aplikace pro mobilní zařízení, která bude mít za úkol plnění geolokační databáze. Výstupními daty pro plnění jsou privátní a veřejná IP adresa s návazností na polohou přístroje ve formě GPS souřadnic.

K dosažení hlavního cíle slouží naplnění dílčích cílů, což v základu představuje stanovení polohy telefonu pomocí vestavěného GPS zařízení, použití funkce pro přenos dat s odesíláním na emailovou schránku uživatele, FTP, SFTP a server. Další kroky zahrnují srovnaní polohy přístroje se získanými daty z webových aplikací profilujících se právě na geolokační databáze a zpracování nepřesnosti určené polohy. Samotná aplikace je vytvořena v programovacím jazyce Python pro OS Android.

Ze struktury stanovených cílů vyplývá složení bakalářské práce. Teoretická část se zabývá geolokací mobilních zařízení, IP adresací a službami sloužící k přenosu dat. Praktická část obsahuje postup se zhotovením aplikace.

Hlavní motivací výběru a vypracování projektu byl můj zájem o vývoj mobilních aplikací a programovací jazyk Python. Velmi zajímavá byla pro mne možnost pracovat s technologii frameworku Kivy a jeho základní strukturou, která zabezpečuje funkčnost výsledné aplikace.

1 GEOLOKACE MOBILNÍHO ZAŘÍZENÍ

Tato kapitola přibližuje technologický vývoj GPS, problematiku geolokace mobilních zařízení a její aktuální možné využití v dnešní společnosti s příkladem možného využití.

1.1 Technologie GPS

Technologie GPS (*Global Positioning System*) vznikla ve Spojených státech amerických pod správou Ministerstva obrany. Jelikož je tento systém považován za vojenský systém, kde je důležitým faktorem státní bezpečnost, přesnost a obchodní potenciál, vzniklo několik dalších systémů pro určení polohy. Patří mezi ně například ruský GLONASS (*Globalnaja navigacionnaja sputnikovaja sistěma*), japonský QZSS (*Quasi-Zenith Satellite System*) nebo evropský GALILEO [1].

Zaměříme se na technologii GPS-NAVSTAR (dále jen GPS), která pochází ze zmíněných Spojených států.
Rozlišujeme dvě kategorie:

- standardní služba navigace SPS (*Standard Positioning Service*),
- přesná služba navigace PPS (*Precision Positioning Service*).

Hlavním důvodem rozdělení technologie GPS do dvou kategorií je bezpečnost.

Standardní služba navigace SPS (dále jen SPS) je přístupná civilnímu obyvatelstvu s potřebným zařízením, což je v tomto případě GPS přijímač. Přesnost pro určení polohy se pohybuje v horizontální ose 3,8 m a ve vertikální 6,2 m.

Přesná služba navigace PPS (dále jen PPS) zastupuje vojenský navigační systém. Přístup k tomuto systému mají jen autorizovaní uživatelé s licencí od vlády Spojených států amerických. Licenci představuje jakýsi klíč, který slouží k dešifrování PPS signálu. Vzhledem k tomu, že se jedná o důležitý vojenský navigační systém, klade se zde důraz na přesnost. Přesnost určení polohy je v horizontální ose 1,2 m a ve vertikální ose 1,9 m [2].

V dnešní době již není problém pořídit mobilní zařízení s GPS přijímačem. GPS přijímač nám udává polohu za pomocí nominálních 24 družic ve vesmíru, kterou vyjadřujeme jako zeměpisnou šířku (*latitude*) a zeměpisnou délku (*longitude*) [3].

Princip GPS

GPS družice vysílá signál, který na základě zabudovaných atomových hodin udává přesný čas. Každý přenos trvá 30 sekund a přenese se během něj 1500 bitů šifrovaných dat.

Měření vzdálenosti GPS přijímače vzhledem k družici je založeno na časových signálech. GPS přijímač zachytí signál od různých družic s jistým prodlením, protože mikrovlnám trvá zlomek vteřiny, než rychlostí světla dospějí od družice k přijímači. Časový rozdíl převedeme do vzdálenosti mezi GPS přijímačem a družicí. Výslednou polohu následně zjistíme pomocí rozdílů mezi signály vysílanými družicemi [4].

Při získávání GPS souřadnic vznikají časové prodlevy v řádech několika sekund až minut. Tato prodleva vzniká při počáteční komunikaci GPS přijímače s družicí. Po zahájení vysílání je nutné, aby si GPS přijímač načetl (inicializoval) informace o jednotlivých družicích (tzv. almanach). Pokud bychom chtěli inicializovat celý almanach, bylo by to velice časově náročné, proto načítáme pouze část, která je dostačující a časově nenáročná. Pro zajištění geolokace pomocí GPS je potřeba navázat spojení alespoň se čtyřmi družicemi [5].

Časové prodlevy nevznikají jen inicializací. Důležitou roli zde hrají také faktory, jako jsou vysoké budovy, úzké uličky, předchozí použití GPS přijímače, podnebí a s tím související oblačnost.

1.2 GPS v mobilních zařízeních

Každé „chytré“ mobilní zařízení v sobě může obsahovat GPS přijímač, kterým dokáže lokalizovat aktuální polohu a interpretovat ji uživateli přístroje.

Vývojáři operačních systému a aplikací pro mobilní zařízení s touto vlastností neustále pracují. Snaží se využít možnosti získat GPS data z mobilních zařízení, které následně párují s použitým rozsahem IP adresy a ukládají do rozsáhlých databází. Díky rozsahu získané IP adresy jsou poté aplikace schopny přibližně lokalizovat další zařízení, využívající stejnou či blízkou síť, bez použití GPS souřadnic.

Nejznámějšími a nejrozsáhlejšími veřejnými geolokačními databázemi jsou v současnosti MaxMind a IP2Location.

Důsledkem vývoje těchto databází je například úprava obsahu ve vyhledávačích, cílená webová reklama a další marketingové služby. Využití nalezneme i v bezpečnosti webových aplikací při autentizaci a samozřejmě nelze opomenou využití technologie A-GPS, která nám pomáhá při rychlejší fixaci polohy GPS přijímače. Příklad popisující práci geolokační databáze viz obr. 1.1.

Na obrázků se odehrává příkladově vymyšlený scénář možné práce geolokační databáze, v našem případě k vyšetřování zločinu.
Klíčové linie scénáře:

1. cestující se občerstvuje v restauraci,
2. útočník napadá síť elektronických peněženek,
3. vyšetřovatelé hledají útočníka.

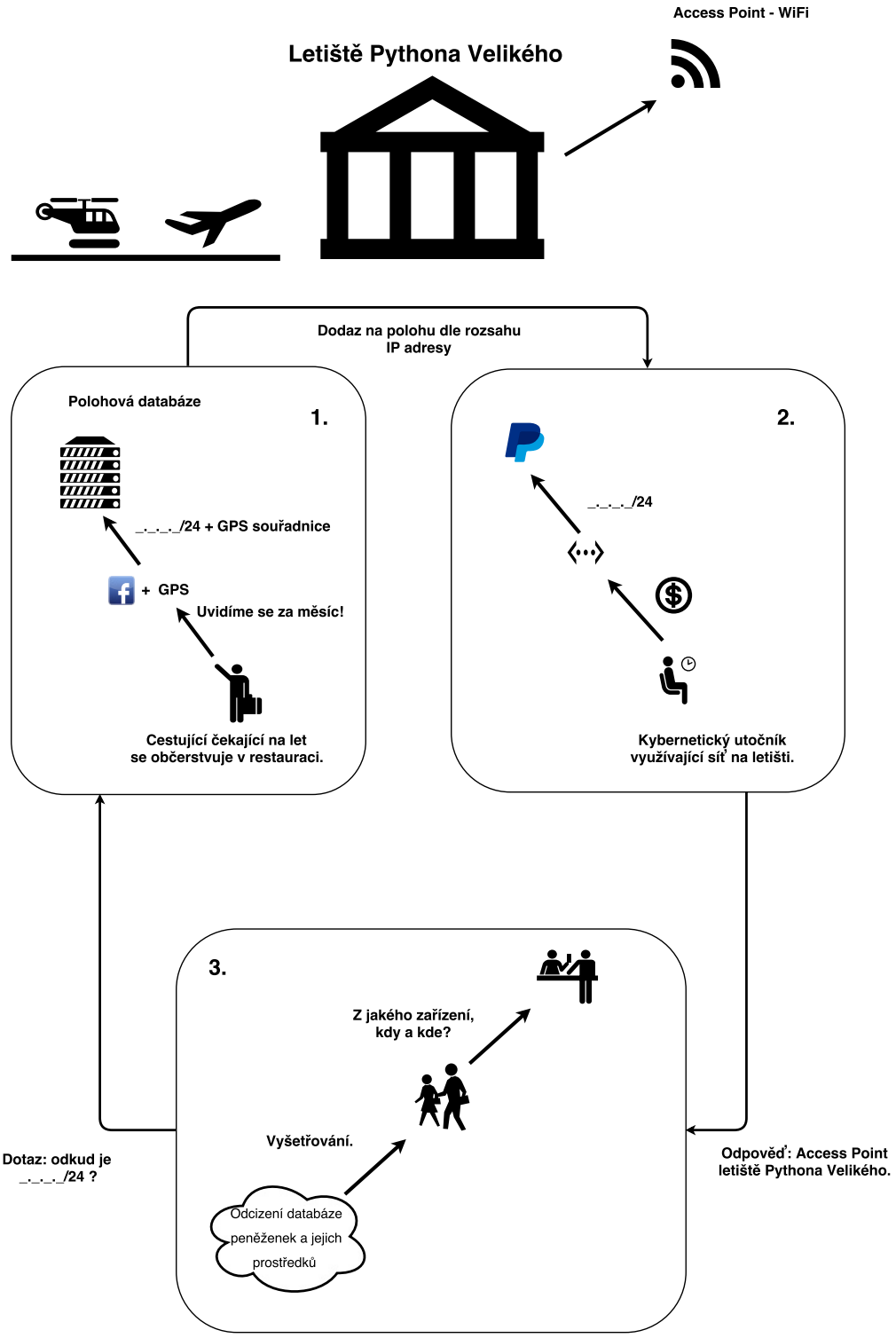
První linie scénáře se odehrává v restauraci letištní haly, kde se náš cestující občerstvuje. Mobilní zařízení je připojeno pomocí AP (*Access Point*) letiště.

Cestující si krátí čekání na let brouzdáním na internetu. Je velmi společenský při používání sociálních sítí. Vytvoří tedy svůj osobní status na svém profilu, aby okruhu přátel sdělil odlet. Díky abnormální kapacitě baterie se nachází GPS přijímač stále v pohotovosti bez vědomí našeho cestujícího. Při internetové komunikaci dochází ke spárovaní použité veřejné IP adresy letiště, pomocí které navázal spojení. Výsledné údaje se uloží do geolokační databáze, kde dle získaných IP adres stejného rozsahu dochází ke zpřesnění výsledné polohy.

O pár desítek metrů dál, v letištní hale, započala druhá linie scénáře. Sedí zde neznámý kybernetický útočník. Mobilní zařízení komunikuje pomocí AP letiště. Útočník úspěšně provedl útok a odchází s několika miliony dolarů domů.

Nyní se dostáváme ke konci tohoto scénáře. Uživatelé elektronických peněženek zjistili, že jim byly odcizeny jejich finanční prostředky. Vyšetřovatelé nejdříve nahlédli do přístupových záznamů serverů. Kybernetický útočník zde byl precizní při odstraňování přístupových informací, nicméně netušil, že zde je připojen server, který zabezpečuje pouze monitoring provozu.

Dle tohoto serveru byla zjištěna veřejná IP adresa, pomocí které se dostal útočník na jednotlivé servery. Z neznámých důvodů nevyužil přesměrování síťového provozu v rámci více sítí. Vyšetřovatelé za použití geolokační databáze jednoznačně určí místo útoku. Následně stačí získat kamerové záznamy letištní haly, popřípadě platby karetních terminálů v době útoku. Na letišti Pythona Velikého totiž lze platit pouze elektronickými terminály pro platební karty.



Obr. 1.1: Příklad využití párovaní GPS souřadnic s rozsahem IP adres

2 IP ADRESACE

Tato kapitola řeší určováním a přidělováním adresního prostoru IP a s tím související technologií NAT. Dále rozeberu jednotlivé organizace a jejich fungování v rámci přidělování a správy adresního prostoru.

2.1 Dělení adresního prostoru

IP adresa označuje jednoznačně síťové rozhraní zařízení. Pokud má zařízení více síťových rozhraní, je každému přidělena zvláštní IP adresa. Čím více zařízení se v síti nachází, tím více dochází k úbytku adresního prostoru.

Pokud chceme zajistit bezchybné navázání spojení v síti založené na modelu TCP/IP, je důležité, aby měl každý uzel unikátní IP adresu [6].

Za účelem navázání komunikace mezi zařízeními jsou tyto organizace zodpovědny za jednoznačné přidělování IP adresy, jména domén a udržování protokolů zabezpečujících správný průběh komunikace.

ISOC

V době největšího rozmachu (90. léta) vznikla řada neziskových organizací, které se zabývaly standardizací, výzkumem a v neposlední řadě vzděláváním. Organizace ISOC (*Internet Sociaty*) zajišťuje podporu a propagaci neziskových organizací, které pracují na internetových standardech [7].
Neziskové organizace jsou:

- IETF (*Internet Engineering Force*),
- IESG (*Engineering Steering Group*),
- IAB (*Internet Architecture Board*),
- IRTF (*Internet Research Task Force*).

Výše popsaná organizace IRTF stojí za tvorbou internetových standardů RFC, které zabezpečují pravidla komunikace mezi sítěmi. Z hlediska adresního prostoru hraje největší roli organizace jménem ICANN (*Internet Corporate for Assigned Names and Numbers*).
Skládá se ze tří účelových organizací:

- GNSO (*Generic Names Supporting Organisation*),
- ASO (*The Address Supporting Organisation*),
- ccNSO (*The Country Code Names Supporting Organisation*).

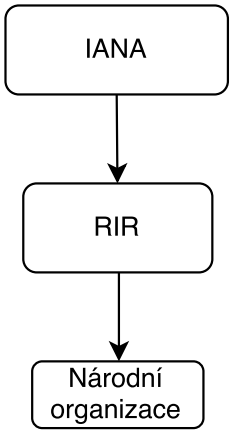
GNSO je obdoba pro generické domény (domény nejvyššího řádu, do této kategorie patří například doména `.org`).

ASO se stará o oblast IP adres a spadá zde organizace RIR (*Regional Internet Registeries*), která zajišťuje alokaci adresního prostoru v rámci své geografické působnosti.

Organizace ccNSO zajišťuje správný chod národních domén.

IANA

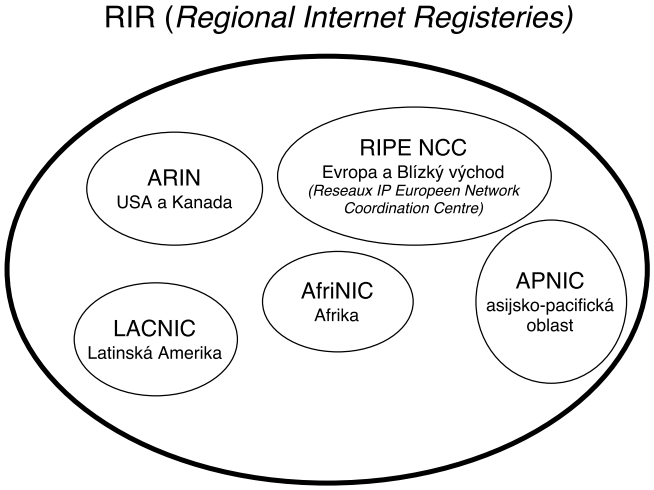
V dnešní době je IANA (*Internet Assigned Numbers Authority*) zastřešována organizací ICANN. Této organizaci je kladena velká váha, jelikož má na starosti globální alokaci IP adres(viz obr. 2.1), správu číselníku rodiny protokolů TCP/IP a také zajišťuje bezproblémový provoz kořenových jmenných serverů Internet [7].



Obr. 2.1: Struktura přerozdělování adres

RIR

RIR (*Regional Internet Registeries*) se skládá z pěti celků (viz obr. 2.2), které alokují adresní prostor v rámci své geografické působnosti. Tyto celky jsou provozovateli jmenných serverů pro reverzní domény (název domény přeložen do tvaru IP adresy) sítí třídy C, tzn. že jsou zodpovědný za chod reverzních domén.



Obr. 2.2: Rozdělení na jednotlivé RIR organizace

Národní organizace

Jednotlivé země rozděleny dle RIR mají přidělenou doménu či více domén. Správu těchto domén zajišťuje vždy konkrétní národní organizace – tzv. národní registrátor. V České republice tuto správu zajišťuje zájmové sdružení CZ.NIC [7].
Národní registrátoři jsou sdružováni v podobě mezinárodní organizace:

- LAC TLD (*Organización de ccTLDs Latinoamericanos y del Caribe*),
- CENTR (*Council of European National Top-Level Domain Registries*),
- APTLD (*Asia Pacific Top Level Domain Association*).

Národní organizace rozdělují domény druhé úrovně. Naopak mezinárodní sdružení nerozdělují ani IP adresy, ani jména domén. Jsou to jen zájmové sdružení [7].

CZ.NIC

Zájmové sdružení bylo v roce 1998 založeno předními poskytovateli internetu. Působí na českém území a jeho hlavní činností je správa databáze zaregistrovaných domén pro Českou republiku. Informace z databáze (například kdo je vlastník domény) jsou veřejně přístupné na webových stránkách tohoto sdružení.

2.1.1 Veřejné adresy

V období od položení základů internetu až po rok 1993 se utvářela norma RFC-796. Dle této normy se IP adresa skládá ze čtyř bajtů. Adresní prostor se dělí do tříd viz tab. 2.1, kde z IP adresy je jasné, co je adresa sítě a co je adresa zařízení umístěného v této síti [7].

Tab. 2.1: Rozdělení IP adres dle jednotlivých tříd

Třída	**1. oktet**	**2. oktet**	**3. oktet**	**4. oktet**
A	1–127	adresa zařízení		
B	128–191		adresa zařízení	
C	192–223			adresa zařízení
D	224–239			
E	>239			

K tomu, abychom správně určili adresu sítě z IP adresy, nám slouží síťová maska. Její složení je stejné jako u IP adresy, tedy čtyři bajty. Bity označeny číslem 1 patří adrese sítě, ostatní bity mají číslo 0. Pro jednotlivé výše uvedené třídy je každá maska zcela odlišná [7].

Po roce 1993 došlo k vydání norem RFC-1517 až 1520 pod známějším názvem CIDR (*Classless Inter-Domain Routing*). Díky těmto normám se již adresní prostor nerozděloval dle tříd, ale určoval se dle síťové masky [7].

Síťová maska

Síťová maska měla stále velikost čtyři bajty, avšak IP adresy již nebyly rozděleny do tříd. Maska byla libovolná. Zde je nutno říci, že v tomto případě je důležité vždy uvádět IP adresu i se síťovou maskou. Masku již nelze odvozovat dle třídy, jako tomu bylo u normy RFC-796.

Libovolná síťová maska umožňuje z adresního prostoru sítě vytvářet další subsítě, což je efektivní zejména k ušetření adresního prostoru pro případné vytváření dalších sítí.

Aby zápis síťových masek byl jednodušší a srozumitelnější, zavedl se pojem prefix. Prefix určuje počet jedniček v binárním zápisu síťové masky a píše se za lomítkem IP adresy [7].
Příklad zápisu: 192.168.1.13/28.

Pokud bychom znali adresu zařízení a nevěděli s jakou maskou byla použita, nelze adresu sítě zjistit.

2.1.2 Privátní adresy

Pro ušetření adresního prostoru slouží privátní IP adresy. Před zavedením tohoto řešení měly všechny uzly veřejnou IP adresu (viditelná všemi uzly), zatímco po zavedení se vytvořily tzv. privátní IP adresy (viditelné jen omezenému okruhu uzlů), které jsou schopny navazovat spojení mimo svou privátní síť. K tomu nám pomáhá technologie NAT, která zajišťuje překlad adres při vnější komunikaci [7].

Pokud rozdělujeme IP adresy ve vnitřní síti, je důležité řídit se normou RFC-1918, kde se IP adresy rozdělují do tzv. tříd viz tab. 2.2.

Tab. 2.2: Rozdělení IP adres dle jednotlivých tříd v rámci privátní sítě

Třída	**IP adresa/prefix**	**rozsah IP adres**
A	10.0.0.0/8	10.0.0.0 až 10.255.255.255
B	172.16.0.0/12	172.16.0.0 až 172.31.255.255
C	192.168.0.0/16	192.168.0.0 až 192.168.255.255

2.2 Překlad adres pomocí NAT

Technologie NAT (*Network Address Translator*) je technologie, která nám umožňuje navázat komunikaci mezi privátní a veřejnou sítí. Tímto procesem lze docílit zásadního ušetření IP adresního prostoru. Funguje na bázi přepisování a překladu IP adresy odesílatele a příjemce. Pro přepis IP adres stanice je nutné nastavení NAT na směrovači privátní sítě. Dále tuto technologii popisují normy RFC-2663, 2709, 2766, 2993 [7].

2.2.1 Jednoduchý NAT

Jednoduchý NAT slouží k navázání spojení privátní sítě s veřejnou sítí Internet. Pokud máme ve své privátní síti námi přidělenou IP adresu a dojde ke spojení, směrovač při spojení přepíše privátní IP adresu na jednu z IP adres, kterou má k dispozici od poskytovatele Internetu. Stačí, aby se ve směrovači nacházela převodní tabulka, dle které směrovač zjistí, na jakou IP adresu má přeložit adresu privátní sítě [7].

2.2.2 Rozšířený NAT

Tato varianta pracuje na podobném principu jako jednoduchý NAT. Je definována jako NAPT (*Network Address and Port Translation*). Oproti jednoduchému NAT zde dochází k výraznějšímu ušetření adresního prostoru. Rozšířený NAT totiž dokáže využívat čísla portu a typ protokolu. To znamená, že klientům připojeným ke stejné privátní síti, směrovač přepíše navzájem stejnou IP adresu jen s odlišným portem [7].

3 SLUŽBY SLOUŽÍCÍ K PŘENOSU DAT

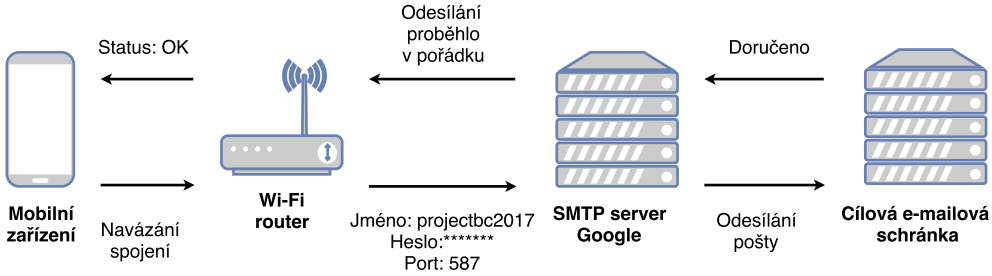
Tato kapitola se zabývá službami sloužící pro přenos dat v síťové komunikaci. Pro představu komunikace bude umístěn grafický průběh přenosu u každého z jednotlivých služeb přenosů dat.

3.1 Služba pro emailovou komunikaci

E-mailová komunikace se v dnešní době řadí k nejpoužívanějšímu oficiálnímu komunikačnímu prostředku mezi uživateli Internetu. Vytlačuje historicky papírovou formu dopisu, kde hlavní výhodou e-mailové komunikace je rychlost, příloha a samozřejmě minimální nákladnost [8].

Protokol SMTP (*Simple Mail Transfer Protocol*) je zastoupen dokumentem RFC 5321. S přihlédnutím k modelu ISO/ISO pracuje na aplikační vrstvě. Pro komunikaci využívá transportního protokolu TCP (*Transmission Control Protocol*), konkrétně port 25 [8].

SMTP je znakově orientovaný, což přináší značné výhody pro následné ladění protokolu. Přenos výstupních dat pomocí služby SMTP můžeme vyjádřit viz obr. 3.1.



Obr. 3.1: Průběh komunikace pomocí služby SMTP

3.2 Služba pro nešifrovaný přenos dat

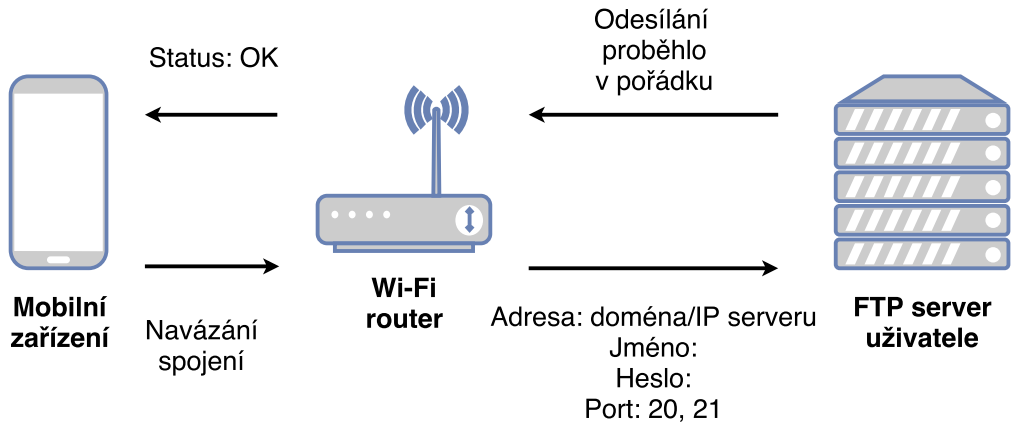
Stanice využívající službu FTP (*File Transfer Protocol*) mezi sebou vystupují jako klient – server. Při datovém přenosu je vždy nutné, aby odeslaná data klientem doputovala v pořádku na stranu serveru. Tento proces zabezpečuje transportní protokol TCP, port 20 pro přenos dat a port 21 pro přenos příkazů s následující reakcí. Tyto porty se řadí k 1. skupině portů (*privilegovaná*), tudíž pro komunikaci jsou nutná práva superuživatele [9].

Přenos dat pomocí služby FTP je nezabezpečen. Veškerá přenesená data, včetně přihlašovacích údajů do serveru, jsou čitelná bez zabezpečení šifrováním. Další komplikace nastává při přenášení dat. V průběhu činnosti je služba FTP zaneprázdněna a na přenos příkazů nereaguje [10].

Příkazy FTP se skládají z klíčového slova a parametru.
Příkazy FTP:

- služební (*service commands*),
  MKD – vytvoření nového adresáře,
  PWD – zjištění aktuálního pracovního adresáře,
  RMD – smazání adresáře,
- parametry přenosu (*access control commands*),
  USER – uživatelské jméno,
  PASS – heslo,
- řízení přístupu (*transfer parameter commands*),
  TYPE - typ přenášených dat,
  PASV - změna na pasivní mód.

Přenos výstupních dat pomocí služby FTP můžeme vyjádřit viz obr. 3.2.



Obr. 3.2: Průběh komunikace pomocí nešifrované služby FTP

3.3 Služba pro šifrovaný přenos dat

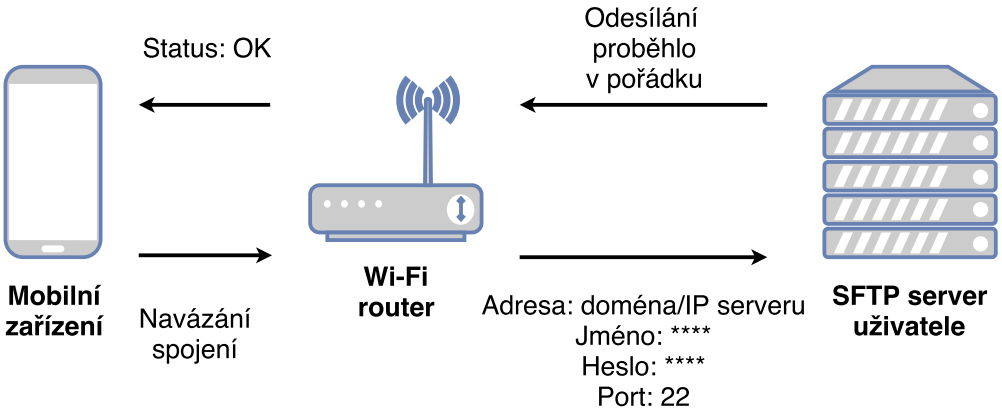
Jedním ze základních pilířů komunikace mezi koncovými body je bezpečnost. Tuto problematiku řeší služba SFTP (*SSH File Transfer Protocol*), který svou podstatou vyjadřuje alternativu k již zmíněné službě FTP.

Komunikace ve srovnání s FTP využívá pouze jeden přenosový kanál, kterým jsou přenášeny příkazy i samostatná data. Výchozí port služby SFTP je 22 [9].

Přenos dat pomocí SFTP využívá šifrovaného přenosu s použitím komunikačního protokolu SSH (*Secure Shell*). Šifrováním je zajištěn přenos dat v čitelné formě pouze oprávněným uživatelům.

Aby šifrování bylo umožněno, je potřeba veřejného a privátního klíče. Princip klíčování spočívá v tom, že data mohou být šifrována veřejným klíčem, dešifrována však nikoliv. Tudíž pokud chceme soubor dešifrovat, potřebujeme spárovaný privátní klíč [10]. Přenos výstupních dat pomocí služby SFTP znázorňuje obr. 3.3.
Průběh šifrování pomocí SSH:

1. autentizace – správnost identity (autentizace klient-server),
2. integrita – data jsou přenesena bez zásahu třetí osoby (neměnná),
3. šifrování – přenesená data čitelná pouze oprávněným uživatelem.



Obr. 3.3: Průběh komunikace pomocí šifrované služby SFTP

4 ZHOTOVENÍ APLIKACE

Tato kapitola popisuje vytvoření nástroje ve formě aplikace pro systém Android, díky kterému lze párovat veřejnou IP adresu se souřadnicemi GPS. Výstupní data se následně ukládají do geolokační databáze, kde se s nimi dále pracuje. Práce na projektu byla rozdělena do několika etap.
Etapy tvorby aplikace:

- první etapa se zaměřovala na získání veřejné IP adresy. Využil jsem webovou aplikaci `http://httpbin.org/ip`, která mi zajistila výpis veřejné IP adresy. Pomocí modulu `urlib` jsem výpis přenesl do mé aplikace. K získání privátní IP adresy přidělené směrovačem jsem použil modul `simplejson`, díky kterému jsem mohl využít syntaxe jazyka Java. Tato skutečnost mi umožnila navázat funkci pro získání privátní IP adresy na rozhraní API (*Application Programming Interface*) operačního systému Android. Následně již bylo pouze potřeba dostat IP adresu do oktetového tvaru.
- Ve druhé etapě bylo nutné zajistit, aby aplikace v pořádku komunikovala s GPS přijímačem mobilního zařízení. Pro tuto funkci bylo efektivní použít modul `plyer`. GPS souřadnice se stále obnovují v závislosti na určování polohy zařízení, kde se právě nacházíme.
- Cílem třetí etapy bylo navázání spojení s webovými aplikacemi, které již mají přístup ke geolokačním databázím. Pomocí modulu `simplejson` jsem pro vzájemné porovnání databází vypsal jejich data v textovém výstupu aplikace. Dále jsem zde využil modulu `MapView` pro interpretaci získaných a aktuálních souřadnic v mapě.
- V poslední etapě jsem vyřešil kompletní automatizaci a výsledný účel aplikace. Pomocí vhodně použitých modulů `smtplib`, `ftplib` a `paramiko` jsem zajistil přenos výstupních dat pomocí protokolů služeb SMTP, FTP a SSH.

Z hlediska grafického provedení jsem vytvořil grafické rozhraní aplikace pomocí frameworku Kivy a modulu `ScreenManager`. Tato část byla pro mne nejnáročnější. Bylo nutné zajistit textové výpisy, které budou proměnné v rámci změny výstupních dat, a uspořádat jednotlivé grafické prvky dle uživatelských požadavků a trendů GUI (*Graphical User Interface*).

4.1 Struktura aplikace

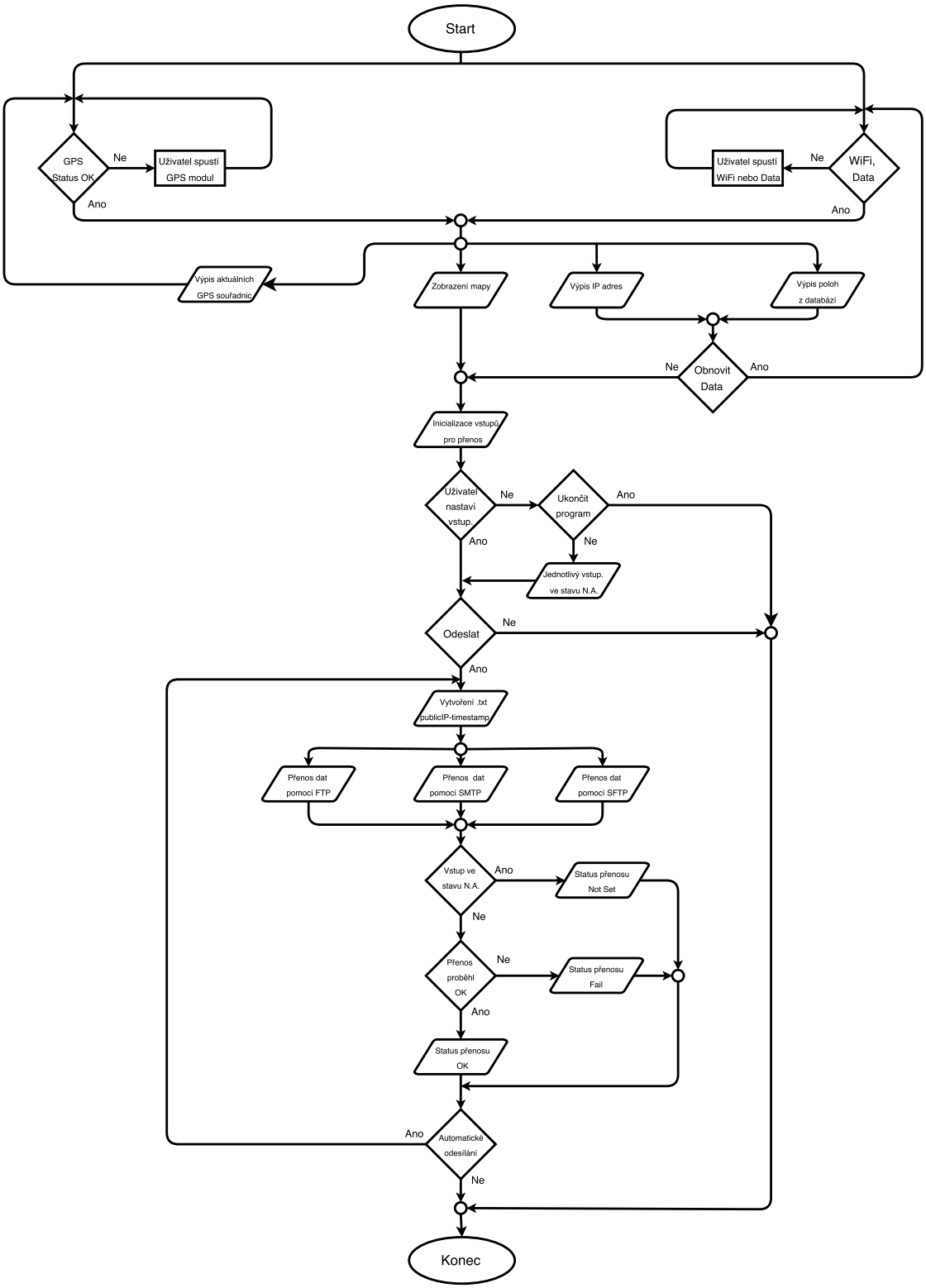
Struktura programu se skládá ze sedmi hlavních částí:

1. vytvoření funkcí pro GPS lokaci, IP adresaci v privátní a veřejné sítí s následným uložením do souboru,
2. funkce pro přenos dat pomocí protokolů SMTP, SSH a FTP,
3. funkce pro automatické odesílání dat při změně veřejné IP adresy,
4. textové zobrazení dat z geolokačních databázi,
5. grafické zobrazení ve formě mapy pro srovnání odchylky používaných databázi,
6. funkce `run` zajišťující spuštění aplikace,
7. vytvoření grafického rozhrání pro zobrazení výsledků.

Kompletní funkčnost aplikace zajišťuje framework Kivy. Bylo zde nutné implementovat dodatečné moduly viz příloha A.1, jako je například python-for-android, který zajišťuje snadnou kompilaci do výsledného instalačního souboru s koncovkou `.apk` a další.

Ke kompilaci byl použit program `Buildozer`, který pracoval na OS (Operační systém) Fedora verze 26. Kompilátor využíval již zmíněný modul `python-for-android`, který zajišťoval plynulou kompilaci.

Vývoj a následné testování probíhalo na mobilní platformě OS Android verze 5.1.1 s nástavbou CyanogenMod.
Základní procesy aplikace tvořící dílčí části vývojového diagramu viz 4.1.

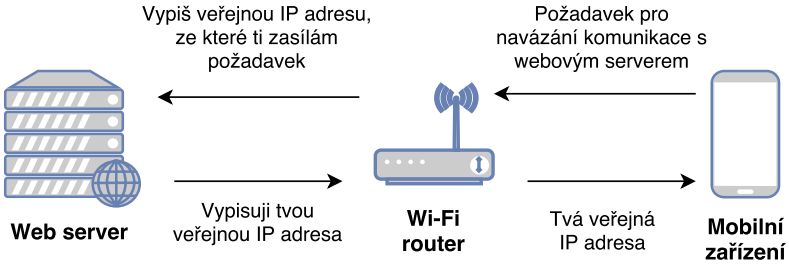


Obr. 4.1: Vývojový diagram aplikace

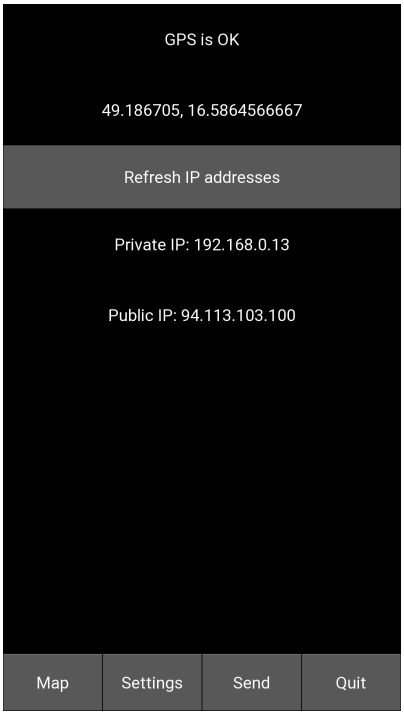
4.2 Získání veřejné IP adresy

Veřejnou IP adresu získáme pomocí aplikace jen tehdy, pokud je zařízení přidělena IP adresa poskytovatelem. V opačném případě aplikace tento stav ošetří výjimkou a upozorní na nutnost připojení datového či bezdrátového připojení. Pro výpis veřejné adresy využívám webovou aplikaci `http://httpbin.org/ip`. Komunikaci mezi mobilní a webovou aplikací můžeme vyjádřit viz obr. 4.2.

Za využití modulu `urlib` lze převést výpis veřejné IP adresy do vytvářené aplikace viz obr. 4.3. Pokud se nacházíme v privátní síti, je zde nutné mít správně nastaven směrovač.



Obr. 4.2: Využití modulu `urlib` pro získání veřejné IP adresy



Obr. 4.3: Ukázka výpisu veřejné IP adresy

Funkce pro získání veřejné IP adresy je definována ve třídě `TestApp(App)` viz výpis funkce 4.1.

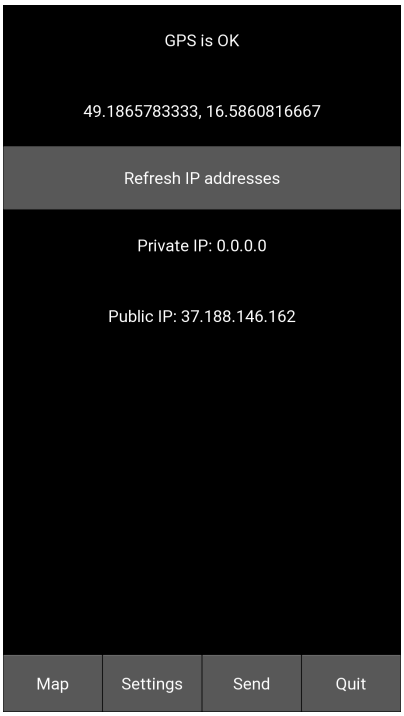
Výpis 4.1: Funkce pro získání veřejné IP adresy

```
def setGlobalIP(self):
      try:
          def getPublicIP():
              solutionGlobal = load(urlopen('http://httpbin.org/ip',timeout=6))['origin']
              solution = '%s' % (solutionGlobal)
              return solution

          self.labelTextGlobal = getPublicIP()

      except Exception:
          import traceback
          traceback.print_exc()
          self.labelTextGlobal = 'N.A.'
```

4.3 Získání privátní IP adresy

Pro získání privátní IP adresy bylo nutné zajistit komunikaci s API OS Android.

Dodatečný modul `simplejson` frameworku Kivy umožňuje pomocí tříd `PythonActivity`, `SystemProperties` a `Context` přistoupit k systémovým informacím zařízení. V mém případě není privátní IP adresa přidělena z důvodu datového připojení viz obr. 4.4.



Obr. 4.4: Ukázka výpisu privátní IP adresy

IP adresa je systémově zapsána jako jednotné číslo v desítkové soustavě. Zde je nutné využít funkce, která toto číslo převede na oktetový tvar viz výpis funkce.

Postupné získání privátní adresy je stejně jako získání veřejné IP adresy definováno ve třídě `TestApp(App)` viz výpis funkce 4.2.

Ve funkci pro získání privátní IP adresy mohou nastat dva chybné stavy:

1. zařízení je přidělena privátní IP adresa,
2. nelze alokovat IP adresu.

Dojde-li ke stavu „Zařízení je přidělena privátní IP adresa“, znamená to, že v tomto případě byla privátní IP adresa přidělena směrovačem či vnitřním systémem zařízení.

Pokud dojde ke stavu „Nelze alokovat IP adresu“, je zde zřejmé, že privátní IP adresa nebyla přidělena, tudíž se nenacházíme v privátní síti a komunikujeme „napřímo“. Oba tyto stavy jsou ošetřeny viz výpis 4.2.

Výpis 4.2: Funkce pro získání privátní IP adresy

```
def setPrivateIP(self):
      try:
          def int_to_ip(ipnum):
              octet1 = int(ipnum / 16777216) % 256
              octet2 = int(ipnum / 65536) % 256
              octet3 = int(ipnum / 256) % 256
              octet4 = int(ipnum) % 256
              if octet4 == 0 :
                  self.labelTextInterface = 'N.A.'
              return '%d.%d.%d.%d' % (octet4, octet3, octet2, octet1)

          def getPrivateIP():
              from jnius import autoclass
              PythonActivity = autoclass('org.renpy.android.PythonActivity')
              SystemProperties = autoclass('android.os.SystemProperties')
              Context = autoclass('android.content.Context')
              wifi_manager = PythonActivity.mActivity.getSystemService(Context.WIFI_SERVICE)
              ipAddress = wifi_manager.getConnectionInfo()
              ipAddress = ipAddress.getIpAddress()
              ipAddress = int_to_ip(int(ipAddress))
              return str(ipAddress)

          self.labelTextInterface = getPrivateIP()

      except Exception:
          import traceback
          traceback.print_exc()
          self.labelTextInterface = 'N.A.'
```

4.4 Práce s GPS souřadnicemi

Pro výpis GPS souřadnic jsem použil modul `plyer`. Modul mi umožňuje komunikaci s GPS přijímačem mobilního zařízení. Implementované funkce v modulu `location` (výpis aktuálních souřadnic) stále přistupují k systémovým datům mobilního zařízení a aktualizují zobrazované GPS souřadnice v rámci vytvořené aplikace.

Souřadnice jsou vypsány pod sebou jako zeměpisná šířka (*latitude*) a zeměpisná délka (*longitude*) viz již zmíněny obr. 4.4.

Může nastat situace, kdy v mobilním zařízení nemusí být přítomen GPS přijímač. Tuto situaci ošetří aplikace při spuštění a následně upozorní uživatele. Jestliže se GPS přijímač v mobilním zařízení nachází, ale je ve vypnutém stavu, funkce `status` uživatele upozorní na jeho vypnutý stav.

Po zapnutí GPS přijímače uživatelem společně se zobrazením GPS souřadnic v aplikace se výstup funkce `status` změní na „OK“.

Funkce pro získání polohy zařízení je definována ve třídě `TestApp(App)` viz výpis funkce 4.3.

Výpis 4.3: Funkce pro získání GPS souřadnic

```
1
2   def start(self, minTime, minDistance):
3       gps.start(minTime, minDistance)
4
5   @mainthread
6   def on_location(self, **kwargs):
7         try:
8             self.gps_location_lat = '{lat}'.format(**kwargs)
9             self.gps_location_lon = '{lon}'.format(**kwargs)
10            self.gps_location = '%s, %s' % (self.gps_location_lat, self.gps_location_lon)
11            self.GPSStatus = 'GPS is OK'
12
13            i = 0
14            while (i != 1):
15                i=1
16                self.saveOutput()
17                break
18
19        except:
20            import traceback
21            traceback.print_exc()
22            self.gps_location_lat = 'N.A.'
23            self.gps_location_lon = 'N.A.'
```

4.4.1 Využití geolokačních databází

Hlavním úkolem aplikace je párování veřejné IP adresy a GPS souřadnic. Takových databázi již existuje spousty, nicméně je nutno podotknout, že jsou v některých případech značně nepřesné. Za pomocí modulu `simplejson` a webových aplikací lze tyto data získat a dále s nimi pracovat. Lze následně vzájemně porovnat v rámci jejich databázové odchylky.
Zdroje webových aplikací:

- `http://freegeoip.net`,
- `http://ip-api.com`,
- `http://ipinfo.io`.

Pro představu funkce pro práci s použitou webovou aplikací `FreeGeoIp` je definována viz výpis funkce 4.4.

Výpis 4.4: Funkce pro získání dat z webové aplikace FreeGeoIP

```
def setFreeGeoIP(self):
      try:
          def getFreeGeoIP():
              latitude = load(urlopen('http://freegeoip.net/json',timeout=6))['latitude']
              longitude = load(urlopen('http://freegeoip.net/json',timeout=6))['longitude']
              city = load(urlopen('http://freegeoip.net/json',timeout=6))['city']
              region = load(urlopen('http://freegeoip.net/json',timeout=6))['region_name']
              country = load(urlopen('http://freegeoip.net/json',timeout=6))['country_name']
              location = ('Freegeoip.net␣Database:␣%s,␣%s␣\nCity:␣%s␣\nRegion:␣%s␣\nCountry:␣%s'
              % (latitude, longitude, city,region, country))
              return location

          self.labelTextFreeGeoIP = getFreeGeoIP()

      except Exception:
          import traceback
          traceback.print_exc()
          self.labelTextFreeGeoIP = 'Freegeoip.net␣Database:␣N.A.'
```

4.4.2 Vizualizace získaných výstupů geolokačních databází

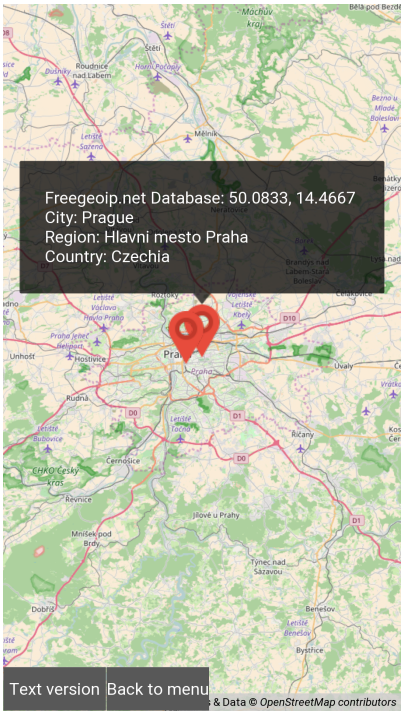
Klíčovým okamžikem při práci s výstupními daty byla vizualizace GPS souřadnic, kde bylo vhodné srovnat mezi sebou databázové souřadnice GPS se souřadnicemi zařízení.

Použil jsem veřejnou IP adresu zařízení pro zjištění dat, mezi která patří GPS souřadnice, město, kraj a stát. Hlavní parametr, ze kterého se doprovodné informace získavají, jsou GPS souřadnice. Vizualizace dat probíhá ve dvou zobrazeních, a to textově i graficky.

Textová forma znázorňuje jednotlivě získaná databázová data spolu s GPS souřadnicemi zařízení zobrazených a zarovnaných pod sebou viz obr. 4.5.

Grafické zobrazení využívá veřejné databáze map `Open StreetMap`, kde pomocí modulu `MapView` je aplikace schopna zobrazit v konkrétních bodech jednotlivé GPS souřadnice. Toto zobrazení považuji za nejefektivnější z hlediska porovnání odchylky mezi databázovými souřadnicemi viz obr. 4.6.

Obr. 4.5: Ukázka textové vizualizace



Obr. 4.6: Ukázka grafické vizualizace

4.5 Realizace přenosu dat

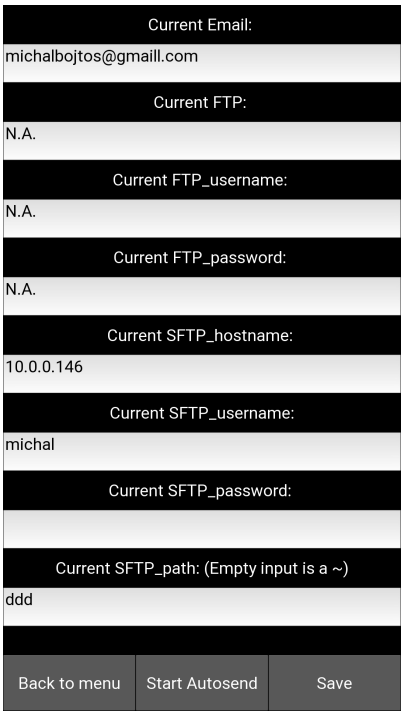
Výstupní data aplikace, mezi která patří GPS souřadnice, veřejná a privátní IP adresa, jsou ukládána do textového souboru ve tvaru `VeřejnáIP-ČasovéRazítko`, podrobný popis funkce viz výpis funkce 4.5 . Textový soubor tvoří klíčová uživatelská data pro následné použití protokolů sloužící k přenosu. Výstupní data textového souboru se ukládají ve tvaru, viz obr. 4.7.

Výpis 4.5: Funkce pro získání časového razítka textového souboru

```
1  def setTimestamp(self):
2        import datetime
3        import time
4        try:
5            def getTimestamp():
6                ts = time.time()
7                timestamp = datetime.datetime.fromtimestamp(ts).strftime('%Y-%m-%d_%H:%M:%S')
8                print timestamp
9                return timestamp
10
11           self.labelTimeStamp = getTimestamp()
12       except Exception:
13           import traceback
14           traceback.print_exc()
15           self.labelTimeStamp = 'N.A.'
```

Obr. 4.7: Ukázka výstupu textového souboru

Uložení výstupních dat aplikace probíhá viz výpis funkce A.2. Stejným způsobem se ukládá konfigurační nastavení, které slouží pro implementaci základních parametrů používaných protokolů služeb SMTP, FTP a SFTP. Parametry protokolů nastavíme v grafickém prostředí aplikace viz obr. 4.8.



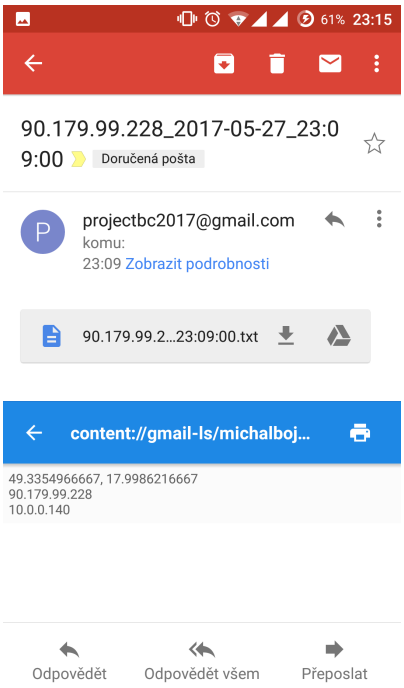
Obr. 4.8: Nastavení parametrů jednotlivých služeb

Funkce pro uložení uživatelského nastavení je definována ve třídě `TestApp(App)` viz výpis funkce A.3.

4.5.1 Přenos pomocí služby SMTP

E-mailovou komunikaci mezi aplikací a e-mailovou schránkou uživatele zabezpečuje služba SMTP. Pro tento účel byla založena adresa `projectbc2017@gmail.com`, díky které lze využít externí SMTP server poskytovatele e-mailové schránky.

Za pomocí modulu `smtplib` ve spolupráci s vytvořenou funkcí pro odesílání výstupních dat se aplikace připojí k vytvořené e-mailové adrese a přihlásí se pomocí přihlašovacích údajů. Aplikace dále připojí přílohu `VeřejnáIP-ČasovéRazítko.txt` a odešle na e-mailovou schránku uživatele viz obr. 4.9. Průběh jednotlivých kroků modulu `smtplib` viz výpis funkce 4.6.



Obr. 4.9: Úspěšný přenos pomocí protokolu SMTP

Výpis 4.6: Funkce pro přenos dat pomocí protokolu SMTP

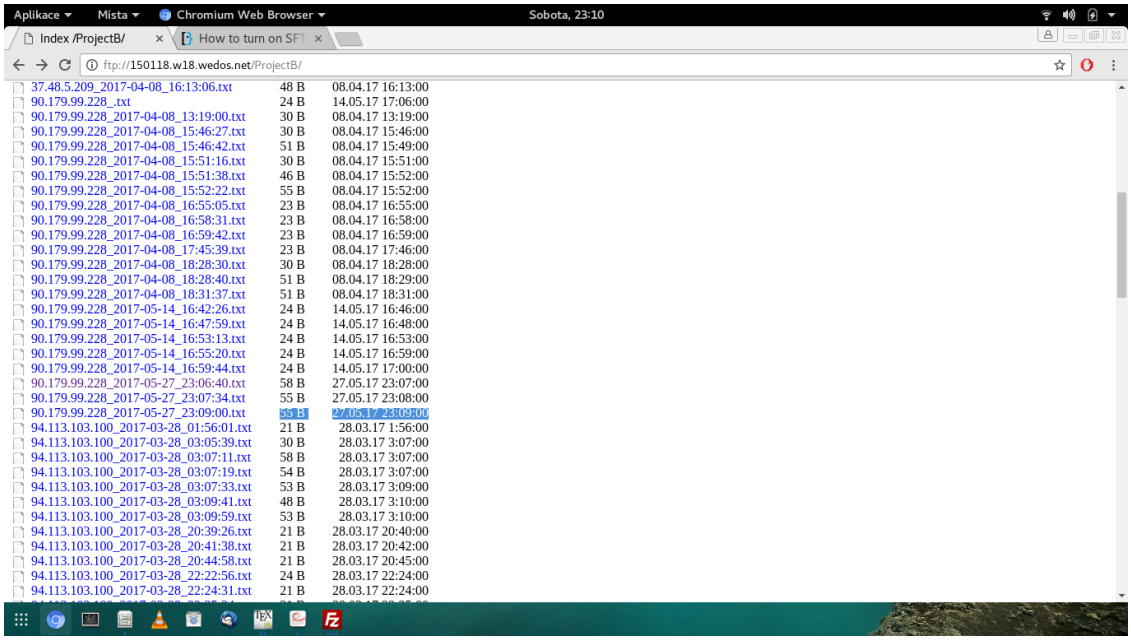
```
def SendEmail(self):
      try:
        import time
        time.sleep(2)

          import smtplib
          to = self.labelTextEmail
          message = MIMEMultipart('related')
          user = 'projectbc2017@gmail.com'
          password = '*********'
          message['From'] = user
          message['Subject'] = '%s_%s' % (self.labelTextGlobal,self.labelTimeStamp)
          smtpserver = SMTP('smtp.gmail.com', 587, timeout=6)
          smtpserver.ehlo()
          smtpserver.starttls()
          smtpserver.ehlo
          smtpserver.login(user, password)
          f = file('/storage/emulated/0/output.txt')
          attachment = MIMEText(f.read())
          attachment.add_header('Content-Disposition', 'attachment',
          filename='%s_%s.txt' %(self.labelTextGlobal,self.labelTimeStamp))
          message.attach(attachment)
          smtpserver.sendmail(user, to, message.as_string())
          smtpserver.close()
          self.labelTextEmailStatus = 'OK'

      except Exception:
          import traceback
          traceback.print_exc()
          if to == 'N.A.':
              self.labelTextEmailStatus = 'Not set'
          else:
              self.labelTextEmailStatus = 'Fail'
```

4.5.2 Přenos pomocí služby FTP

Protokol FTP jsem pro přenos dat implementoval z důvodu jeho značného využívání mezi uživateli a jednoduchosti při přenosu typických dat, v tomto případě `.txt`.

Modul `ftplib` získá nastavené proměnné parametry umožňující přístup do úložiště FTP serveru. Dojde ke kontrole přítomnosti adresář `ProjectB`. Pokud již adresář existuje, přistoupí do něj a uloží zde výstupní data.
Je-li situace opačná, vytvoří na FTP úložišti tento adresář a následně s ním dále pracuje viz obr. 4.10. Přístupové metody funkce pro přenos pomocí služby FTP viz výpis funkce 4.7.

Po ukončení přenosové relace výstupních dat dochází k ukončení přístupu do FTP serveru. Hlavním důvodem je datové vytěžování aplikace a zachování funkčnosti.



Obr. 4.10: Úspěšný přenos pomocí protokolu FTP

Výpis 4.7: Funkce pro přenos dat pomocí protokolu FTP

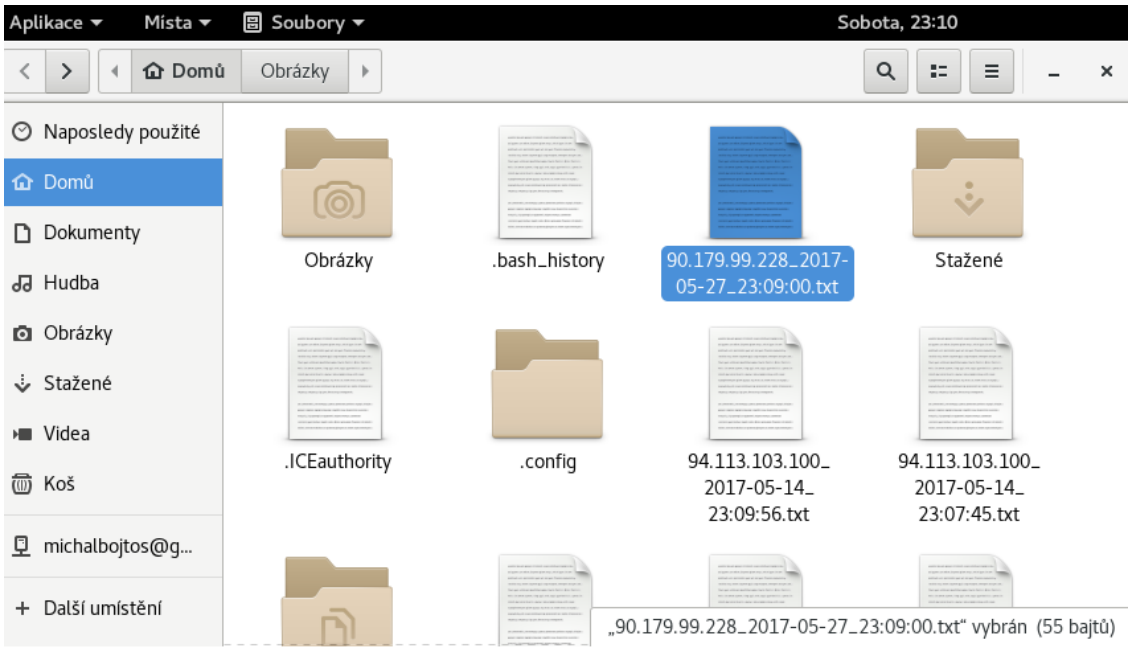
```
def SendFTP(self):
      try:
        import time
          time.sleep(2)
          import ftplib
          print self.labelTextFTP,self.labelTextFTP_username
          ,self.labelTextFTP_password
          server = self.labelTextFTP
          print server
          user = self.labelTextFTP_username
          password = self.labelTextFTP_password
          ftpCon = FTP()
          ftpCon.connect(server,timeout=6)
          ftpCon.login(user=user, passwd=password)

          if 'ProjectB' in ftpCon.nlst():  # check if 'foo' exist inside 'www'
              print 'YES'
              ftpCon.cwd('ProjectB')  # change into "foo" directory
              file = open(('/storage/emulated/0/output.txt'),'rb')
              ftpCon.storbinary(('STOR /ProjectB/%s_%s.txt'
              % (self.labelTextGlobal, self.labelTimeStamp)), file)
              file.close()

          else:
              print 'NO'
              ftpCon.mkd('ProjectB')  # Create a new directory called foo on the server.
              ftpCon.cwd('ProjectB')  # change into 'foo' directory
              file = open(('/storage/emulated/0/output.txt'),'rb')
              ftpCon.storbinary(('STOR /ProjectB/%s_%s.txt'
              % (self.labelTextGlobal, self.labelTimeStamp)), file)
              file.close()

          ftpCon.close()
          self.labelTextFTPStatus = 'OK'

      except Exception:
          import traceback
          traceback.print_exc()
          if ((server == 'N.A.') and (user == 'N.A.') and (password == 'N.A.')):
              self.labelTextFTPStatus = 'Not set'
          else:
              self.labelTextFTPStatus = 'Fail'
```

4.5.3 Přenos pomocí služby SFTP

Při přenosu dat s ohledem na význam a možnosti přenosu je vždy potřeba klást důraz na bezpečnost. K tomuto účelu jsem neopomenul implementovat protokol SFTP.

Princip přenosu protokolu je obdobný, jako je tomu u FTP serveru, přičemž přenos je zabezpečený. Uživatel v konfiguračním nastavení zadá adresu SSH serveru a přístupové údaje. Přístupové údaje se skládají z parametrů jména a hesla viz výpis funkce 4.8 .

Pro přenos tohoto typu existuje mnoho modulů, avšak z hlediska funkčního řešení byl vybrán modul `paramiko`. Z již zadaných parametrů v konfiguračním nastavení se aplikace přihlásí na SFTP server a na zvolenou cestu v konfiguračním nastavení (pokud není nastavena, výchozí cesta je domovský adresář) uloží výstupní data aplikace viz obr. 4.11. Po přenosu zde nemá smysl zachovávat přenosovou relaci. Dochází proto k ukončení ihned po přenosu.



Obr. 4.11: Úspěšný přenos pomocí protokolu SFTP

Výpis 4.8: Funkce pro přenos dat pomocí protokolu SFTP

```
    def SendSFTP(self):
        try:
          import time
            time.sleep(2)
            import paramiko
            hostname = self.labelTextSFTP_hostname
            username = self.labelTextSFTP_username
            password = self.labelTextSFTP_password
            port = 22
            timeout = 6
            path = self.labelTextSFTP_path
            print path
            transport = paramiko.Transport(hostname, port, timeout)
            transport.connect(username=username, password=password)
            sftp = paramiko.SFTPClient.from_transport(transport)
            localpath = '/storage/emulated/0/output.txt'
            filepath = '%s%s_%s.txt' % (path,self.labelTextGlobal, self.labelTimeStamp)
            sftp.put(localpath,filepath)
            sftp.close()
            transport.close()
            self.labelTextSFTPStatus = 'OK'

        except Exception:
            import traceback
            traceback.print_exc()
            if ((hostname == 'N.A.') and (username == 'N.A.') and (password == 'N.A.')):
                self.labelTextSFTPStatus = 'Not set'
            else:
                self.labelTextSFTPStatus = 'Fail'
```

4.5.4 Automatické odesílání pozičních dat aplikace

Aplikace je vytvořena s důrazem na uživatelské ovládání. Snahou bylo vytvořit automatizovanou funkci, která má za úkol dohlížet na změnu veřejné IP adresy.

Vytvořená funkce viz výpis funkce 4.9 každých 90 sekund zkontroluje, zda je veřejná IP adresa stále shodná. Pokud zde nenastane shoda, aplikace uloží a odešle výstupní data. Jaké protokoly má zvolit, to již řeší konfigurační nastavení. Jestliže se ve vstupním poli přístupových údajů nachází výchozí hodnota `N.A.`, protokol není použit. V opačném případě přenos ukončí statusem `OK` nebo `Fail` viz obr. 4.12.

Výpis 4.9: Funkce pro přenos dat při změně veřejné IP adresy

```
def AutoSend(self,dt):
self.setGlobalIP()
self.setPrivateIP()
      try:
          if (self.currentIP != self.labelTextGlobal):
              self.saveOutput()
              self.SendEmail()
              self.SendFTP()
              self.SendSFTP()
              self.currentIP = self.labelTextGlobal
              return self.currentIP
      except Exception:
          import traceback
          traceback.print_exc()

def Auto(self,number):
      if number == 1:
          Clock.schedule_interval(self.AutoSend,90/1.)
          popup = Popup(title='AutoSend'
          , content=Label(text='Check␣your␣Public␣IP␣every␣90s,\nif␣it␣changed,send'),
                        auto_dismiss=True, size_hint=(None, None), size=(750, 400))
          popup.open()
      if number == 0:
          Clock.unschedule(self.AutoSend)
```

Obr. 4.12: Ukázka jednotlivých stavů po přenosu dat aplikací

4.6 Porovnání textově a graficky vůči výsledkům geolokačních databází

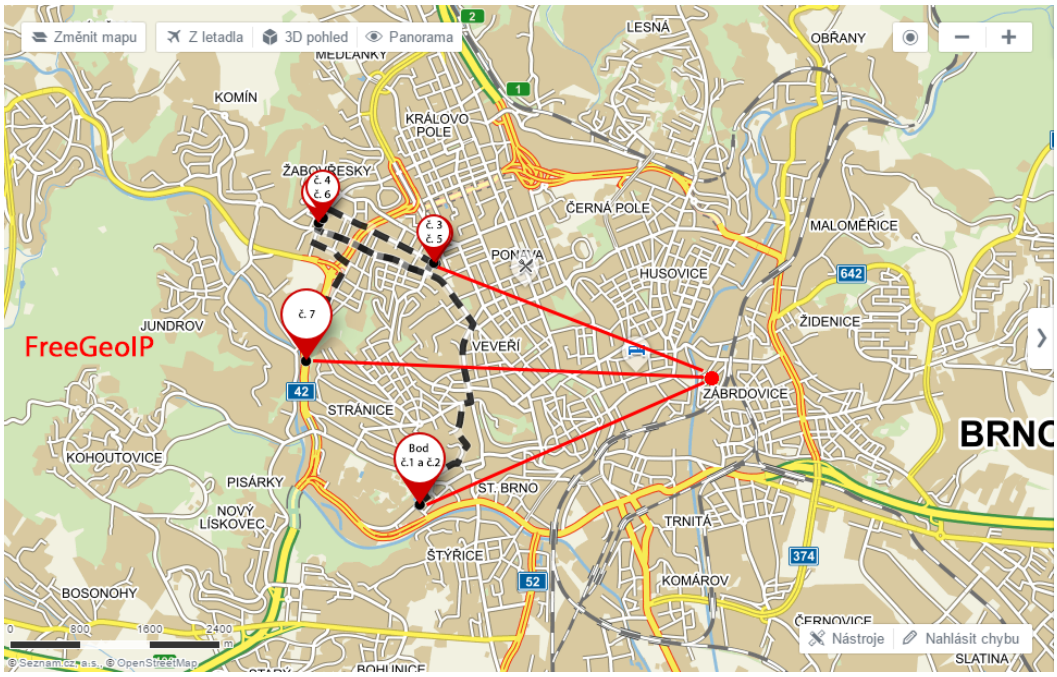
Hlavní již zmíněnou funkcí aplikace je spárování veřejně IP adresy společně s GPS souřadnicemi. Využil jsem funkci automatického odesílání dat pro sběr informací v časovém rozptylu běžného pracovního dne. Mobilní zařízení bylo nutné uvést do stavu „vypnuto“ úsporný režim a zařízení připojit k externí baterii z důvodu cestování.

Cílem bylo porovnat GPS souřadnice geolokačních databázi se skutečnými a změřit jejich přibližnou odchylku. Výsledky porovnání GPS souřadnic jsou uvedeny pro FreeGeoIP viz tab. 4.1, pro IP-API viz tab. 4.2 a pro IPINFO viz tab. 4.3. Výstup pokusu byl s využitím `www.mapy.cz` vytvořen i v grafické podobě. Z důvodu čitelnosti mapových podkladů odchylky větší než 5 kilometrů nebyly zaznamenány. Srovnání jednotlivých odchylek v grafické podobě viz obr. 4.13, obr. 4.14 a obr. 4.15.

Odchylky byly spočítány webovovou aplikací `http://www.gpsvisualizer.com/calculators`. Z výsledků vyplývá, že pokud jsem připojen pomocí mobilních dat, výsledek je značně nepřesný působením plošného pokrytí mobilních operátorů na našem území. V opačném případě, pokud mobilní zařízení využívá internetového připojení místního poskytovatele internetu, odchylka se pohybuje v řádech jednotek až desítek kilometrů.

Tab. 4.1: Porovnání získaných dat z webové aplikace FreeGeoIP

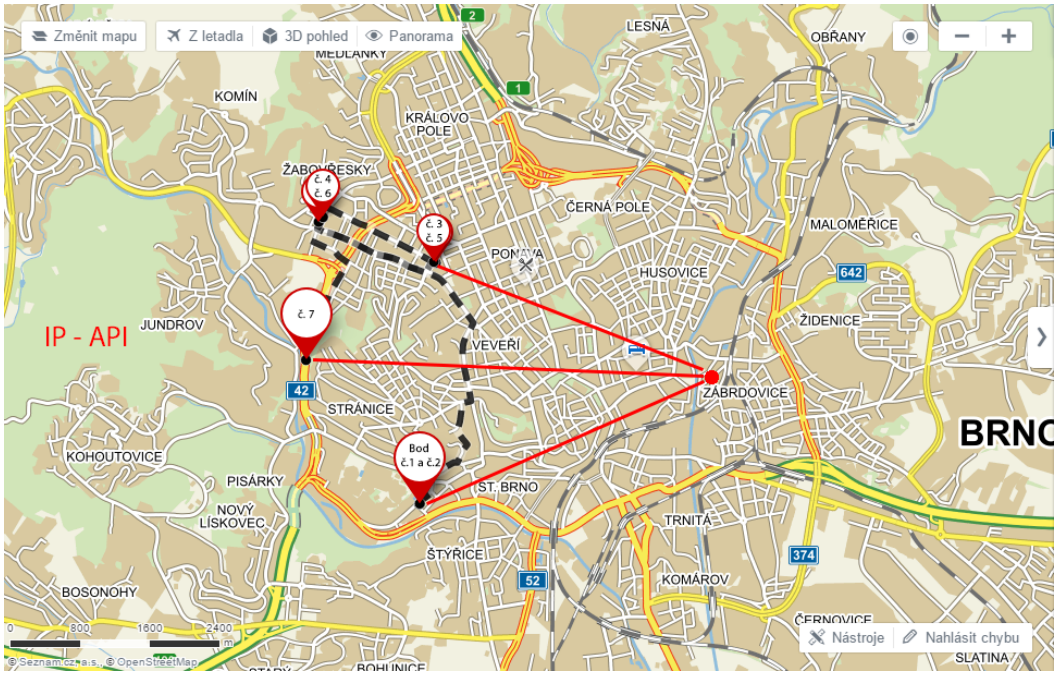
IP adresa	FreeGeoIP	Skutečnost	Odchylka [km]	Čas záznamu
94.113.103.100	49.2, 16.6333	49.18652, 16.58589	3.767	08:31
37.188.233.95	50.0848, 14.4112	49.18657, 16.5859933	186.162	08:58
86.49.84.112	49.2, 16.6333	49.21136, 16.588522	3.499	10:34
37.188.156.72	50.0848, 14.4112	49.2155633, 16.57003666	183.432	11:51
86.49.84.112	49.2, 16.6333	49.211455, 16.588197	3.525	16:56
37.188.191.144	50.0848, 14.4112	49.21605166, 16.570715	183.444	17:19
94.113.103.100	49.2, 16.6333	49.2013583, 16.568035	4.759	17:50



Obr. 4.13: Ukázka grafického srovnání FreeGeoIP

Tab. 4.2: Porovnání získaných dat z webové aplikace IP-API

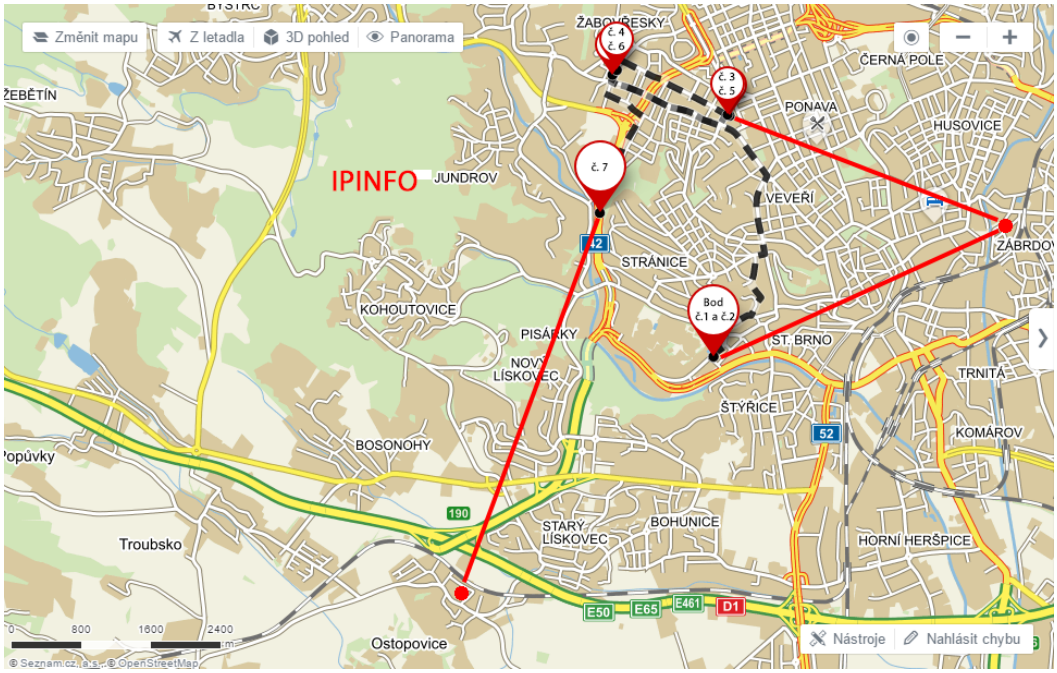
IP adresa	IP-API	Skutečnost	Odchylka [km]	Čas záznamu
94.113.103.100	49.2, 16.6333	49.18652, 16.58589	3.767	08:31
37.188.233.95	50.0755, 14.4378	49.18657, 16.5859933	183.998	08:58
86.49.84.112	49.2, 16.6333	49.21136, 16.588522	3.499	10:34
37.188.156.72	50.0755, 14.4378	49.2155633, 16.57003666	181.267	11:51
86.49.84.112	49.2, 16.6333	49.211455, 16.588197	3.525	16:56
37.188.191.144	50.0833, 14.4667	49.21605166, 16.570715	179.963	17:19
94.113.103.100	49.2, 16.6333	49.2013583, 16.568035	4.759	17:50



Obr. 4.14: Ukázka grafického srovnání IP-API

Tab. 4.3: Porovnání získaných dat z webové aplikace IPINFO

IP adresa	IPINFO	Skutečnost	Odchylka [km]	Čas záznamu
94.113.103.100	49.1618, 16.5462	49.18652, 16.58589	3.992	08:31
37.188.233.95	50.0848, 14.4112	49.18657, 16.5859933	186.162	08:58
86.49.84.112	49.2000, 16.6333	49.21136, 16.588522	3.499	10:34
37.188.156.72	50.0848, 14.4112	49.2155633, 16.57003666	183.432	11:51
86.49.84.112	49.2000, 16.6333	49.211455, 16.588197	3.525	16:56
37.188.191.144	50.0848, 14.4112	49.21605166, 16.570715	183.444	17:19
94.113.103.100	49.1618, 16.5462	49.2013583, 16.568035	4.679	17:50



Obr. 4.15: Ukázka grafického srovnání IPINFO

4.7 Vytvoření grafického prostředí pro ovládání aplikace

Grafické prostředí jsem vytvořil za pomocí implementovaných modulů ve frameworku Kivy, jedná se zejména o modul `ScreenManager` spolu s doplňujícími prvky jako jsou `Button` a `Label`. V úvodní části je prostředí načteno z textového souboru, ve kterém jsou uvedené grafické prvky pro ovládání aplikace.

Jednotlivá okna aplikace se následně zadefinují do třídy s parametrem `Screen`. K zobrazení těchto oken dojde díky zavolání tříd v hlavní startovací třídě aplikace `build` viz příloha A.5.

Po spuštění aplikace se vykreslí úvodní menu spolu s aktivací tlačítek a funkcí pro zjištění IP adres a GPS souřadnic. Spodní část každého okna tvoří tlačítka pro ovládání aktuálního zobrazeného okna viz obr. 4.3.

Dalším krokem je využít nabídky menu. Po stisknutí tlačítka `Map` se uživatel dostane do vizuálního zpracování dat polohových databázi a GPS souřadnic. Je zde také možnost přepnout zobrazení do grafického režimu, který uživatelsky lépe pracuje při odhalování odchylky jednotlivých databází.

Po návratu do hlavního menu je dalším klíčovým tlačítkem `Settings`. Po stisknutí tohoto tlačítka se uživatel dostane do nastavení přístupových údajů pro přenos výstupních dat. Je nutné, aby uživatel po každé změně přístupových údajů provedl stisknutí tlačítka `Save`, díky kterému se údaje uloží do zvláštního textového souboru v mobilním zařízení. Po spuštění aplikace je mobilní zařízení prohledáno a pokud se tam tento jedinečný soubor s údaji pro nastavení nachází, doplní údaje v aplikaci již za uživatele, viz obr. 4.8. Dále se zde nachází tlačítko pro zapnutí/vypnutí automatického odesílaní dat při změně veřejné IP adresy v intervalu 90 sekund.

Tlačítko `Send` hlavního menu zabezpečuje odeslání dat pomocí služeb SMTP, FTP a SFTP. Po provedení odeslání se uživateli zobrazí okno, ve kterém má možnost zjistit konečné průběhy jednotlivých služeb pomocí stavů viz obr. 4.12.
Možnosti stavu služeb:

- OK – Přenos proběhl v pořádku,
- Not set – přístupové údaje nebyly ke službě nastaveny, přenos neproběhl,
- Fail – selhání při přenosu dat, špatně nastavené přístupové údaje či problém na straně serveru.

5 ZÁVĚR

Zadáním bakalářské práce bylo vytvořit aplikaci pro určení polohy mobilních zařízení. Za pomocí uvedené literatury, jazyka Python, frameworku Kivy a dodatečných modulů jsem sestrojil aplikaci, která dokáže dle zadání určit polohu mobilního zařízení a následně párovat veřejnou IP adresu se souřadnicemi GPS. Implementováním služeb pro přenos dat ve formě odesílaní na emailovou schránku uživatele, FTP a SFTP server je aplikace schopna plnit geolokační databázi.

V teoretické části jsem postupně rozebral problematiku geolokace mobilního zařízení a IP adresace v síti Internet společně s mechanismy technologie NAT. Dále jsem zde uvedl základní principy jednotlivých služeb FTP, SFTP a SMTP sloužící k přenosu dat pro zajištění přenosu jednotlivých výstupů aplikace.

Aplikaci jsem sestrojim v operačním systému Fedora verze 26. Systém jsem zvolil z důvodu kompilace aplikace do formátu .apk. Formát bylo nutné vytvořit pro snadné nainstalování aplikace do mobilního zařízení vyžívající operační systém Android.

V prvních krocích tvorby jsem zajistil získání veřejné a privátní IP adresy. Pro docílení této etapy jsem využil komunikace mezi aplikací a webovou aplikací, díky které došlo k získání veřejné IP adresy. Pomocí API Android byla získána privátní IP adresa. Lokalizace zařízení zabezpečuje modul Plyer, skládající se z přístupových metod k GPS přijímači mobilního zařízení. Výstupní data ve formě IP adres a GPS souřadnic jsou poté párována, uložena do textového souboru a připraveny pro přenos.

Ovládání aplikace je zabezpečeno grafickým rozhráním frameworku Kivy, kde lze nastavit přístupové parametry jednotlivých přenosů uživatelem. Další podstatnou funkcí je vytvořená grafická a textová vizualizace GPS souřadnic se získanými daty geolokačních databází.

Sestrojením aplikace pro určení polohy mobilního zařízení byl naplněn hlavní cíl této práce. Aplikace je připravena pro další vývoj zabývající se problematikou práce.

LITERATURA

[1] Systém Galileo byl modernizován a je zpět v plném provozu. In: *EUROPEAN SPACE AGENCY* [online]. Česká republika: EUROPEAN SPACE AGENCY, 2015 [cit. 2016-11-26]. Dostupné z: http://www.esa.int/cze/ESA_in_your_country/Czech_Republic/System_Galileo_byl_modernizovan_a_je_zpet_v_plnem_provozu

[2] ŠEBESTA, J. *Globální navigační systémy* [online]. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav radioelektroniky, 2012, 132 s. [cit. 2016-11-30]. ISBN 978-80-214-4500-0. Dostupné z: http://www.urel.feec.vutbr.cz/~sebestaj/RAR/literatura/Globalni_navigacni_systemy.pdf

[3] MODRÁK, Z. *Webová aplikace zobrazující polohu IP stanic*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2015, 57 s. Diplomová práce. Vedoucí práce Doc. Ing. Dan Komosný, Ph.D.

[4] O technologii GPS. In: *Mio* [online]. Česká republika: Mio, 2012 [cit. 2016-12-08]. Dostupné z: https://eu.mio.com/cs_cz/global-positioning-system_jaky-signal-pouziva-gps.htm

[5] RYDVAL, S. Princip a fungování GPS. In: *NaWEBka* [online]. Česká republika: Rydval, 2005 [cit. 2016-12-08]. Dostupné z: http://www.rydval.cz/phprs/view.php?cisloclanku=2005110301

[6] JEŘÁBEK, J. *Komunikační technologie*. Brno: Vysoké učení technické v Brně Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2013, 172 s. ISBN 978-80-214-4713-4.

[7] KABELOVÁ, A. a L. DOSTÁLEK. *Velký průvodce protokoly TCP/IP a systémem DNS*. 5., aktualiz. vyd. Brno: Computer Press, 2008, 488 s. ISBN 978-80-251-2236-5.

[8] BOHÁČ, Leoš. *Protokoly aplikační vrstvy - DNS, SMTP, HTTP*. České vysoké učení technické v Praze Fakulta elektrotechnická, 2013, 47 s. Dostupné také z: http://data.cedupoint.cz/oppa_e-learning/2_KME/048.pdf

[9] HRDINKA, Jan. *Realizace zabezpečení FTP serveru (SFTP a FTPS) a zabezpečeného HTTP (HTTPS) serveru*. Vysoká škola ekonomická v Praze, 2011, 51 s. Dostupné také z: http://info.sks.cz/www/zavprace/soubory/76160.pdf. Bakalářská práce. Vedoucí práce Ing. David Kimánek, Ph.D.

[10] KOMOSNÝ, Dan. *Síťové operační systémy*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2017, 126 s. Dostupné také z: https://moodle.vutbr.cz/pluginfile.php/283869/mod_folder/content/0/BSOS_teorie.pdf?forcedownload=1

SEZNAM SYMBOLŮ, VELIČIN A ZKRATEK

A-GPS asistovaný polohový systém – Assisted Global Navigation System

AP přístupový bod – Access Point

APTLD asociace pro správu národních domén Asie a Pacifiku – Asia Pacific Top Level Domain Association

ASO organizace pro správu IP adres – The Address Supporting Organisation

ccNSO organizace zajišťující správný chod národních domén – The Country Code Names Supporting Organisation

CENTR rada evropských národních doména nejvyšší úrovně registrátorů – Council of European National Top-Level Domain Registries

CIDR sada pravidel pro beztřídní směrování – Classless Inter - Domain Routing

FTP internetový protokol pro přenos souborů – File Transfer Protocol

GLONASS globální družicový polohový systém ruské armády – Globalnaja navigacionnaja sputnikovaja sistěma

GPS globální polohový systém – Global Positioning System

GNSO organizace pro správu generických domén – Generic Names Supporting Organisation

GUI grafické uživatelské rozhrání – Graphical User Interface

IAB výbor pověřený dohledem nad technickým a inženýrským rozvojem Internetu – Internet Architecture Board

IANA autorita pro globální alokaci IP adres – Internet Assigned Numbers Authority

ICANN organizace pro správu protokolů RFC a IP adresního prostoru – Internet Corporate for Assigned Names and Numbers

IESG orgán určený IAB k dohledu nad prací organizace IETF – Engineering Steering Group

IETF komise pro technickou stránku internetu – Internet Engineering Force

IP protokol síťové vrstvy – Internet protocol

IRTF komise pro technickou stránku internetu – Internet Research Task Force

ISO mezinárodní organizace pro normalizaci – International Standards Organization

ISOC mezinárodní nezisková organizace – Internet Sociaty

LAC TLD organizace ccTLDs Latinské Ameriky a Karibiku – Organización de ccTLDs Latinoamericanos y del Caribe

NAT překlad síťových adres – Network Address Translator)

OS operační systém – Operation System

OSI propojení otevřených systémů – Open Systen Interconnection

PPS přesný polohový systém– Precision Positioning Service

QZSS třídružicový regionální navigační systém pro Japonsko – Quasi - Zenith Satellite System

RFC označení řady dokumentů popisujících internetové protokoly, systémy apod. – Request For Comments

RIR organizace zajišťující alokaci adresního prostoru v rámci své geografické působnosti – Regional Internet Registeries

SFTP internetový protokol pro bezpečný přenos souborů – SSH File Transfer Protocol

SMTP internetový protokol pro přenos zpráv elektronické pošty – Simple Mail Transfer Protocol

SPS standardní polohový systém – Standard Positioning Service

SSH zabezpečený komunikační protokol – Secure Shell

TCP protokol transportní vrstvy – Transmission control protocol

SEZNAM PŘÍLOH

A ZDROJOVÝ KÓD APLIKACE

A.1 Hlavní použité moduly v aplikaci

V této kapitole jsou umístěny standardní části zdrojového kódu aplikace. Další použité zdrojové kódy jsou uvedeny v textu práce pro zdůraznění jejich atypičnosti.

Výpis A.1: Hlavní použité moduly v aplikaci

```
1  from kivy.app import App
2  from kivy.lang import Builder
3  from kivy.uix.screenmanager import ScreenManager, Screen
4  from plyer import gps
5  from kivy.properties import StringProperty
6  from simplejson import load
7  from urllib2 import urlopen
8  from kivy.clock import Clock, mainthread
9  from email.mime.text import MIMEText
10 from email.mime.multipart import MIMEMultipart
11 from ftplib import FTP
12 from smtplib import SMTP
13 from kivy.garden.mapview import mapview
14 import sys
15 from kivy.uix.modalview import ModalView
16 from kivy.uix.popup import Popup
17 from kivy.uix.label import Label
```

A.2 Funkce pro uložení výstupních dat do textového souboru

Výpis A.2: Funkce uložení výstupních dat do textového souboru

```
1  def saveOutput(self):
2        try:
3            self.setTimestamp()
4            fob = open(('/storage/emulated/0/output.txt'), 'w') #/storage/emulated/0/output.txt
5            fob.write(('%s␣\n%s␣\n%s') %(self.gps_location,self.labelTextGlobal,self.labelTextInterface))
6            fob.close()
7
8        except Exception:
9            import traceback
10           traceback.print_exc()
11           fob = open(('/storage/emulated/0/output.txt'), 'w')
12           fob.write(('%s␣\n%s␣\n%s') % (self.gps_location, self.labelTextGlobal, self.labelTextInterface))
13           fob.close()
```

A.3 Funkce pro uložení uživatelského nastavení

Výpis A.3: Funkce pro uložení uživatelského nastavení

```
def saveSettings
(self,email,ftp,ftp_username,ftp_password,sftp,sftp_username,sftp_password,sftp_path)

      try:
          fob = open('/storage/emulated/0/ProjectB_settings.txt', 'w') #storage/emulated/0/settings.tx
          fob.write(email + "\n")
          fob.write(ftp + "\n")
          fob.write(ftp_username + "\n")
          fob.write(ftp_password + "\n")
          fob.write(sftp + "\n")
          fob.write(sftp_username + "\n")
          fob.write(sftp_password + "\n")
          fob.write(sftp_path + "\n")
          fob.close()

      except Exception:
          import traceback
          traceback.print_exc()
          fob = open('/storage/emulated/0/ProjectB_settings.txt', 'w')
          fob.write("Exception, please, try again.")
          fob.close()
```

A.4 Funkce pro získaní časového razítka textového souboru

Výpis A.4: Funkce pro získání časového razítka textového souboru

```
def setTimestamp(self):
      import datetime
      import time
      try:
          def getTimestamp():
              ts = time.time()
              timestamp = datetime.datetime.fromtimestamp(ts).strftime('%Y-%m-%d_%H:%M:%S')
              print timestamp
              return timestamp

          self.labelTimeStamp = getTimestamp()
      except Exception:
          import traceback
          traceback.print_exc()
          self.labelTimeStamp = 'N.A.'
```

A.5 Grafické prostředí pro ovládání aplikace

Výpis A.5: Vytvoření grafických prvků pro ovládání aplikace

```
Builder.load_string("""
<MenuScreen>:
    BoxLayout:
        orientation: 'vertical'
        Label:
            size_hint_y: None
            height: self.parent.height * 0.1
            text: app.GPSStatus
        Label:
            size_hint_y: None
            height: self.parent.height * 0.1
            text: app.gps_location
        Button:
            size_hint_y: None
            height: self.parent.height * 0.09
            text: 'Refresh IP addresses'
            on_release: app.setPrivateIP()
            on_release: app.setGlobalIP()
        Label:
            size_hint_y: None
            height: self.parent.height * 0.1
            text: 'Private IP: '+ app.labelTextInterface
        Label:
            size_hint_y: None
            height: self.parent.height * 0.1
            text: 'Public IP: '+ app.labelTextGlobal
        BoxLayout:
            orientation:'horizontal'
            Button:
                size_hint_y: None
                height: self.parent.height * 0.158
                text: 'Map'
                on_release: root.manager.current = 'map'
            Button:
                size_hint_y: None
                height: self.parent.height * 0.158
                text: 'Settings'
                on_release: root.manager.current = 'settings'
            Button:
                size_hint_y: None
                height: self.parent.height * 0.158
                text: 'Send'
                on_release: root.manager.current = 'confirm'
                on_release: app.saveOutput()
                on_release: app.SendEmail()
                on_release: app.SendFTP()
                on_release: app.SendSFTP()
            Button:
                size_hint_y: None
                height: self.parent.height * 0.158
                text: 'Quit'
                on_release: exit()
<DatabaseScreen>:
    BoxLayout:
        orientation: 'vertical'
        Label:
            size_hint_y: None
            height: self.parent.height * 0.15
            text: 'Your GPS: '+ app.gps_location
        Button:
            size_hint_y: None
            height: self.parent.height * 0.09
            text: 'Refresh IP Databases'
            on_release: app.setFreeGeoIP()
            on_release: app.setIpApi()
            on_release: app.setIpInfo()
        Label:
            size_hint_y: None
            size_hint_x: None
            height: self.parent.height * 0.15
            width: self.parent.width * 1
            text: app.labelTextFreeGeoIP
        Label:
            size_hint_y: None
```

```
            size_hint_x: None
            height: self.parent.height * 0.15
            width: self.parent.width * 0.925
            text: app.labelTextIpApi
        Label:
            size_hint_y: None
            size_hint_x: None
            height: self.parent.height * 0.15
            width: self.parent.width * 0.90
            text: app.labelTextIpInfo
        BoxLayout:
            orientation: 'horizontal'
            Button:
                size_hint_y: None
                height: self.parent.height * 0.26
                text: 'Back to menu'
                on_release: root.manager.current = 'menu'
            Button:
                size_hint_y: None
                height: self.parent.height * 0.26
                text: 'Map version'
                on_release: root.manager.current = 'map'
<SettingsScreen>:
    email: email_input
    ftp: ftp_input
    ftp_username: ftp_username_input
    ftp_password: ftp_password_input
    sftp: sftp_input
    sftp_username: sftp_username_input
    sftp_password: sftp_password_input
    sftp_path: sftp_path_input
    BoxLayout:
        orientation: 'vertical'
        Label:
            size_hint_y: None
            height: self.parent.height * 0.055
            text: 'Current Email: '
        TextInput:
            size_hint_y: None
            height: self.parent.height * 0.055
            id: email_input
            text: app.labelTextEmail
        Label:
            size_hint_y: None
            height: self.parent.height * 0.055
            text: 'Current FTP: '
        TextInput:
            size_hint_y: None
            height: self.parent.height * 0.055
            id: ftp_input
            text: app.labelTextFTP
        Label:
            size_hint_y: None
            height: self.parent.height * 0.055
            text: 'Current FTP_username: '
        TextInput:
            size_hint_y: None
            height: self.parent.height * 0.055
            id: ftp_username_input
            text: app.labelTextFTP_username
        Label:
            size_hint_y: None
            height: self.parent.height * 0.055
            text: 'Current FTP_password: '
        TextInput:
            size_hint_y: None
            height: self.parent.height * 0.055
            id: ftp_password_input
            text: app.labelTextFTP_password
        Label:
            size_hint_y: None
            height: self.parent.height * 0.055
            text: 'Current SFTP_hostname: '
        TextInput:
            size_hint_y: None
            height: self.parent.height * 0.055
            id: sftp_input
            text: app.labelTextSFTP_hostname
        Label:
```

```
            size_hint_y: None
            height: self.parent.height * 0.055
            text: ' Current SFTP_username: '
        TextInput:
            size_hint_y: None
            height: self.parent.height * 0.055
            id: sftp_username_input
            text: app.labelTextSFTP_username
        Label:
            size_hint_y: None
            height: self.parent.height * 0.055
            text: 'Current SFTP_password: '
        TextInput:
            size_hint_y: None
            height: self.parent.height * 0.055
            id: sftp_password_input
            text: app.labelTextSFTP_password
        Label:
            size_hint_y: None
            height: self.parent.height * 0.055
            text: 'Current SFTP_path: (Empty input is a ~)'
        TextInput:
            size_hint_y: None
            height: self.parent.height * 0.055
            id: sftp_path_input
            text: app.labelTextSFTP_path
        BoxLayout:
            orientation: 'horizontal'
            Button:
                size_hint_y: None
                height: self.parent.height * 0.66
                text: 'Back to menu'
                on_release: root.manager.current = 'menu'
                on_release: app.saveSettings(email_input.text,ftp_input.text,ftp_username_input.text
                ,ftp_password_input.text,sftp_input.text,sftp_username_input.text,sftp_password_input.text
                ,sftp_path_input.text)
                on_release: app.setEmail()
                on_release: app.setFTP()
                on_release: app.setFTP_username()
                on_release: app.setFTP_password()
                on_release: app.setSFTP()
                on_release: app.setSFTP_username()
                on_release: app.setSFTP_password()
                on_release: app.setSFTP_path()
            ToggleButton:
                size_hint_y: None
                height: self.parent.height * 0.66
                text: 'Start Autosend' if self.state == 'normal' else 'Stop Autosend'
                on_state: app.Auto(1) if self.state == 'down' else app.Auto(0)
            Button:
                size_hint_y: None
                height: self.parent.height * 0.66
                text: 'Save'
                on_release: app.SavePopup()
                on_release: app.saveSettings(email_input.text,ftp_input.text,ftp_username_input.text
                ,ftp_password_input.text,sftp_input.text,sftp_username_input.text,sftp_password_input.text
                ,sftp_path_input.text)
                on_release: app.setEmail()
                on_release: app.setFTP()
                on_release: app.setFTP_username()
                on_release: app.setFTP_password()
                on_release: app.setSFTP()
                on_release: app.setSFTP_username()
                on_release: app.setSFTP_password()
                on_release: app.setSFTP_path()
<ConfirmScreen>:
    BoxLayout:
        orientation: 'vertical'
        Label:
            size_hint_y: None
            height: self.parent.height * 0.12
            text: 'Email connection: '+ app.labelTextEmailStatus
        Label:
            size_hint_y: None
            height: self.parent.height * 0.12
            text: 'FTP connection: '+ app.labelTextFTPStatus
        Label:
            size_hint_y: None
            height: self.parent.height * 0.12
```

```
            text: 'SFTP connection: '+ app.labelTextSFTPStatus
        BoxLayout:
            orientation: 'vertical'
            Label:
                size_hint_y: None
                height: self.parent.height * 0.2
                text: app.labelTextHelp
                font_size:'14sp'
            BoxLayout:
                orientation: 'horizontal'
                Button:
                    size_hint_y: None
                    height: self.parent.height * 0.160578
                    text: 'Credits'
                    on_release: root.manager.current = 'credits'
                Button:
                    size_hint_y: None
                    height: self.parent.height * 0.160578
                    text: 'OK'
                    on_release: root.manager.current = 'menu'
<CreditsScreen>:
    BoxLayout:
        orientation: 'vertical'
        Label:
            size_hint_y: None
            height: self.parent.height * 0.12
            text: '** Project B **'
            font_size:'25sp'
        Label:
            size_hint_y: None
            height: self.parent.height * 0.12
            text: 'Author: Michal Bojtos'
            font_size:'18sp'
        Label:
            size_hint_y: None
            height: self.parent.height * 0.12
            text: 'Created: 2016/2017'
            font_size:'18sp'
        BoxLayout:
            orientation: 'vertical'
            Button:
                size_hint_y: None
                height: self.parent.height * 0.128
                text: 'back to results'
                on_release: root.manager.current = 'confirm'
<MapViewScreen>:
#:import sys sys
#:import MapSource mapview.MapSource
    MapView:
        lat: app.latF
        lon: app.lonF
        zoom: 10
        map_source: MapSource(sys.argv[1], attribution="") if len(sys.argv) > 1 else "osm"
        MapMarkerPopup:
            lat: app.latF
            lon: app.lonF
            popup_size: dp(330), dp(130)
            Bubble:
                BoxLayout:
                    orientation: "horizontal"
                    padding: "5dp"
                    Label:
                        text: app.labelTextFreeGeoIP
                        markup: True
        MapMarkerPopup:
            lat: app.latIpA
            lon: app.lonIpA
            popup_size: dp(330), dp(130)
            Bubble:
                BoxLayout:
                    orientation: "horizontal"
                    padding: "5dp"
                    Label:
                        text: app.labelTextIpApi
                        markup: True
        MapMarkerPopup:
            lat: app.latIpI
            lon: app.lonIpI
            popup_size: dp(330), dp(130)
```

```
            Bubble:
                BoxLayout:
                    orientation: "horizontal"
                    padding: "5dp"
                    Label:
                        text: app.labelTextIpInfo
                        markup: True
        BoxLayout:
            orientation: 'horizontal'
            Button:
                size_hint_y: None
                size_hint_x: None
                height: self.parent.height * 1.2
                width: self.parent.width *2.8
                text: 'Text version'
                on_release: root.manager.current = 'database'
            Button:
                size_hint_y: None
                size_hint_x: None
                height: self.parent.height * 1.2
                width: self.parent.width *2.8
                text: 'Back to menu'
                on_release: root.manager.current = 'menu'
""")
```

B OBSAH PŘILOŽENÉHO CD

- / ..... kořenový adresář přiloženého CD
  - BP-xbojto00.pdf ..... bakalářská práce
  - ProjektB.apk ..... výsledná aplikace
  - zdrojove kody ..... zdrojové kódy aplikace
    - buildozer.spec
    - main.py
  - obrazky ..... použité obrázky
    - screenshot confirm.png
    - screenshot map.png
    - screenshot menu.png
    - screenshot privatni.png
    - screenshot settings.png
    - screenshot smtp.png
    - screenshot text.png
    - screenshot ukazka-ftp.png
    - screenshot ukazka-sftp.png
    - screenshot vystup.png
    - ukazka grafickeho srovnani-FreeGeoIP.png
    - ukazka grafickeho srovnani-IP-API.png
    - ukazka grafickeho srovnani-IPINFO.png
  - diagramy ..... použité diagramy
    - diagram-adresy.pdf
    - diagram-FTP.pdf
    - diagram-letiste.pdf
    - diagram-SFTP.pdf
    - diagram-SMTP.pdf
    - diagram-rir.pdf
    - diagram-verejna-IP.pdf
    - diagram-vyvojovy.pdf