Jihočeská univerzita v Českých Budějovicích

Přírodovědecká fakulta

Tvorba analytického nástroje ke zjišťování vazeb pro potřeby forenzních analýz ICT

Diplomová práce

Bc. Jan Houška

Vedoucí práce: Ing. Jaroslav Kothánek, Ph.D.

České Budějovice 2015

Jihočeská univerzita v Českých Budějovicích
Přírodovědecká fakulta

ZADÁVACÍ PROTOKOL MAGISTERSKÉ PRÁCE

Student: **B12432 / Bc. Jan Houška**
(jméno, příjmení, tituly)

Obor – zaměření studia: **1802T001 / Aplikovaná informatika**

Katedra: **Ústav aplikované informatiky**

Školitel: **Ing. Jaroslav Kothánek, Ph.D., jkothanek@prf.jcu.cz**
(jméno, příjmení, tituly, u externího š. název a adresa pracoviště, telefon, fax, e-mail)

Garant z PřF:
(jméno, příjmení, tituly, katedra – jen v případě externího školitele)

Školitel – specialista, konzultant: Ing. Jaroslav Kothánek, Ph.D.
(jméno, příjmení, tituly, u externího š. název a adresa pracoviště, telefon, fax, e-mail)

Téma magisterské práce:

Tvorba analytického nástroje ke zjišťování vazeb pro potřeby forenzních analýz ICT

Cíle práce :

Návrh a tvorba analytického softwarového prostředku pro analýzu komunikace a vazeb na základě analýzy informací získaných z výsledků forenzních analýz z oblasti ICT

Úkoly :

- Proveďte analýzu výstupů nejrozšířenějších forenzních nástrojů (Mobil Edit, Oxygen Forensic Suite, FTK Forensic Toolkit atd.)
- Identifikujte klíčové informace, které je možno využít pro určení vazeb mezi jednotlivými typy techniky (mobilní telefony, výpočetní technika více osob)
- Navrhněte interface načtení klíčových informací komunikaci osob a dalších vazeb mezi zkoumanou technikou (mobilní telekomunikační technika, výpočetní technika)
- Navrhněte aplikaci, která bude analyzovat vazby mezi jednotlivými výsledky zkoumání ICT techniky (např. komunikace osob, společné kontakty, apod.) a tyto zpracujte do písemného a grafického výstupu
- Na základě získaných informací předchozích bodů získejte dostupné informace k osobám z otevřených zdrojů světové sítě Internet a dle těchto zjištění vytvořte další možné vazby mezi zkoumanými subjekty. V rámci této analýzy určete váhy možné

pravdivosti zjištěných údajů

- Vytvořte aplikaci, která bude provádět analýzu vazeb dle shora uvedených bodů
- Proveďte ověření svých výsledků a stanovte případná rizika těchto analýz

Základní doporučená literatura :

Morrissey, S., iOS Forensic Analysis for iPhone, Ipad and iPod touch, Apress

Prenner, M., Analýza historie komunikace softwarového prostředku ICQ

Sammes, T.,, Jenkinson, B.,, Forensic Computing a Practitioner´s Guide, Springer

Carrier, B., File System Forensic Analysis

Odkazy:

www.accessdata.com

www.belkasoft.com

www.mobiledit.com

www.Oxygen-forensic.com

Financování práce :.....

Vedoucí práce :.....Ing. Jaroslav Kothánek, Ph.D.....podpis : .....

U externích vedoucích fakultní garant práce.....podpis : .....

Garant oboru mag. studia ..... podpis : .....

Vedoucí katedry .....RNDr. Libor Dostálek..... podpis .....

Případný souhlas vedoucího ústavu AV .....podpis : .....

V Českých Budějovicích dne 27. 8. 2013

Převzal/a dne..... 29. 8. 2013 ..... podpis : .....

Bibliografické údaje

Houška, J., 2015: Tvorba analytického nástroje ke zjišťování vazeb pro potřeby forenzních analýz ICT. [Development of analytical tool for relation detection required in digital forensics. Mgr. Thesis, in Czech.] – 79 p., Faculty of Science, The University of South Bohemia, České Budějovice, Czech Republic.

Anotace

Cílem této práce je návrh a implementace aplikace, která bude na základě výstupů z vybraných forenzních nástrojů analyzovat a hledat vazby mezi jednotlivými účastníky komunikace. Práce nejprve popíše výstupy vybraných programů používaných pro digitální forenzní analýzu a možnosti jejich použití pro import do vyvíjené aplikace. Následující kapitoly budou věnovány popisu celého vývojového cyklu aplikace.

Hlavním výstupem práce bude hotová aplikace splňující zadání a umožňující nejen hledání vazeb na základě výstupů z forenzních nástrojů, ale i hledání dalších možných vazeb z otevřených zdrojů.

Anotation

The objective of this thesis is to design and implement an application, which will on the basis of outputs from selected forensic tools analyse and search for relations among individual participants in communication. The paper will first describe procedures of digital forensics and selected programs used for digital forensics. Following chapters will be dedicated to description of the whole development cycle of the application.

The main outcome of the thesis will be a finished application meeting the requirements of the assignment and enabling not only search for relations based on outputs from forensic tools, but also search for additional possible relations from open sources.

Prohlašuji, že svoji diplomovou práci jsem vypracoval samostatně pouze s použitím pramenů a literatury uvedených v seznamu citované literatury.

Prohlašuji, že v souladu s § 47b zákona č. 111/1998 Sb. v platném znění souhlasím se zveřejněním své diplomové práce, a to v nezkrácené podobě elektronickou cestou ve veřejně přístupné části databáze STAG provozované Jihočeskou univerzitou v Českých Budějovicích na jejích internetových stránkách, a to se zachováním mého autorského práva k odevzdanému textu této kvalifikační práce. Souhlasím dále s tím, aby toutéž elektronickou cestou byly v souladu s uvedeným ustanovením zákona č. 111/1998 Sb. zveřejněny posudky školitele a oponentů práce i záznam o průběhu a výsledku obhajoby kvalifikační práce. Rovněž souhlasím s porovnáním textu mé kvalifikační práce s databází kvalifikačních prací Theses.cz provozovanou Národním registrem vysokoškolských kvalifikačních prací a systémem na odhalování plagiátů.

Český Krumlov, 10.12. 2015

Bc. Jan Houška

Poděkování
Zde bych rád poděkoval panu Ing. Jaroslavu Kothánkovi, Ph.D. za odborné vedení, ochotu a konzultace při realizaci této práce a své rodině za podporu během celého studia.

1. Úvod

Digitální forenzní analýza patří do rozsáhlé skupiny forenzních věd, které se aplikují při vyšetřování a dokazování trestných činů. K zajištění důkazů z informačních a komunikačních technologií existuje řada specializovaných nástrojů, které jsou ve většině případů úzce zaměřeny na konkrétní komunikační technologii nebo zařízení. V praxi se tedy lze setkat s případy, kdy část komunikace je specializovaným prostředkem získána z mobilního telefonu, zatímco k zajištění informací z osobního počítače byl použit nástroj jiný, specializovaný na zkoumání dat na pevných discích a jiných používaných záznamových médiích. Rekonstrukce a zkoumání takto zajištěných částí komunikace a vazeb mezi účastníky pak představuje problém nebo časovou ztrátu. Cílem této práce je tedy návrh a implementace aplikace, která by nejen umožňovala agregaci informací získaných pomocí různých forenzních nástrojů a následně jejich zkoumání, ale aby dále sloužila k hledání dalších informací o možných vazbách mezi účastníky zajišťované komunikace pomocí volně dostupných informací z internetu.

1.1 Cíle práce

Prvotním cílem práce je návrh a tvorba softwarového prostředku pro analýzu komunikace a vazeb na základě analýzy informací získaných z výsledků forenzních analýz z oblasti ICT. Aplikace bude umožňovat ucelený pohled na vazby a komunikaci mezi jednotlivými účastníky, která může probíhat pomocí více typů komunikačních technologií.

Pro dosažení prvotního cíle bude nutné provést analýzu výstupů z nejrozšířenějších nástrojů forenzní analýzy. Následně bude nutné určit, které informace je možné využít pro určení vazeb mezi jednotlivými osobami a dále analyzovat jejich komunikaci.

Druhotným cílem práce je návrh a implementace funkcí aplikace, které by umožnily vyhledat další možná spojení nejen mezi účastníky komunikace, ale i dalšími osobami, které mohou mít vazby na jednotlivé účastníky.

1.2 Rozvržení práce

Práce je členěna do následujících kapitol:

- **Teorie:** Stručně seznamuje s digitální forenzní analýzou a používanými komunikačními prostředky.
- **Analýza:** Zabývá se nástroji používanými k získávání informací a jejich výstupy.
- **Návrh:** Obsahuje návrh aplikace.
- **Implementace:** Popisuje konkrétní realizaci aplikace. Zaměřuje se na nejdůležitější části implementace.
- **Závěr:** Zhodnocuje přínos aplikace a dosažení cílů.

2. Teorie

2.1 Digitální forenzní analýza

Forenzní analýza digitálních dat nebo také Digitální forenzní analýza (DFA) patří do široké skupiny forenzních věd, tedy vědních oborů, které jsou aplikovány při vyšetřování a dokazování trestných činů. V praxi se jedná o aplikaci standardních vědních oborů (např. soudní psychologie) nebo aplikaci samostatných forenzních disciplín (např. daktyloskopie).[1]

Michale A. Coloaynnides [2] definoval digitální forenzní analýzu jako soubor technik a nástrojů, které jsou používány pro vyhledání důkazů v uživatelově počítači a které mohou být následně použity v jeho neprospěch. Tyto techniky podléhají přísným pravidlům, aby informace získané jejich pomocí bylo možné použít jako důkazní materiál. Důkazy získané pomocí digitální forenzní analýzy nemusejí přímo souviset s počítačovou kriminalitou.

Nejčastěji jsou tyto informace získány z osobních počítačů, mobilních zařízení a záznamových médií. Při získávání těchto informací se lze setkat s platnými daty (soubory), neplatnými daty (smazanými soubory) a daty v podobě fragmentů (částečně přepsanými soubory).

2.1.1 Logická a fyzická data

V předešlé kapitole bylo zmíněno, že při zkoumání obsahu disku, záznamového média, nebo paměti mobilního zařízení jsou data získávána z platných dat, neplatných dat a fragmentů. V následující kapitole jsou tyto pojmy vysvětleny.

- **Platná data** – Jsou platné soubory zapsané v souborovém systému a lze k nim přistupovat i bez specializovaných forenzních nástrojů. V takovémto případě hovoříme o *logické analýze* respektive o *logické extrakci*.
- **Neplatná data** - Jsou soubory, které jsou smazané, respektive v souborovém systému označené jako smazané. Nicméně se na disku stále fyzicky nacházejí a k jejich získání a následnému zkoumání už je potřeba forenzních nástrojů.

- **Fragmenty –** Jsou neplatná data, která již na disku byla částečně přepsána platnými daty. K jejich získání a případnému zobrazení dostupných informací je taktéž nutné použití specializovaných forenzních nástrojů. V případě neplatných dat a fragmentů mluvíme o *fyzické analýze* respektive o *fyzické extrakci*.

2.1.2 Forenzní analýza mobilních zařízení

Postup forenzní analýzy mobilních zařízení se v některých ohledech liší od analýzy osobních počítačů. Asi největším problémem je vysoká rozdílnost zkoumaných zařízení. Zatímco v případě osobních počítačů jsou rozdíly v zařízeních a jejich úložištích minimální, v případě mobilních zařízení existují značné rozdíly - přítomnost či nepřítomnost operačního systému, velké rozdíly jsou i v samotných používaných operačních systémech i jejich verzích.

Největším limitem je samotný fakt, že v případě mobilních zařízení nelze paměť fyzicky vyjmout a provádět postup jako v případě osobních počítačů. Proto se pro zajišťování důkazů z mobilního zařízení zpravidla používají specializované softwarové nástroje, které určitým způsobem spolupracují s operačním systémem daného zařízení, popřípadě kombinace hardwarového zařízení a softwarového prostředku (např. UFED viz. kapitola Forenzní nástroje).

V případě zajišťování důkazů je postup takový, že se zapnutý telefon nebo tablet připojí k forenzní stanici a pomocí spuštěného forenzního nástroje lze buď požadovaná data „vytáhnout“ nebo lze vytvořit bitovou kopii.

Samotný postup je dán výběrem forenzního nástroje. Vytvoření bitové kopie a následnou fyzickou analýzu (viz. kapitola Logická a fyzická analýza) z vybraných forenzních nástrojů umožňuje UFED, nicméně není dostupný pro všechny typy zařízení.

3. Analýza

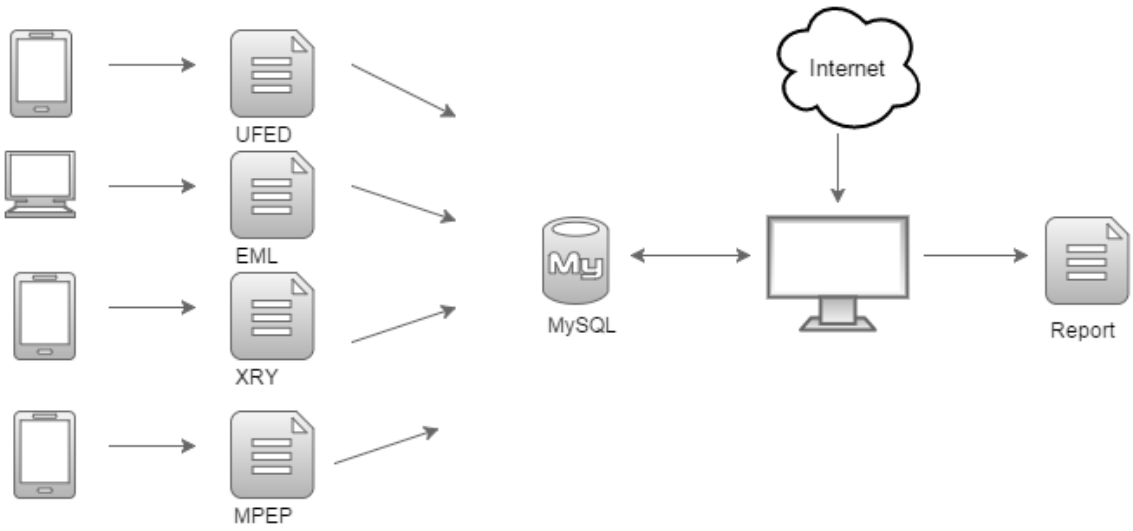
Tato kapitola popisuje řešený problém a vybrané komunikační prostředky, definuje pojem vazba a následně popisuje vybrané forenzní nástroje.

3.1 Řešený problém

Soudní znalci používají pro získávání informací o komunikaci řadu specializovaných nástrojů, které slouží k získání dat z prověřované techniky. V praxi se nejčastěji lze setkat s případem, kdy se k zajištění komunikace nebo obecně důkazů, používá nejvhodnější prostředek v závislosti na zajišťované technice.

Pro zajištění důkazů z osobních počítačů se často používá nástroj FTK Forensic Toolkit. Zatímco k zajištění důkazního materiálu z mobilních zařízení se nejčastěji používají nástroje přímo určené k forenzní analýze smartphonů nebo tabletů.

Toto používání rozdílných nástrojů s rozdílným formátem získaných informací i rozdílných typů souborů způsobuje komplikace při kompletaci a analýze celé komunikace. Z tohoto problému tedy vznikl požadavek na vytvoření aplikace, která by, nejen soudním znalcům, umožňovala ucelený pohled na získaná data o komunikaci osob, které používají několik typů komunikace popřípadě několik účtů. Další funkcí by pak měla být možnost nalézt další vazby na internetu. Viz. obrázek 3.1.



Obrázek 3.1. Řešení problému více rozdílných reportů

3.2 Vybrané komunikační prostředky

Tato práce se zabývá komunikací, která byla zajištěna pomocí forenzní analýzy mobilních zařízení – mobilních telefonů, smartphonů a tabletů. Dále pak e-mailovou komunikací, která mohla být zajištěna pomocí forenzní analýzy osobních počítačů.

3.3 Definice vazeb mezi kontakty

Pro potřeby této práce a aplikace budou mezi osobami A a B definovány tři možné typy vazeb:

- Existuje záznam o komunikaci mezi osobami A a B.
- Existuje záznam o osobě A v adresáři kontaktů osoby B.
- Existuje osoba C, která komunikovala nebo má v adresáři kontaktů osoby A a B.

3.4 Data potřebná k vytvoření vazeb

Pro vytvoření vazeb budou potřebná data, která budou získána pomocí forenzních nástrojů ze zkoumaných mobilních zařízení. Dle předešlé kapitoly a praxe forenzního zkoumání budou potřeba následující data:

- **Uložené kontakty** – záznamy z adresáře tj. jméno kontaktu, telefonní číslo popřípadě emailová adresa.
- **Záznamy o hovorech** – záznamy o proběhnutých telefonních hovorech, včetně informací o délce trvání, zda-li byl hovor příchozí nebo odchozí a čase, kdy samotný hovor proběhl.
- **Textové zprávy** – uložené textové zprávy nebo emaily a informace o čase doručení nebo odeslání, jejich příjemci a adresátovi a dále pak text samotné zprávy.

3.5 Forenzní nástroje

V této kapitole si autor klade za cíl stručně popsat vybrané forenzní nástroje, které se používají pro digitální forenzní analýzu mobilních zařízení. Zde autor považuje za nutné upozornit na fakt, že ačkoliv v zadání diplomové práce jsou zmíněny jako příklady dva programy, kterými by se mohl autor zabývat (Mobil Edit a Oxygen

Forensic Suite), byly pro účely diplomové práce, po konzultaci s vedoucím práce, vybrány programy jiné. Důvodem pro změnu v případě Oxygen Forensic Suite byla nedostupnost tohoto prostředku zapříčiněná změnou licenčních podmínek. Autoři už neposkytují bezplatné zkušební licence. Program Mobil Edit byl vynechán proto, že autorem zvolené programy tento nástroj v několika ohledech převyšují.

3.5.1 UFED

Zařízení od firmy Cellebrite, UFED Touch [6] v kombinaci s programem UFED Physical Analyzer je v současné době špičkou mezi nástroji pro digitální forenzní analýzu mobilních zařízení.

UFED Touch je hardwarové zařízení běžící na systému Windows XP, které se používá pro extrakci dat z připojeného zařízení. K dispozici je logická i fyzická extrakce dat, a to v závislosti na podpoře připojeného zařízení. V současné době je schopen získávat data fyzickou extrakcí z více jak 4 300 typů zařízení a logická data z více jak 6 800 zařízení. V neposlední řadě je také dobré zmínit českou lokalizaci nástroje.

K samotné extrakci je nutné připojení telefonu dle pokynů samotného UFEDu, a to v závislosti na typu připojovaného zařízení. Součástí balení je i rozsáhlá sada datových kabelů k připojení mobilních zařízení podporovaných typů a speciální SIM karty, které slouží ke klonování zajišťovaných SIM karet. Výstupní soubor ze zařízení musí být uložen na připojený externí disk nebo flashdisk, jelikož zařízení neumožňuje ukládat data na vestavěný disk.

Získaný soubor je následně analyzován pomocí programu UFED Physical Analyzer. Ten umožňuje získat velké množství informací ze získaného výstupního souboru respektive telefonu. Mezi nejdůležitější okruhy informací patří:

- Informace o zařízení včetně času aktivace telefonu a IMSI,
- Informace o bezdrátových sítích,
- Historie webu, hledané položky a cookies,
- Hesla,
- Kontakty,
- Záznamy o hovorech,
- SMS a MMS zprávy, popř. chaty,

- Kalendáře,
- Údaje z GPS přijímače o poloze zařízení.

Všechny získané informace je pak možné exportovat do zprávy (reportu), která může být v několika formátech a obsahovat buď všechny, nebo jen zvolené okruhy informací. O samotném výstupu pojednává kapitola Výstupy forenzních nástrojů.

Dle subjektivního názoru autora se jedná o nejlepší nástroj pro zkoumání mobilních zařízení z těch, které jsou zmíněny v této práci.

3.5.2 Mobile Phone Examiner Plus

Mobile Phone Examiner Plus [7] je program pro digitální forenzní analýzu mobilních telefonů od společnosti AccessData, tedy od stejné, která je autorem i výše zmíněného FTK Forensic Tool. Od UFEDu se liší tím, že je tvořen pouze samostatným programem, který je použit pro extrakci dat a jejich následné analýze.

Z extrahovaných informací k nejdůležitějším patří:

- Kontakty,
- Historie volání,
- SMS a MMS zprávy,
- Historie prohlížečů se záložkami a cookies,
- Poznámky,
- Údaje z GPS přijímače o poloze zařízení.

Následně je opět možné vytvoření závěrečné zprávy (reportu) se zvolenými informacemi. Samotná závěrečná zpráva je, dle názoru autora práce, méně přehledná než v případě UFED. V případě formátů HTML, XLS a XLSX je orientace v závěrečné zprávě značně komplikovaná. Více o tomto problému v kapitole Výstupy forenzních nástrojů.

3.5.3 XRY

Posledním vybraným forenzním nástrojem pro digitální forenzní analýzu mobilních zařízení, který je v rámci této práce použit a jehož výstupy budou moci být

později importovány do reportu, je softwarový nástroj XRY od firmy MSAB [8]. Program XRY je k dispozici ve třech verzích - XRY Logical, XRY Physical a XRY Complette.

XRY Logical je verze XRY, která umožňuje logickou extrakci z připojeného zařízení. XRY Physical je vyšší verze programu, která umožňuje fyzickou extrakci ze zařízení díky tomu, že určitým způsobem „obejde“ operační systém. Samotný proces fyzické extrakce je pak rozdělen do dvou po sobě jdoucích kroků. První krok autoři programu nazývají „dump“, kdy jsou zajištěna raw data. Ta jsou v druhém kroku „decode“ rekonstruována a jsou získána jak platná, tak neplatná data.

Třetí verzí XRY je verze XRY Complete, která je kombinací dvou předešlých. Tato kombinace nabízí možnost fyzické i logické extrakce a následné porovnání výsledků získaných oběma způsoby.

Získaná data jsou:

- Kontakty,
- SMS zprávy,
- E-maily,
- Mediální soubory,
- Dokumenty.

Tak jako u předešlých nástrojů i zde je možné vytvořit závěrečnou zprávu (report) a stejně tak je možné si zvolit obsah a formáty souborů, do kterých bude uložena. Více o závěrečné zprávě v kapitole Výstupy forenzních nástrojů.

3.6 Výstupy forenzních nástrojů

V této kapitole jsou popsány výstupy z forenzních nástrojů a jejich struktura, které byly popsány v kapitole Forenzní nástroje. Následně je zvolen formát výstupního souboru, pro který bude implementován import.

Při samotném výběru formátu autor zohledňuje strukturu získaného souboru a dále možnosti implementace, zejména z hlediska funkčnosti a dostupnosti bezplatných knihoven.

Pro získání dat byl použit mobilní telefon LG 440 L4 II. Zařízení běží na operačním systému Android ve verzi 4.1. Obsahovalo 827 kontaktů, 1010 SMS zpráv a 517 záznamů o hovorech.

3.6.1 Výstupy z UFED Physical Analyzer

Při vytváření reportu čili závěrečné zprávy je v případě programu UFED možné zvolit z několika formátů, do kterých bude závěrečná zpráva uložena. Zároveň je možné zvolit obsah. Popis struktury závěrečné zprávy je možné nalézt v příloze C.

Formáty výstupu

Výsledná závěrečná zpráva může být vytvořena ve více formátech. UFED nabízí následující formáty:

- PDF,
- DOCX,
- HTML,
- XML,
- XLS,
- XLSX.

Ať je závěrečná zpráva vytvořena v jakémkoliv formátu, struktura výše popsaná zůstává zachována. Výhodou oproti některým konkurenční nástrojům je poměrně dobrá přehlednost závěrečných zpráv. Ta je dána samotnou strukturou dokumentu a použitím tabulek. Zpráva má i obsah, díky kterému lze ve zprávě přejít na zvolenou kapitolu

a usnadňuje orientaci v ní. Bohužel, tato vlastnost není pro všechny nástroje samozřejmostí.

Výběr formátu pro vyvíjenou aplikaci

Při výběru formátu, který bude použit jako vstup pro import dat do databáze vyvíjené aplikace, vybral autor formát XLS. Ten byl vybrán díky struktuře, kdy jednotlivá kapitola z výše zmíněné struktury odpovídá jednomu listu v sešitě závěrečné zprávy. Díky tomu je zjednodušen výběr jednotlivých kapitol. Parsovány budou jednotlivé listy sešitu obsahující požadované informace.

Dalším důvodem, proč byl vybrán právě soubor XLS, je celkový rozsah závěrečné zprávy a tím i rychlost načítání. Například rozsah zprávy vytvořené ze zkušebního zařízení ve formátu PDF byl 1395 stran A4. I z tohoto důvodu se autor přiklonil k použití formátu XLS. V případě výstupní zprávy ve formátu XML by pak byl import komplikován faktem, že výstup nebyl validní XML soubor (nebyl by validní dle DTD).

3.6.2 Výstupy z Mobile Phone Examiner Plus

Při vytváření závěrečné zprávy je možné zvolit, jaké informace do ní budou zahrnuty. Následně je možné vybrat typy souborů, ve kterých bude závěrečná zpráva uložena. Popis struktury závěrečné zprávy je možné nalézt v příloze C.

Formáty výstupu

Nástroj Mobile Phone Analyzer Plus poskytuje možnost uložit závěrečnou zprávu do následujících formátů:

- PDF,
- HTML,
- RTF,
- XLSX,
- CSV,
- TXT,

- XML.

Zde je ovšem nutné poznamenat, že na rozdíl od UFED, je závěrečná zpráva exportována do výstupních souborů poněkud nešťastně. V případě HTML a PDF formátu neobsahuje závěrečná zpráva obsah s odkazy na konkrétní kapitoly. Vzhledem k jejímu rozsahu absence tohoto prvku značně komplikuje procházení a orientaci v souboru. Rozsah závěrečné zprávy ze zkušebního zařízení byl v případě PDF 989 stran A4.

Dalším problémem je rozložení informací ve formátu XLSX. Na rozdíl od UFED, kde každý list sešitu XLS a XLSX logicky odpovídá jedné kapitole závěrečné zprávy (např. kontakty), v případě MPEP tomu tak není. Zde je pro každou stranu, která odpovídá jedné straně A4 v PDF formátu, jeden list sešitu. V praxi to pak znamená, že výsledný XLSX soubor ze zkušebního zařízení měl 989 listů. Ty nejsou nijak pojmenovány (sešit1 až sešit 898) a orientace v souboru je pak značně problematická. Co se týká souborů XML, zde se opakuje problém se strukturou, výstupní soubor není validní.

Výběr formátu pro vyvíjenou aplikaci

Z výše zmíněných důvodů byl pro import do vyvíjené aplikace zvolen formát CSV. MPEP vytváří pro každou kapitolu závěrečné zprávy samostatný soubor s obsahem příslušným dané kapitole. Pro tytu účely byly vybrány soubory:

- Call History.csv - historie volání,
- Phonebook.csv – seznam kontaktů,
- Sms.csv – SMS zprávy.

3.6.3 Výstupy XRY

Stejně jako předešlé nástroje i XRY umožňuje zvolit obsah závěrečné zprávy z několika kapitol a následně zvolit také výstupní formát. Popis struktury závěrečné zprávy je možné nalézt v příloze C.

Formáty výstupu

Program XRY umožňuje uživateli uložení závěrečné zprávy do následujících formátů:

- XLSX,
- DOCX,
- XML,
- PDF.

Výběr formátu pro vyvíjenou aplikaci

Pro import z XRY byl vybrán formát XML, ačkoliv samotný formát souboru XML je zvolen trochu nestandardně – je složen z prázdných elementů. Kořenový element *views* obsahuje jednotlivé elementy *view*, jejichž atributem *name* je pak jméno jednotlivé kapitoly ze závěrečné zprávy.

Příkladem je kapitola *Contacts/Kontakty*, kde element *view* obsahuje jednotlivé elementy *item*. Element *item* reprezentuje jednotlivý záznam v kontaktech. Každý element *item* obsahuje pět elementů se stejným jménem *field*, jehož atributem *name* je *Jméno* a atributem *value* je název samotného kontaktu. Následuje příklad záznamu ze seznamu kontaktů na *kontakt „Záchranka“*.

```
<item>
      <field name="Related Application" value="com.android.contact.
      sim" class="INFO"/>
      <field name="Jméno" value="ZACHRANKA" class="CONTACT"/>
      <field name="Mobil" value="155" class="PHONE"/>
      <field name="Index" value="45" class="INFO"/>
      <field name="Název účtu" value="SIM" class="INFO"/>
</item>
```

3.7 E-mailová komunikace

Pro import e-mailových zpráv byl vybrány formát EML. Tento formát je používaný zejména v Microsoft Outlook Express nebo Mozzila Thunderbird. Důvodem, proč byl vybrán právě tento formát je to, že jsou dostupné volně šiřitelné programy, které umožňují export z jiných formátů (např. PST nebo DBX) právě do formátu EML.

3.8 Požadavky na aplikaci

V této kapitole jsou popsány požadavky na aplikaci, které musí být splněny při vývoji.

3.8.1 Funkční požadavky

Funkční požadavky definují funkcionalitu vyvíjené aplikace.

- Aplikace umožňuje import výstupů (souborů) z vybraných forenzních nástrojů.
- Aplikace umožňuje členění na více případů.
- Aplikace umožňuje členění na jednotlivé osoby (profily).
- K jednotlivým profilům je možné přidat více účtů (e-mail nebo telefonní číslo).
- K jednotlivému účtu náleží seznam kontaktů.
- Aplikace umožňuje prohlížet zajištěnou komunikaci.
- Aplikace umožňuje nalezení vazeb mezi vybranými osobami.
- Aplikace umožňuje nalezení dalších relevantních informací na internetu.
- Aplikace umožňuje grafický a textový výstup vazeb.

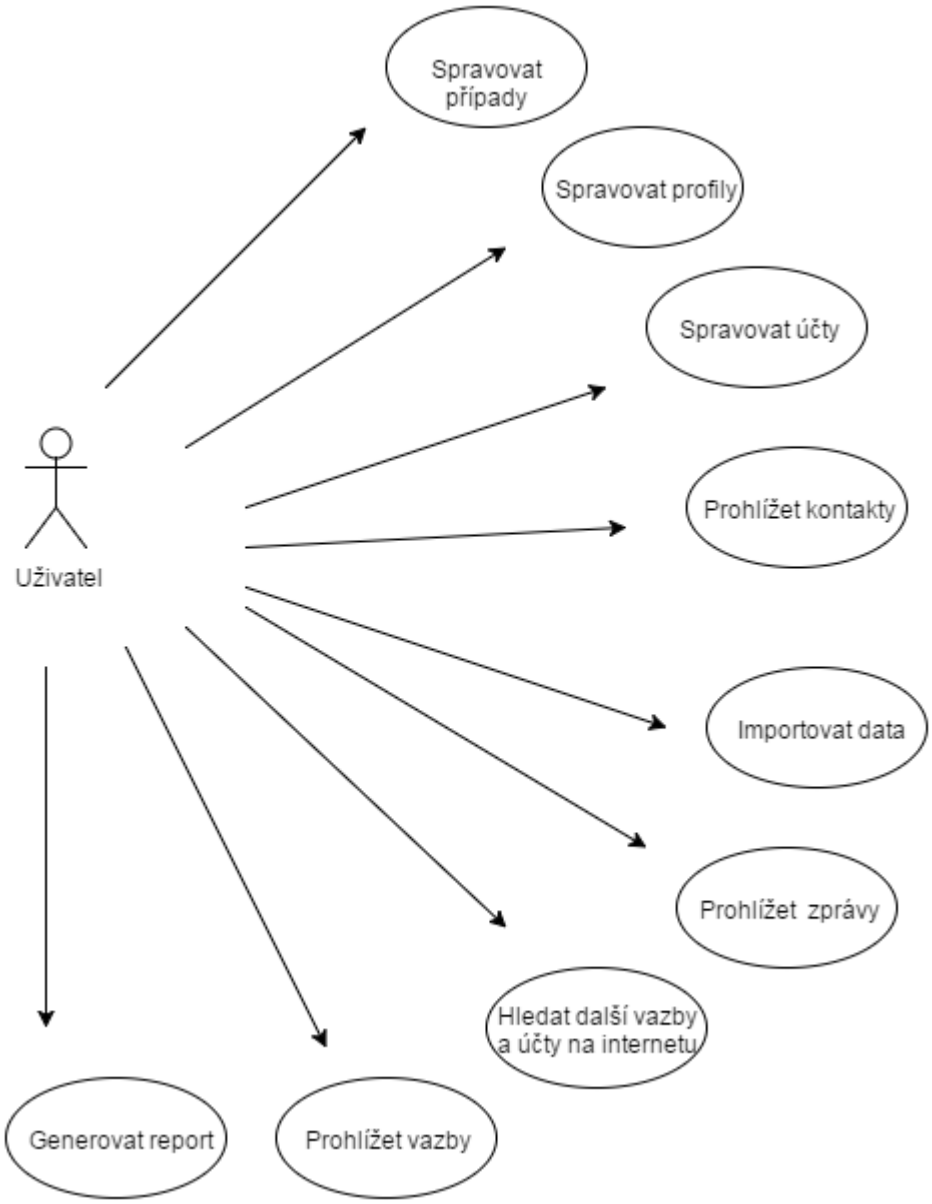
3.8.2 Nefunkční požadavky

Nefunkční požadavky reflektují prostředí, kde bude aplikace nasazena a definuje technologie, které budou použité během implementace.

- Aplikace běží na operačním systému Windows.
- Data jsou uchována v relační databázi.
- Aplikace splňuje základní požadavky na spolehlivost a odezvu.
- Zdrojový kód obsahuje dostatek komentářů.
- Použití aplikace nebude vyžadovat finanční prostředky pro splnění licenčních podmínek.

3.8.3 Případy užití

Na základě funkčních požadavků zmíněných v minulé kapitole vytvořil autor diagram použití.



Obrázek 3.2, Use Case diagram

4. Návrh

Cílem této kapitoly je popsat, jak bude aplikace fungovat a co by měla uživateli umožňovat. Dále budou popsány použité technologie a návrh databáze a struktura samotné aplikace.

4.1 Funkce aplikace

V této kapitole si autor klade za cíl popsat návrh samotného fungování aplikace, tedy reflektovat funkční požadavky aplikace. Funkce aplikace budou rozděleny do několika pomyslných bloků:

- **Správa případů** – Aplikace bude umožňovat pracovat na několika případech. Je tedy nutné, aby uživatel mohl přidávat, modifikovat a mazat případy.
- **Import dat** – Aplikace bude umožňovat import dat z výstupních souborů vybraných forenzních nástrojů.
- **Zobrazení a správa kontaktů a komunikací** - Aplikace bude uživateli umožňovat prohlížení kontaktů jednotlivých účtů, příslušných zpráv a volání.
- **Vizualizace vazeb** – Aplikace bude uživateli umožňovat přehledně zobrazit vazby mezi vybranými osobami.
- **Hledání dalších informací z otevřených zdrojů** – Aplikace bude umožňovat uživateli vyhledat další možné kontakty nebo účty vybrané osoby.

4.2 Zobrazení kontaktů a komunikací

Tato kapitola slovně popisuje funkcionalitu a konceptuální model, který je v následující kapitole.

Aplikace bude umožňovat uživateli správu osob – **Profilů**. Každý profil nese atribut *jméno* uživatele a dále jsou mu přiřazeny účty. Pro rozlišení mezi profily osob, které budou pro zkoumání důležité a těmi, které figurují pouze jako kontakt, bude profil obsahovat *boolean* atribut *main.* V praxi tedy aplikace bude pracovat s *hlavními profily* a „běžnými“ profily.

Účet reprezentuje komunikační účty uživatele. V praxi se bude jednat o telefonní čísla nebo e-mailové účty a každý uživatel může mít přiřazeno více účtů.

Rating účtu pak nabývá hodnot od 1 do 100 a při používání aplikace se bude jednat o hodnotu, která určuje pomyslnou míru příslušnosti, se kterou účet náleží k danému profilu. Tato hodnota je důležitá zejména pro případy, kdy budou účty přiřazovány pomocí funkce, která bude umožňovat hledání z otevřených zdrojů. Při importu dat z ověřených souborů – importu z výstupů forenzních nástrojů bude automaticky nastavena na nejvyšší hodnotu. V případě získání účtu z otevřených zdrojů bude maximální hodnota o jeden bod nižší, aby byl jasně viditelný rozdíl mezi účtem z ověřeného zdroje a účtem nalezeným na internetu. Dále bude účet obsahovat informaci o svém **typu,** aby bylo možné jasně rozlišit, zda se jedná o účet e-mailu, nebo telefonu.

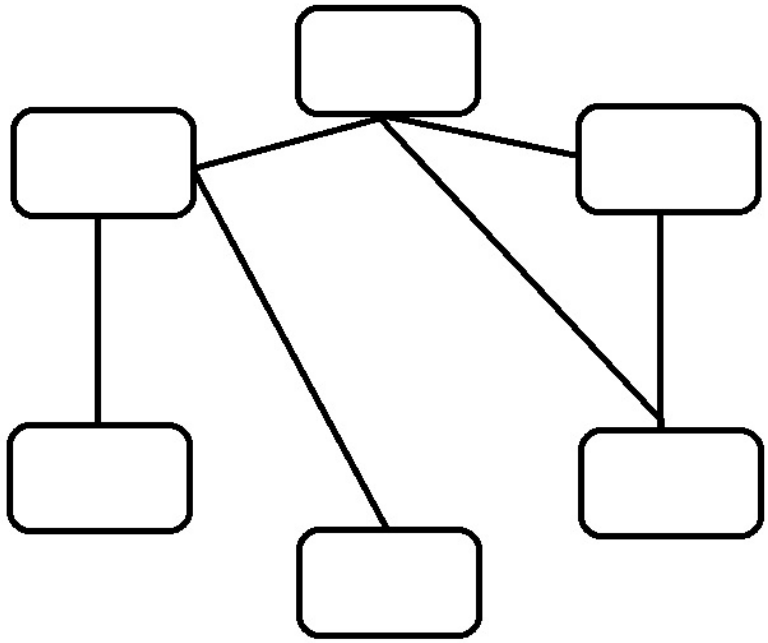
Každý účet bude mít **seznam kontaktů**, který bude importován z výstupních souborů forenzních nástrojů nebo přidáním možného nalezeného kontaktu z internetu. Z hlediska modelu bude kontakt profilem, jehož hodnota *main* bude *false*. Následně bude kontaktu přiřazen účet, který bude opět reprezentovat jeho e-mail nebo telefonní číslo.

Z každého účtu pak probíhá komunikace – **zprávy**. V případě aplikace se jedná o hovory, textové zprávy a e-maily. Každá zpráva má více atributů – *účet odesílatele*, *účet příjemce*, *čas*, *trvání* v případě hovoru, *předmět v* případě emailu a samotný *text* zprávy.

4.3 Návrh vizualizace vazeb

Tato kapitola popisuje koncept grafického zobrazení vazeb, který bude implementován do grafického rozhraní vyvíjené aplikace. Vazby byly definovány v kapitole Definice vazeb mezi kontakty. Samotný návrh a následná implementace zohledňuje požadavek na přehlednost, a to i pro vyšší počet vizualizovaných profilů osob.

Na následujícím obrázku je pomocí obdélníků znázorněno šest osob (profilů osob). Pro dosažení přehlednosti budou obdélníky vykreslovány do kruhu. Spojnice mezi obdélníky znázorňují vazby mezi osobami.



Obrázek 4.1, Návrh funkce Vizualizace vazeb

4.4 Návrh hledání dalších informací z otevřených zdrojů

Tato kapitola popisuje návrh implementace funkce, která bude aplikaci umožňovat hledání dalších možných vazeb a účtů, které mohou souviset s osobou, jejíž komunikace byla předtím zajištěna.

4.4.1 Princip funkce pro hledání informací

K hledání souvislostí se autor dle zadání zaměřil na možnosti hledání z otevřených zdrojů sítě Internet. Princip takového hledání možných vazeb bude spočívat v nalezení webových stránek, které obsahují požadovaný výraz (e-mailovou adresu, telefonní číslo nebo jméno profilu). Následně budou URL těchto stránek uloženy do seznamu.

Na webových stránkách v tomto seznamu budou poté hledána další telefonní čísla a e-maily. Dle počtu stránek, které budou obsahovat dvojici požadovaného výrazu a nově nalezeného telefonního čísla nebo e-mailu, bude zhodnocena možná souvislost mezi požadovaným výrazem a nalezeným výrazem. K hodnocení souvislosti bude započítáno i to, z jakých webových stránek jednotlivé nalezené dvojice pocházejí.

4.4.2 Hodnocení nalezených informací

Dle zadání má autor navrhnout váhy možné pravdivosti nalezených údajů. V tomto smyslu bude pro nalezená telefonní čísla a e-maily (dále účet) vypočítán *rating*. Maximální hodnota ratingu pro účet nalezený pomocí této funkce bude 99. Hodnotou 100 budou ohodnoceny pouze účty, které byly importovány z ověřených zdrojů tedy výstupů z forenzních nástrojů. Hodnota je závislá na počtu stránek, kde byl účet nalezen, a dále na hodnocení stránky, na jaké byl nalezen. Tyto hodnoty jsou uloženy v textovém souboru *sources.txt* a umožňují uživateli přidávání vlastních URL a jejich hodnocení. Hodnocení v souboru může nabývat hodnot od jedné do deseti.

Autor při návrhu vychází z pokusů, které během návrhu proběhly, a zároveň ze zkušeností vedoucího práce, se kterým byla metoda hodnocení konzultována. *Rating* je

definován jako suma ratingů všech stránek, na kterých byl účet nalezen. Pro samotný účet je hodnota vypočítána dle následujícího vzorce:

$$\frac{rating}{100} = \sum \frac{1}{5} \cdot (\frac{r}{3{,}3} + 1)$$

Autor vychází z předpokladu, že je-li nalezený účet nalezen spolu s hledaným účtem na pěti rozdílných webových stránkách, je vysoká pravděpodobnost, že existuje souvislost mezi hledaným a nalezeným účtem, a proto bude hodnota ratingu nastavena na maximální hodnotu. Dále vychází z předpokladu, že je-li účet nalezený pouze na jedné stránce, jejíž hodnocení v souboru *sources.txt* má nejvyšší hodnotu (*r=10*), měla by se hodnota ratingu pohybovat okolo 80. Tato hodnota odpovídá nalezení účtu na čtyřech stránkách, k nimž nebylo dostupné hodnocení v souboru *sources.txt* (*r=0*). Z tohoto důvodu byl ve vzorci zvolen koeficient 3,3.

Příkladem budiž e-mail, který byl nalezen na dvou webových stránkách. K jedné nebylo v souboru sources.txt žádné hodnocení (r=0), k druhé bylo hodnocení r=5. Vypočítaný rating pak odpovídá hodnotě 70 (v aplikaci je implementováno zaokrouhlení na celá čísla).

4.4.3 Výběr technologie

Pro vyhledání webových stránek s požadovaným výrazem nejprve autor zvažoval použití webového vyhledávače *Google*, respektive jeho Custom Search API pro implementaci vyhledávání v aplikaci. Nicméně možné použití tohoto API se neukázalo jako vhodné, a to hlavně díky licenčním omezením.

K použití je potřeba registrace a získání API Key, který je nutný pro použití vyhledávacích funkcí. V bezplatné verzi je pak API omezeno tak, že jednomu API Key vrací jen tisíc odkazů za dvacet čtyři hodin. V případě použití tohoto API by byl nucen uživatel aplikace vlastnit svůj API Key, nebo, v horším případě, by aplikace využívala jeden. V tom případě by se při použití několika uživateli počet vrácených odkazů sdílel. Toto řešení bylo tedy vyhodnoceno jako zcela nepřijatelné.

Z výše zmíněných důvodů autor zvolil možnost využití českého vyhledávače *Seznam.cz*. V tomto případě nebude použito API, ale bude pracováno s webem pomocí knihovny *HTML Agility Pack*, která slouží pro práci respektive čtení zdrojového kódu

webu. Toto řešení v případě vyhledávače Google není možné, protože jeho zdrojový kód je generován skripty a proto není možné použít *HTML Agility Pack* pro vyhledání odkazů.

4.5 Výběr technologií

V této kapitole jsou stručně popsány softwarové nástroje a technologie, které byly použity pro návrh a implementaci. Byly vybrány na základě zkušeností autora práce získaných během řešení školních i mimoškolních projektů.

4.5.1 Výběr prostředí, programovacího jazyka a IDE

Microsoft .NET Framework

V této kapitole je ve stručnosti popsán .NET Framework a některé jeho základní vlastnosti. Více možné nalézt v [9] a [10].

Microsoft .NET Framework (součást platformy .NET) je komplexní a robustní prostředí pro vývoj a provozování aplikací, zejména na platformě Windows (existují ale i distribuce pro Mac OS X nebo UNIX). Umožňuje vyvíjet konzolové aplikace, aplikace s grafickým rozhraním (Windows Forms – WF a Windows Presentation Foundation - WPF), webové aplikace (ASP.NET) nebo servisně orientované aplikace (WCF).

Ve stručnosti se .NET Framework skládá z běhového systému Common Language Runtime (CLR) a knihoven systému .NET Framework.

Programovací jazyk C#

C# je vysokoúrovňový objektově orientovaný programovací jazyk. Byl vyvinut firmou Microsoft pro požadavky platformy .NET Framework. C# je založen na Javě a C++, ze kterého čerpá syntaxi. V současnosti se pravděpodobně jedná o nejpoužívanější jazyk v prostředí .NET [9].

Vývojové prostředí

Jako hlavní vývojové prostředí (IDE) autor práce zvolil **Visual Studio 2013** ve verzi Professional. Ta je dostupná od října 2013 spolu s .NET 4.5.1. Jedná se bezesporu

o jedno z nejpokročilejších dostupných vývojových prostředí, které obsahuje řadu nástrojů pro komplexní vývoj softwaru.

4.5.2 Výběr datového úložiště

Výběr RDBMS

Tato kapitola popisuje fázi výběru RDBMS (Relational Database Management System) pro vyvíjenou aplikaci. Jedná se o stručné srovnání možných řešení a shrnutí důvodů pro výběr vhodného RDBMS.

Při výběru databáze přicházely v úvahu dva možné RDBMS. Výběr závisel zejména na splnění nefunkčních požadavků. Dalším faktorem výběru byly licenční podmínky produktu.

Microsoft SQL Server

První variantou při výběru RDBMS byl produkt od firmy Microsoft – SQL Server 2012 ve verzi Express – tedy s licencí pro volné užití. Jedná se o komplexní a spolehlivý RDBMS, jehož výhodou je bezproblémová spolupráce s Visual Studiem. Verze Express je v některých ohledech limitována, zejména z hlediska výkonu.

MySQL

Druhým RDBMS, který se později ukázal jako optimální pro vývoj aplikace, je MySQL. Jedná se o open-source RDBMS, který byl původně vyvíjen švédskou firmou SQL AB, nyní je vlastněn a vyvíjen firmou Oracle. Dle statistiky [12] se k dubnu roku 2015 jedná o druhý nejrozšířenější RDBMS, hned po Oracle RDBMS. Výhodami MySQL jsou dostatečný a ničím nelimitovaný výkon a licenční podmínky. MySQL využívá dvojí licenční podmínky.

Příklady použití licence MySQL

Tato kapitola obsahuje citovaný text z [13], jehož cílem je objasnit možnosti použití databáze MySQL v závislosti na typu její licence.

„První podnikatel provozuje www stránky. Používá při tom databázi MySql a případně i další programy šířené pod licencí GPL. Tento podnikatel může za své služby vybírat peníze, aniž by porušil podmínky volné licence.

Druhý podnikatel nabízí svým zákazníkům instalaci databáze MySql a případně i další servis. Za své služby si nechává platit. Také tento podnikatel splňuje podmínky volné licence.

Třetí podnikatel vytvořil komerční program, který používá MySql. Databáze není součástí programu a každý zákazník si databázi musí sám sehnat a nainstalovat. Také tento případ je v souladu s volnou licencí.

Čtvrtý podnikatel také vytvořil komerční program. Protože však chtěl zákazníka ušetřit složitého instalování databáze, ovladače k databázi začlenil přímo do svého programu. Tohle řešení odporuje licenci GPL. Podnikatel si může buď dokoupit komerční licenci MySql, nebo může použít ovladače některé levné komerční databáze. Obvykle se rozhodne zákazníkovi vysvětlit, že databáze MySql je "moc drahá".“

Vybraný RDBMS

Z výše zmíněných důvodů a po konzultaci s vedoucím práce byl pro vyvíjenou aplikaci vybrán databázový systém MySQL, který splnil nefunkční požadavky, viz kapitola Nefunkční požadavky. Ten není limitován velikostí databáze ani počtem možných prostředků, zároveň jeho licenční podmínky nepředstavují překážku pro provoz aplikace.

Objektově relační mapování

Pro přístup k datům z databáze bylo použito objektově relační mapování (ORM). Při programování je vždy snaha co nejvěrněji zachytit realitu. V relační databázi je entita reprezentována jako řádek tabulky, v objektovém jazyce je entita reprezentována jako instance třídy. Tato odlišná reprezentace entit vedla ke vzniku programovací techniky, označované jako ORM (objektově relační mapování). Ta zajišťuje konverzi mezi relační databází a objekty, se kterými se pracuje v objektově orientovaném jazyce. Díky technice získáme možnost jednotného přístupu k libovolné entitě.

Použití ORM usnadňuje práci s daty (entitami), která jsou v aplikaci reprezentována jako objekty tím, že je do jisté míry odstraněna nutnost používání SQL

jazyka. Zejména pak u CRUD operací. Dále se ORM stará o automatickou konverzi rozdílných datových typů mezi databázovým systémem a programovacím jazykem.

Entity Framework

„Entity Framework (EF) is an object-relational mapper that enables .NET developers to work with relational data using domain-specific objects. It eliminates the need for most of the data-access code that developers usually need to write.“[16]

Možnosti použití Entity Frameworku

Model First

Tato technika umožňuje vytvoření databáze až na základě vytvořeného vizuálního modelu entit a jejich vztahů. Jsou vytvořeny požadované entity, přidány jejich atributy a zvoleny jejich datové typy. Dále jsou pak přidány relace (vztahy) mezi entitami.

Po vytvoření modelu je vygenerován SQL skript pro vytvoření databáze. Zde se projeví výhoda ORM, tedy nezávislost na konkrétní technologii databáze. Díky příslušnému connectoru je možné použití nejen MS SQL serveru, ale i například MySQL databázového serveru.

Code First

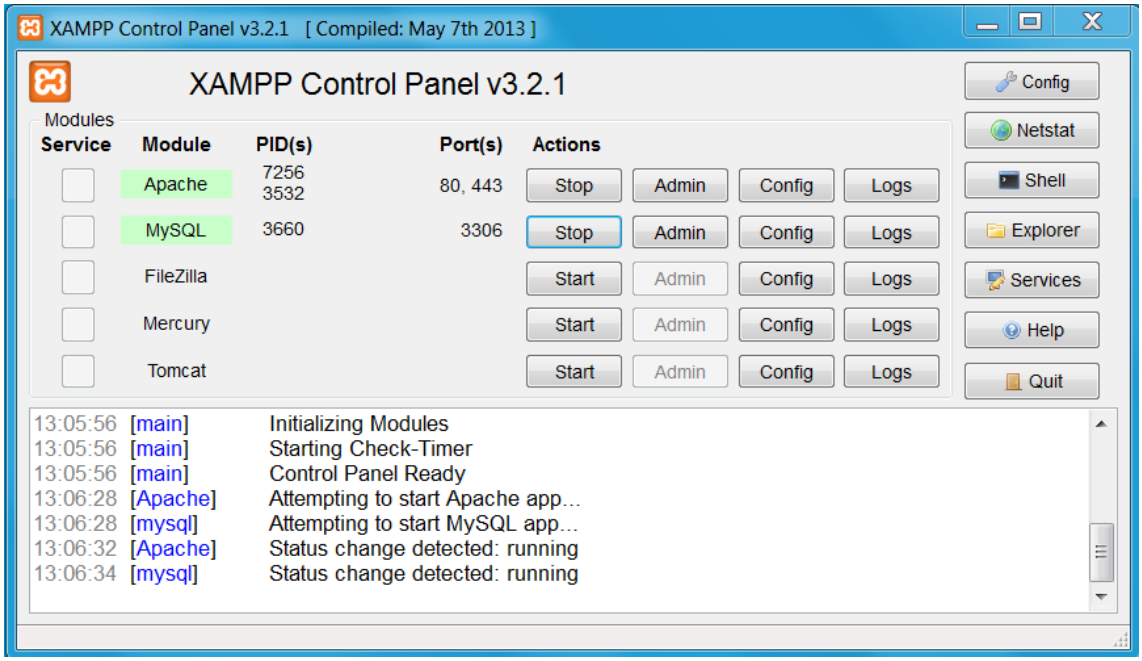
Tato technika je opakem první. Opět není potřeba databáze. Na rozdíl od „model first“ není vytvářen vizuální model databáze. Databáze je vytvořena až na základě vytvořených tříd a jejich instančních proměnných.

Database First

Tato technika se od předešlých dvou liší postupem. Databáze je již vytvořena a na jejím základě jsou díky Entity Frameworku vytvořeny třídy včetně instančních proměnných.

XAMPP

Pro běh MySQL serveru bylo zvoleno řešení od Apachefriends. Nástroj XAMPP Control Panel V 3.2.1 umožňuje uživateli snadnou instalaci Apache serveru na operačních systémech Windows (existují i verze pro ostatní platformy), na kterém je možné snadné nastavení a spuštění služeb PHP, phpMyAdmin, MySQL databáze, Filezilla FTP server, Mercury, Tomcat a Strawberry Perl.



Obrázek 4.2, Grafické rozhraní aplikace XAMMP

Pro potřeby vyvíjené aplikace je nutné spuštění MySQL serveru, nicméně pro samotnou konfiguraci přes phpMyAdmin je nutné i spuštění serveru Apache.

Potřebné knihovny a connector

Pro možnosti práce s MySQL databází a jejího připojení je nutná instalace connectoru a pluginu pro Visual Studio, který umožňuje práci s MySQL ve vývojovém prostředí.

- Connector/NET 6.9.5
- MySQL for Visual Studio 1.2.3

4.6 Návrh databáze

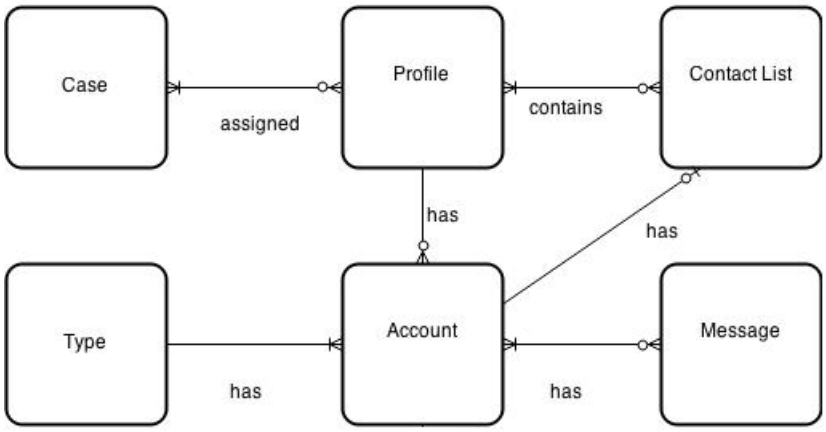
Kapitola popisuje proces návrhu modelu databáze.

4.6.1 Entitně-relační diagram

Entitně-relační diagram ERD se používá pro abstraktní a **konceptuální** znázornění dat a představuje grafické znázornění modelované reality. Základními prvky entitně-relačního modelu jsou entity, atributy a vazby.

- **Entita** – Množina věcí z reálného světa, které mají stejné vlastnosti.
- **Atribut** – Jedna vlastnost entity.
- **Vazba** – Znázorňuje vztahy mezi entitami.

Entitně relační diagram pro vyvíjenou aplikaci je na obrázku 4.4. Z důvodu zachování přehlednosti diagram neobsahuje atributy jednotlivých entit. Ty jsou detailně popsány v následujících kapitolách.



Obrázek 4.3 ERD vyvíjené aplikace

Popis diagramu

První entitou je Případ (Case) umožňující evidovat a zároveň oddělovat jednotlivé případy. Bude obsahovat vazbu na profil osoby (Profile). Entita Profil osoby (Profile) je základní entitou a sdružuje všechny uživatelovi účty, které reprezentuje entita Účet (Account). Entita Typ účtu (Type) s vazbou na Účet reprezentuje typ účtu. Odeslané a přijaté zprávy, včetně hovorů, jsou reprezentovány entitou Zpráva

(Message), která opět obsahuje vazbu na Účet. Poslední entitou je Seznam kontaktů (Contact List), který je navázán na Účet a Profil osoby.

Popis atributů entit

V této kapitole jsou vypsány jednotlivé atributy entit.

Crimcase

Entita Case (Případ) obsahuje následující atributy:

- Název,
- Evidenční číslo,
- Popis.

Profile

Entita Profile (Profil osoby) obsahuje následující atributy:

- Jméno,
- Příznak, zda-li se jedná o hlavní profil,
- Obrázek – fotku osoby.

Account

Entita Účet (Account) obsahuje následující atributy:

- Název, v tomto případě primární klíč.

Type

Entita Typ (Type) obsahuje následující atributy. Z hlediska modelu se jedná o číselník, je použita pro zachování normálních forem.

- Jméno,
- Popis.

Contactlist

Entita Seznam kontaktů (Contact List) obsahuje kromě primárního klíče pouze atributy, které jsou v konceptuálním i fyzickém modelu cizími klíči entit Účet a Profil.

Message

Entita Zpráva (Message) obsahuje následující atributy:

- Čas,
- Doba trvání,
- Předmět,
- Text.

Popis vazeb

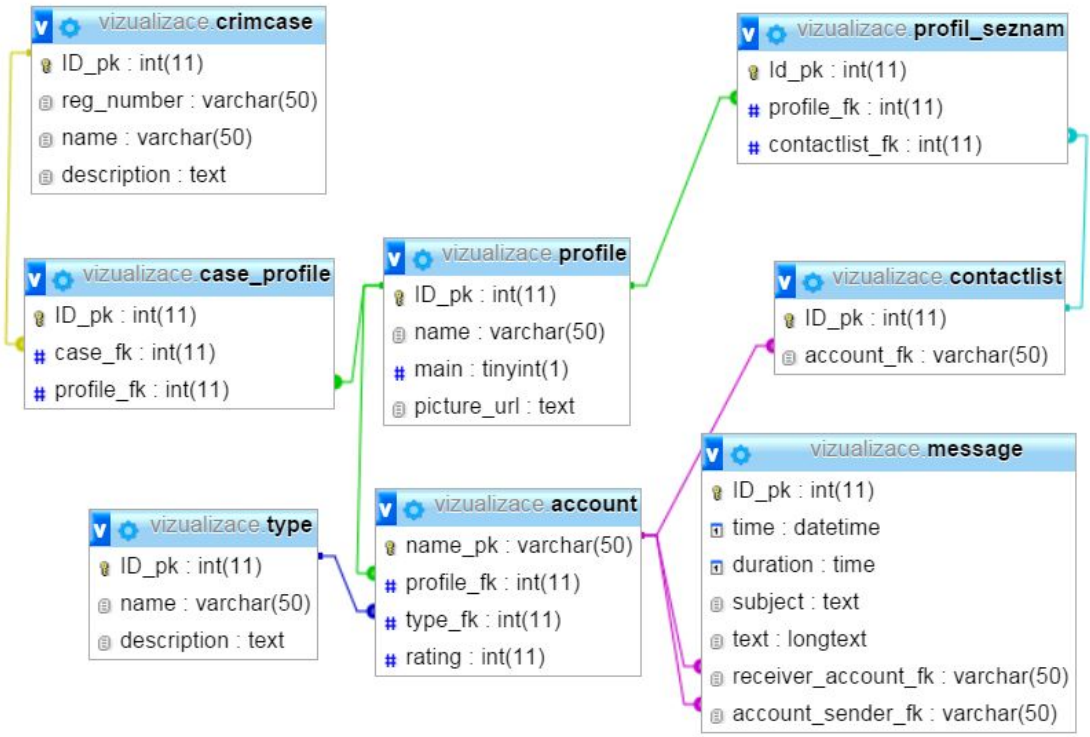
- **Profile → assigned to→ Crimcase** – Profil je přiřazen k jednomu či více případům. Jeden případ může obsahovat více profilů.
- **Profile → has → Account** – Profil může mít více účtů. Každý účet je přiřazen k jednomu profilu.
- **Type → has→ Account** – Účet má svůj typ. K jednomu typu je přiřazeno více účtů.
- **Account → has → Message** – K účtu může být přiřazeno více zpráv. Každá zpráva je přiřazena k více účtům.
- **Account → has → Contactlist** – Účet může mít svůj seznam kontaktů. Každý seznam kontaktů je přiřazen k jednomu účtu.
- **Contact List → contains → Profile** – Seznam kontaktů obsahuje jeden nebo více profilů. Jeden profil může být ve více seznamech kontaktů.

4.6.2 Fyzický model databáze

Tato kapitola popisuje fyzický model databáze, který byl vytvořen na základě požadavků a ERD z předchozí kapitoly. Tabulky a následný celý model byl vytvořen pomocí phpMyAdmin, který je součástí nástroje XAMPP.

Na rozdíl od entitně-relačního modelu jsou zde už zobrazeny primární a cizí klíče. Primární klíč je označen žlutým klíčkem před jménem atributu, za atributem je pak zvolený datový typ. Za zmínku také stojí barevné znázornění vazeb a použití primárních klíčů jako cizích klíčů. Každá vazba má svou barvu, je-li primární klíč jedné tabulky použit jako cizí klíč ve více tabulkách, jsou vazby znázorněny jednou barvou.

Z hlediska vazeb se, na rozdíl od entitně-relačního modelu, změnila vazba mezi seznamem kontaktů (Contactlist) a profily osob (Profiles) a také případem (Case) a profilem osoby (Profile). Z důvodu vazby M:N byly přidány asociační tabulky profil_seznam a case_profile. Dále jsou zde dvě vazby mezi tabulkami Accounts a Messages. V tomto případě je tím dán účet příjemce a odesílatele zpráv.



Obrázek 4.4, Fyzický model z databáze. Získáno z phpMyAdmin

4.7 Požadavky pro běh aplikace

Pro běh aplikace je nutné splňovat následující požadavky:

- OS Windows 7/8.1/10
- .NET Framework 4.5

Výše zmíněné požadavky budou zkontrolovány během instalace. Pro běh aplikace je také mít nainstalovaný databázový server:

- mySQL server (nejlépe XAMPP)

Instalační soubor a samotná databáze bude k dispozici na přiloženém CD.

4.8 Architektura aplikace

Z hlediska architektury aplikace se jedná o klasickou Windows Forms Application, kde prezenční vrstvu zajišťují formuláře grafického rozhraní knihovny Windows Forms. Datová vrstva je prezentována MySQL serverem, do kterého je přistupováno pomocí ORM z EF 6.0, který obsahuje model databáze. Mezi prezenční vrstvou a ORM je část business logiky.

5. Implementace

Kapitola popisuje implementaci aplikace pro vizualizaci vazeb v návaznosti na požadavky a návrh z předešlé kapitoly. Samotná kapitola je rozdělena do několika dílčích kapitol, které popisují implementaci jednotlivých částí aplikace. Dále jsou popsány vybrané použité technologie a případné komplikace spojené s jejich využitím.

5.1 Entity Framework

Jak už bylo zmíněno výše, pro přístup k datům bylo použito objektově relační mapování pomocí Entity Frameworku. Při začátku implementace autor preferoval využití metody Model First, kdy je ve Visual Studiu vytvořen model databáze, následně podle modelu vytvořen SQL skript a ten je spuštěn na databázovém serveru, kde se databáze vytvoří. Tento postup platí pro MS SQL servery, nicméně díky použití MySQL nebyl možný. Důvodem je nekompatibilita aktuálních verzí .NET Connectoru a pluginu MySQL for Visual Studio s nejnovější verzí Entity Framework v6.

Z popsaného důvodu byla, jak už bylo zmíněno výše, vytvořena databáze MySQL pomocí PhpMyAdmin, následně byla použita metoda Entity Framework Database First a dle databáze byl vytvořen EF model.

5.2 Připojení k databázi pomocí EF

Následující kapitola popisuje připojení k databázi a následně jsou demonstrovány příklady použití Entity Frameworku.

5.2.1 Connection string

K databázi přistupuje Entity Framework pomocí třídy *DbContext*. Ve stručnosti nám tato třída zajišťuje možnost pracovat s daty v databázi jako s objekty. Tím pro programátora odpadá nutnost psaní SQL dotazů.

Po vytvoření modelu Entity Frameworku, v implementovaném programu pojmenovaném *Model.edmx*, byla mimo jiné modelem vytvořena i třída *VizualizaceEntities*. Tato třída dědí z výše zmíněné třídy DbContext a součástí konstruktoru je *connection string* pro připojení k databázi. Díky této třídě je pak při potřebě práce s daty v databázi vytvořen kontext, jehož prostřednictvím je přistupováno do databáze. Následuje ukázka hlavičky konstruktoru, kde jsou pro větší přehlednost tučně zvýrazněny instanční proměnné, které byly do connection stringu přidány z důvodu možnosti jeho modifikace uživatelem. Jejich hodnoty určují adresu MySQL serveru, port, uživatelské jméno a heslo.

```
public vizualizaceEntities():base("metadata=res://*/Model.csdl|
res://*/Model.ssdl|res://*/Model.msl; provider=MySql.Data.MySqlClient;provider
connection string=\";server=" + serverAddress + ";port=" + port + ";user id="
+ userID + ";password=" + password +
";Charset=cp1250;persistsecurityinfo=True;database=vizualizace\";")
```

5.2.2 Příklad dotazů při použití Entity Framework

Jak už bylo zmíněno výše, ORM přináší programátorům možnost přistupovat k datům v databázi jako k objektům a odpadá tak nutnost použití SQL dotazů. Následuje příklad použití, kde je vytvořen kontext třídy VizualizaceEntities a následně demonstrováno vytvoření, změna a odstranění záznamu. Jedná se o zjednodušený příklad vytvoření, změny a smazání případu (crimcase) z tabulky crimcase.

- **Vytvoření případu**

```
VizualizaceEntities context = new VizualizaceEntities();
crimcase newcase = new crimcase(jmeno, evidencniCislo, popis);
context.crimcase.add(newcase);
context.SaveChanges();
```

- **Změna případu** (případ *selected* vybraný z *listBoxCases*)

```
VizualizaceEntities context = new VizualizaceEntities();
crimcase selected = (crimcase)listBoxCases.SelectedItem;
selected.name =textBoxName.Text;
selected.description = textBoxDescription.Text;
selected.regnumber = textBoxReqNumber.Text;
context.SaveChanges();
```

- **Smazání případu**

```
VizualizaceEntities context = new VizualizaceEntities();
crimcase toRemove = (crimcase)listBoxCases.SelectedItem
context.crimcase.Remove(toRemove);
context.SaveChanges;
```

5.3 Formulář pro přihlášení

Pro přihlášení bylo implementováno grafické rozhraní, konkrétně formulář LoginForm. Formulář uživateli slouží k zadání požadovaných informací, které jsou nutné pro vytvoření connection stringu a následné připojení k databázi pomocí EntityFrameworku.

Obrázek 5.1, Formulář pro přihlášení k databázi

Aby uživatel nemusel při každém spuštění zadávat přihlašovací údaje, je součástí implementace funkcionalita, kdy jsou zadané údaje uloženy do textového souboru credentials.txt.

V následují ukázce kódu je zobrazena situace, kdy jsou po stisknutí tlačítka *Přihlásit* nastaveny přihlašovací údaje pomocí metody *setCredentials()* (na zadané hodnoty byla použita metoda *Trim()*, která odstraní případné mezery na začátku nebo konci uživatelem zadaných hodnot). Dále je obsah souboru vymazán (přepsán prázdným řetězcem), následně jsou pomocí *StreamWriteru* zapsány přihlašovací údaje.

```
vizualizaceEntities.setCredentials(textBoxUser.Text.Trim(),
textBoxPort.Text.Trim(), textBoxServer.Text.Trim(),
textBoxPasssword.Text.Trim());
```

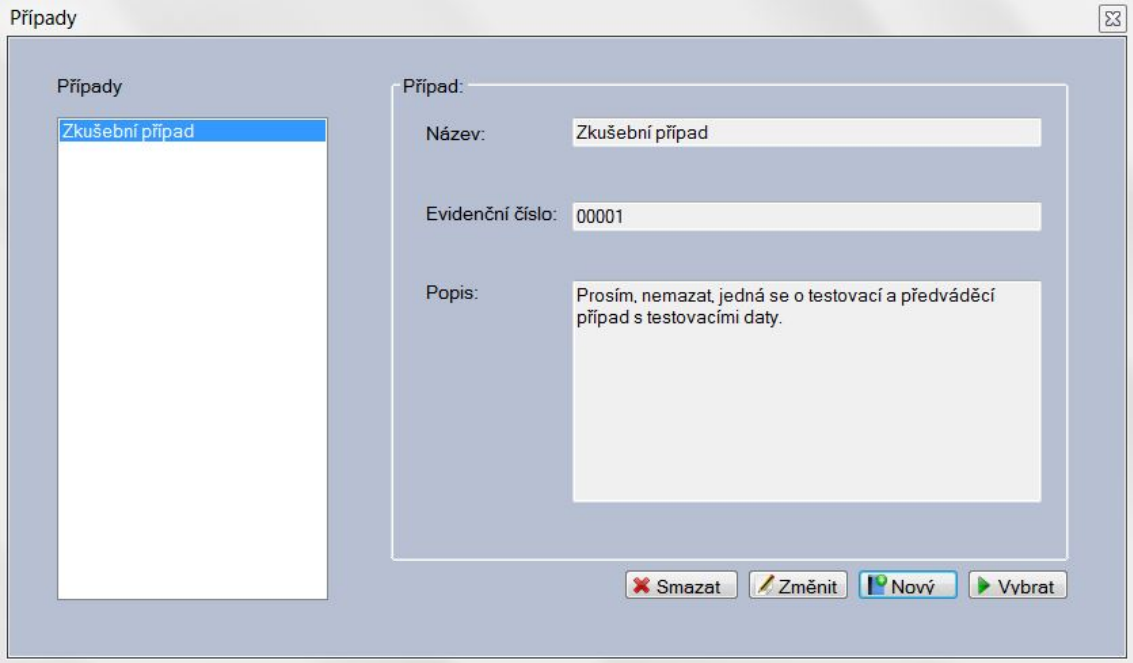
```
File.WriteAllText(File.WriteAllText(Environment.GetFolderPath(Environment.Spec
ialFolder.LocalApplicationData) + "\\Dipcom\\Files\\credentials.txt", "");
StreamWriter writer = new
StreamWriter(Environment.GetFolderPath(Environment.SpecialFolder.LocalApplicat
ionData) + "\\Dipcom\\Files\\credentials.txt);
writer.WriteLine(textBoxServer.Text.Trim());
writer.WriteLine(textBoxPort.Text.Trim());
writer.WriteLine(textBoxUser.Text.Trim());
writer.WriteLine(textBoxPasssword.Text.Trim());
writer.Flush();
writer.Close();
```

Soubor je vytvořen až po prvním spuštění celé aplikace a obsahuje adresu serveru, port, jméno a heslo uživatele. Při dalším spuštění jsou již údaje načteny přímo ze souboru a není tedy nutné, aby je uživatel zadával. V případě, že se přihlášení podaří, dojde k zobrazení formuláře s případy (viz. další kapitola).

V případě, že se po startu aplikace přihlášení nepodaří, např. z důvodu změny adresy, hesla atp. je formulář znovu zobrazen, do textBoxů jsou načteny původní hodnoty a uživatel je vyzván k novému přihlášení. Poté opět dojde k aktualizaci souboru credentials.txt. a následuje formulář s případy (viz. další kapitola). Samotný soubor je umístěn ve složce AppData\Local\Dipcom\Files, která se nachází ve složce uživatele.

5.4 Správa případů

Pro správu případů bylo implementováno grafické rozhraní (formulář *CasesForm.cs*), které umožňuje uživateli vytvářet, měnit a mazat případy. Po vytvoření prvního případu je v dokumentech uživatele vytvořen adresář *Relations*, v něm je pak vytvořen adresář pojmenovaný podle jména případu. Po vytvoření prvního případu už budou vytvářeny jen adresáře pro nové případy, jejichž názvy budou odpovídat názvům případů. V těch budou ukládána data týkající se jednotlivých případů respektive fotografie k profilům.



Obrázek 5.2, Grafické rozhraní Správy případů

Výběrem v listBoxu na levé straně (jsou zde zobrazena jména případů) dojde ke změně textů textBoxů a zobrazení informací o aktuálně vybraném případu.

```
private void listBoxCases_SelectedValueChanged(object sender, EventArgs e)
{
  if (listBoxCases.SelectedItem != null)
   {
     crimcase selected = (crimcase)listBoxCases.SelectedItem;
     textBoxName.Text = selected.name.ToString();
     textBoxReqNumber.Text = selected.reg_number.ToString();
     textBoxDescription.Text = selected.description.ToString();
   }
}
```

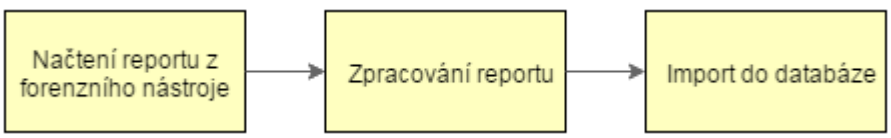
Následuje příklad zdrojového kódu – metody, která vytváří nový případ. Nejprve se pomocí metody *ClearTextBoxes()* vymaže obsah textBoxů. Poté je textBoxům nastaven parametr *readOnly* na *false* a uživateli je umožněno jejich vyplnění. Text tlačítka pro vytvoření je změněn na „Uložit“ a je také zviditelněno tlačítko pro zrušení pro případ, že uživatel nechce ve vytváření případu pokračovat. Dále vypne ostatní tlačítka pomocí metody *EnableButtons()*, kde každý boolean parametr odpovídá jednomu tlačítku. Po vyplnění jména případu, evidenčního čísla a popisu stiskne uživatel tlačítko *Uložit*. Pomocí konstruktoru je vytvořen případ, který je pomocí

kontextu EF uložen do databáze. Následuje opětovné zapnutí tlačítek a textBoxy jsou opět nastaveny jen pro čtení. Nakonec je užita metoda *RestartListbox()*, která obnoví obsah listBoxu se jmény případu.

```
private void CreateCase()
   {
       if (textBoxName.ReadOnly == true)
       {
           ClearTextBoxes();
           TBReadOnly(false); // readonly off
           EnableButtons(false, false, true, false);
           ShowLabels(true);
           buttonNew.Text = "Uložit";
           buttonCancel.Visible = true;
       }
       else
       {
        crimcase newcase = new crimcase(textBoxName.Text,
textBoxDescription.Text, textBoxReqNumber.Text); // vytvoření

           context.crimcase.Add(newcase); //
           context.SaveChanges();// uložení do db
           CreateDirectories(newcase); //vytvoření adresáře případu
           EnableButtons(true, true, true, true);//enable buttons
           TBReadOnly(true);
           ShowLabels(false); //
           buttonNew.Text = "Nový";
           buttonCancel.Visible = false;
           ClearTextBoxes();
           RestartListbox();
            }
   }
```

5.5 Zpracování vybraných výstupů pro import

Postup importu kontaktů z výstupních souborů se skládá ze tří pomyslných kroků.



Obrázek 5.3, Postup importu

5.5.1 Zpracování XLS z UFED

Pro import z UFED byl vybrán soubor XLS. Pro práci s tímto typem souborů přicházely v úvahu tři možná řešení respektive typy knihoven:

- **OLEDB** –Object Linking and Embedding Database (OLEDB) je API navržené společností Microsoft, které umožňuje přistupovat k různým datům stejným způsobem. Z programátorského hlediska se jedná o použití dotazu SQL na soubor XLS.
- **COM Interop Excel** – Component Object Model (COM) umožňuje práci s objektem pomocí nadřazené aplikace. Laicky řečeno, COM Interop umožňuje programátorovi pracovat se soubory podporovanými Excelem právě prostřednictvím Excelu. V takovémto případě však musí být v počítači, kde běží programátorova aplikace, nainstalován i MS Excel.
- **Knihovna třetí strany** – Třetí možností bylo použití již hotové knihovny, která značně usnadňuje práci se samotným souborem, a jsou implementovány metody, které umožňují snadné čtení i zápis např. do příslušných buněk. Autor práce ale bohužel nenalezl knihovnu, která by nebyla pod komerční licencí a nevyžadovala by zakoupení knihovny, nebo takovou, ke které by byla dostatečná dokumentace.

Z výše zmíněných důvodů a vlastností byl pro implementaci vybrán OLEDB.

Pro zpracování výstupů z UFED byla vytvořena třída *ParseUFED*. Samotná implementace zpracování souboru XLS z UFED byla rozdělena do několika kroků, a to z důvodu struktury souboru, který byl popsán v kapitole Výstupy z UFED Physical Analyzer. Následuje příklad kroku a zdrojového kódu, kde je nejprve otevřen list Kontakty a vybrány pouze dva sloupce. Druhý sloupec obsahuje jméno kontaktu a devátý sloupec pak záznam, který obsahuje informace, které je třeba dále zpracovat, to znamená získat přímo telefonní číslo, nebo e-mailovou adresu atd.

Nejprve je vytvořeno spojení s příslušným *connection stringem*. Atribut *provider* určuje právě typ OLEDB driveru, proměnná *pathToFileUFED* obsahuje cestu k vybranému souboru, ze kterého bude proveden import. Extended Properties pak určují typ souboru (zde Excel 8.0 pro typ souboru XLS) a HDR atribut určuje, zda první řádek

obsahuje hodnoty (No) nebo se jedná o jména sloupců (Yes). V tomto konkrétním případě byl zvolen *HDR= No* a to z toho důvodu, že první řádek v importovaném souboru obsahuje v druhém sloupci „nadpis“ *Kontakty(x)*, kde x je počet nalezených kontaktů. V tomto případě se potom nebylo možné dotazovat na druhý sloupec jako na F2. Následuje příklad vytvoření spojení:

```
MyConnection = new System.Data.OleDb.OleDbConnection
(@"provider=Microsoft.Jet.OLEDB.4.0;Data Source=" + pathToFileUFED +
";Extended Properties=\"Excel 8.0;HDR=No;\"");
```

Dále je vytvořen dotaz na příslušný list v importovaném souboru. Zde jsou vybrány dva sloupce F2 a F9 z listu Kontakty a ukládány do *DataTable DTcontacts*. Nakonec je ukončeno spojení.

```
MyCommand = new System.Data.OleDb.OleDbDataAdapter("select [F2],[F9] from
[Kontakty$]", MyConnection);
MyCommand.Fill(DTContats);
MyConnection.Close();
```

V takovémto případě by ale nebylo možné alespoň částečně reagovat na změnu formátu výstupů. Pro tento případ byl přidán jednoduchý konfigurační soubor (ufedconfig.txt ve složce Files), jehož změnou je uživateli umožněno změnit názvy listů, čísla sloupců a řádek, od kterého se začnou načítat data.

Struktura ufedconfig.txt

```
Kontakty=Kontakty$A:I
KontaktyJmeno=F2
Zaznam=F9
```

Data ze souboru jsou načtena pomocí metody *configure(string file)* do datové struktury slovník, konkrétně do instanční proměnné *Dictionary<string, string> config = new Dictionary<string, string>();*, kde klíčem je řetězec, který se nachází před rovnítkem, hodnotou pak řetězec, který se nachází za ním.

```
private void configure(string file)
{
  StreamReader reader = new StreamReader(file, System.Text.Encoding.Default);
  string buffer = reader.ReadLine();
  while(buffer != null && buffer != "")
  {
```

```
        int index = buffer.IndexOf("=");
        config.Add(buffer.Substring(0, index).Trim(),
        buffer.Substring(index+1).Trim());
        buffer = reader.ReadLine();
    }
}
```

Do dotazu na příslušný list byly tedy přidány reference na záznamy ve slovníku. Viz. následující ukázka:

```
MyCommand = new System.Data.OleDb.OleDbDataAdapter("select [" +
config["KontaktyJmeno"] + "],[" + config["Zaznam"] + "] from [" +
config["Kontakty"] + "]", MyConnection);
```

Následně jsou data z tabulky DTContacts z druhého sloupce zpracována pomocí metod *Substring()* a *Trim()*. Děje se tak proto, že buňky druhého sloupce obsahují následující řetězce (telefonní čísla a e-mailovou adresu autor práce změnil):

Telefon-Mobil: +420123456789

Telefon-Mobil: 123-456-789

E-mail-Jiné: mail.mail@mail.cz

ID uživatele-Facebook: 100001010101010

Je tedy nutné oddělit účet od řetězce a v případě, že záznam s telefonním číslem obsahuje pomlčku, tak ji odstranit a následně zvolit typ účtu dle obsahu řetězce. Dále už je kontakt uložen do databáze pomocí metody *importContats()* ze třídy *Import*.

Třída parseUfed obsahuje i metodu *reportMax(String pathToFileUFED)*, která vrací počet záznamů ve vybraných listech importovaného souboru. Tento údaj je potřebný pro vytvoření a správnou funkci *progressBaru* na formuláři *ImportForm*.

Stejně je proveden import i pro listy SMS Zprávy a Záznamy o hovorech. Liší se jen ve výběru sloupců a následné přípravě dat, aby data mohla být vložena do databáze pomocí metody *importMessage()* ze třídy *Import*.

5.5.2 Zpracování CSV z MPEP

Pro import souborů CSV z MPEP byl také vybrán OLEDB. Samotná implementace se ale liší, a to jak v samotném *connection stringu*, tak ve struktuře a počtu načítaných sloupců. Dalším rozdílem je, že k načtení dat je potřeba více souborů. Proto každý musí být v grafickém rozhraní vybrán zvlášť. Následuje příklad, kde jsou opět importovány kontakty.

V samotném vytvoření spojení se zde liší proměnná *pathOnlyMPE*, která reprezentuje pouze cestu k souboru (bez jména souboru). Ta je uložena v proměnné *fileNameMPEP* a je použita na jiném místě dotazu. Dále se liší Extended Properties a v jeho parametru FMT, který určuje oddělovač v CSV souboru. V případě souborů z MPEP je zde čárka.

Následuje příklad spojení a dotazu, ve kterém je stejně jako v případě souborů z UFED umožněno změnou konfiguračního soubor (mpepconfig.txt) alespoň částečně reagovat na možnou změnu formátu vygenerovaného reportu. Struktura souboru a princip zůstal stejný jako v předešlém případě.

```
MyConnection = new System.Data.OleDb.OleDbConnection
(@"provider=Microsoft.Jet.OLEDB.4.0;Data Source=" + pathOnlyMPEP + "; Extended
Properties=\"Text;HDR=No;FMT=Delimited(,);\"");

MyCommand = new System.Data.OleDb.OleDbDataAdapter("select [" +
config["KontaktJmeno"] + "],[" + config["KontaktCislo"] + "],[" +
config["KontaktMail"] + "] from [" + fileNameMPEP + "]", MyConnection);
```

Dotaz se v tomto případě dotazuje na tři sloupce. F1 se jménem, F9 s telefonním číslem a F12 s e-mailovou adresou (nastaveno v konfiguračním souboru). Při práci s *DataTable*, která sloužila pro ukládání výsledků dotazu, musela být brána na zřetel i skutečnost, že v případě výstupního souboru CSV z MPEP je zde menší komplikace. Jméno kontaktu je vždy na řádku s indexem *i*, zatímco záznamy o telefonu a e-mailu ve sloupcích F9 a F12 pro konkrétní jméno kontaktu jsou o řádek níže.

Po naplnění *DataTable* už dochází k samotnému ukládání záznamů do databáze. V tomto konkrétním případě je uložen účet telefonu. Pokud je k dispozici i e-mail, je uložen i účet e-mailu. Import do databáze probíhá pomocí metody importMessage() ze třídy Import.

5.5.3 Zpracování XML z XRY

Pro práci a následný import XML souboru z XRY byla použita třída *XmlDocument* z .Net Frameworku 4.5. Jako v předešlých kapitolách i zde bude popsán příklad importu seznamu kontaktů.

Například je inicializován nový XmlDocument a načten z vybrané cesty *pathToXRY*.

```
XmlDocument doc = new XmlDocument();
doc.Load(pathToXRY);
```

Následně je vytvořen XmlNodeList kontakty, ve kterém jsou všechny elementy *item* z elementu *Contacts/Kontakty* a následně pro každý element nalezen atribut jméno, mobil, e-mail.

```
XmlNodeList kontakty =
doc.SelectNodes(@"/views/view[@name='Contacts/Kontakty']/item");

            foreach (XmlNode node in kontakty)
            {
                XmlNode nameNode =
node.SelectSingleNode(@"./field[@name='Jméno']");
                XmlNode telephoneNode =
node.SelectSingleNode(@"./field[@name='Mobil']");
                XmlNode mailNode =
node.SelectSingleNode(@"./field[@name='Email']");
```

Existuje-li nameNode, telephoneNode, mailNode, jehož hodnota není *null*, je pro každý nalezen atribut *value*, ve kterém je přímo uložena „hodnota“ jména respektive samotné telefonní číslo nebo e-mail. Dále je pak záznam uložen do databáze.

5.5.4 Zpracování EML souborů

V následující kapitole je stručně popsána struktura a zpracování EML souborů. Pro jejich zpracování implementována třída parseEML.

Postup při zpracování EML souborů

Konstruktoru třídy je předána cesta k adresáři, ve kterém jsou uloženy EML soubory. Následně jsou do seznamu emlFiles načteny všechny cesty k EML souborům.

```
public ParseEML(String emlDirectory)
```

```
{
        emlFiles = new List<string>();
        this.emlDirectory = emlDirectory;

        foreach (String x in Directory.GetFiles(emlDirectory))
        {
            if (x.Substring(x.Length - 3) == "eml")
             {
                emlFiles.Add(x);
             }
        }
}
```

K samotnému zpracování byla implementována metoda parseMails(). V metodě je díky foreach cyklu zpracován každý soubor, jehož cesta je uložená v listu emlFiles.

Nejprve je použita metoda returnCharset(). Ta vrací charset použitý v .eml souboru. Nalezený charset (pomocí regulárního výrazu) je pak předán jako jeden ze dvou parametrů metodě decodeMessageText(). Ta převede text e-mailu z kódování quoted-printable do zvoleného charsetu.

```
private static string returnCharset(String filename)
        {
            string pattern = "charset=\".*?\"";
            Regex charsetRegex = new Regex(pattern);
            StreamReader readCharset = new StreamReader(filename);
            String celyText = readCharset.ReadToEnd();


            Match charsetMatch = charsetRegex.Match(celyText);
            String charset = charsetMatch.Value;
            charset = charset.Replace("charset=", "");
            charset = charset.Replace("\"", "");

            return charset;
        }
```

Před načtením samotného textu zprávy jsou načteny a zpracovány jednotlivé údaje – datum, odesílatel, příjemce a předmět. Tyto údaje jsou uvedeny na začátku souboru a od samotného textu zprávy jsou odděleny prázdným řádkem. Tyto údaje jsou

v cyklu načítány pomocí streamReaderu a dle počátku řádku (parametru) jsou pomocí přepínače (switch) načteny do příslušných proměnných (dle počátku řádku).

```
do
     {
         int index = lineBuffer.IndexOf(":"); // index dvojtečky
         string firstWord = ""; // řetězec před dvojtečkou
         string theRest = ""; // řetězec za dvojtečkou

         if (index != -1)
         {
                firstWord = lineBuffer.Substring(0, index);
                theRest = lineBuffer.Substring(index + 1).Trim();
         }
               else
               {
                  firstWord = "noEntry";
                  theRest = lineBuffer;
               }
               switch (firstWord)
               {
                 case ("Date"):
                            date = theRest;
                            lastEntry = date;
                            break;

                 case ("Subject"):
                            predmet = theRest;
                            lastEntry = predmet;
                            break;

                 case ("From"):
                            od = theRest;
                            lastEntry = mail;
                            break;

                 case ("To"):
                            ke = theRest;
                            lastEntry = ke;
                            break;

                 case ("noEntry"): // pro případ, kdy je přes více řádků
                           if(lastEntry != null)
                           {
                           lastEntry = lastEntry.ToString() + theRest;
                           }
                            break;

                        default:
                            lastEntry = null;
                            break;
                    }
                    lineBuffer = reader.ReadLine();
                    lineBuffer = lineBuffer.Replace("\"","");
 }
 while (lineBuffer.Trim() != "");// prázdný řádek, následuje text
```

Následuje zpracování výše nalezených dat. V následující ukázce je demonstrováno zpracování data e-mailu, předmětu a e-mailové adresy odesílatele, která je nalezena pomocí regulárního výrazu.

```
//zpracovani data – převedení do formátu vhodného k uložení do db
DateTime dateTime = DateTime.Parse(date, new
System.Globalization.CultureInfo("cs-CZ", true),
System.Globalization.DateTimeStyles.AssumeLocal);
date = dateTime.ToString();

// zpracovani predmetu
System.Net.Mail.Attachment attachmentPredmet =
System.Net.Mail.Attachment.CreateAttachmentFromString("", predmet);
predmet = attachmentPredmet.Name;

// zpracování odesílatele
System.Net.Mail.Attachment attachmentOd=
System.Net.Mail.Attachment.CreateAttachmentFromString("", od);
od = attachmentOd.Name;
Match matchOd = mail.Match(od);
od = matchOd.Value; // emailová adresa je nalezena pomocí //regulárního výrazu
```

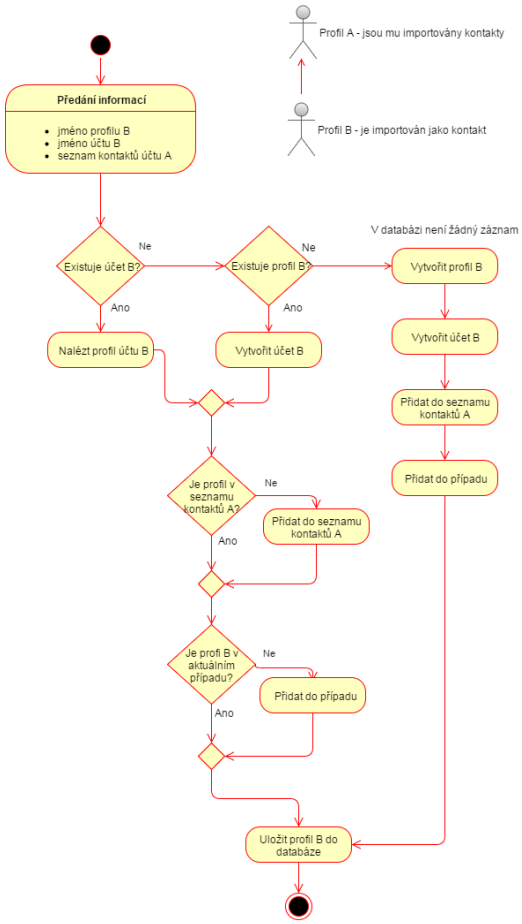
Následuje zpracování samotného textu zprávy a následný import do databáze. Ten se provede pouze v případě, že eml soubor obsahoval e-maily příjemce i odesílatele. V opačném případě by nemohl být uložen, a to nejen z důvodu konzistence dat, ale zároveň ze samotné podstaty vyvíjeného programu (vytvoření vazeb mezi osobami). Import je proveden metodou importMessage() ze třídy Import.

5.6 Import

K samotnému importu kontaktů a zpráv do databáze byla implementována třída Import. Její metody pro import kontaktu a zprávy jsou použity pro vkládání dat do databáze.

5.6.1 Import kontaktu

Pro import kontaktu byla implementována metoda ImportContact(). Ta je implementována tak, že jsou jí v parametrech předány informace o profilu A (jsou mu importovány kontakty), o profilu B (je importován jako kontakt) a číslo případu.



Obrázek 5.4, Activity diagram importu kontaktu

5.6.2 Import zprávy

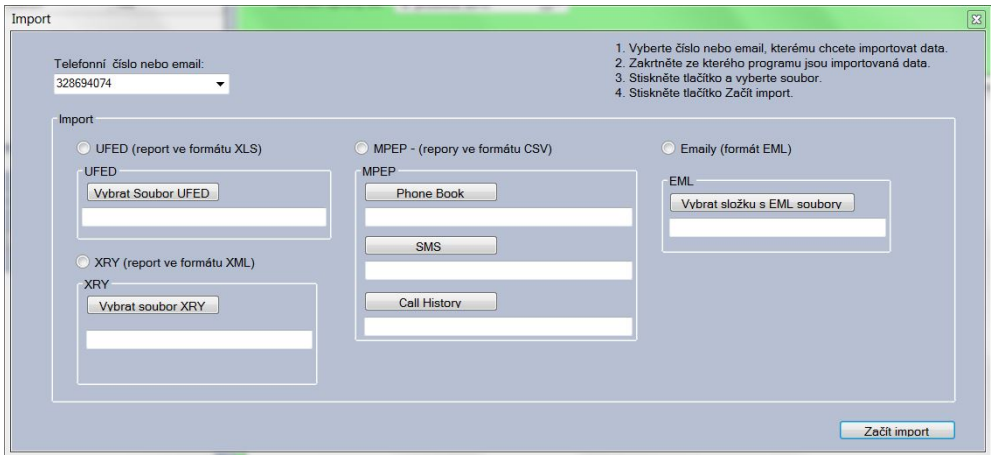
Pro import zprávy (e-mailů, textových zpráv a volání) byla implementována metoda ImportMessage(). Průběh importu zprávy do databáze probíhá z velké části stejně, jako import kontaktu, s tím rozdílem, že na konci je zpráva uložena do databáze.

Nicméně tomuto poslednímu kroku předchází opět kontrola, zda-li profil B a jeho účet existuje v databázi, zda-li přísluší aktuálnímu případu a zda-li je v seznamu kontaktů profilu a účtu A. Pokud nejsou splněny tyto podmínky, postupuje se stejně jako při importu kontaktu.

5.6.3 Grafické rozhraní importu

Pro import bylo implementováno grafické rozhraní ve formě formuláře *ImportForm*. Jemu je předán hlavní profil vybraný na hlavním panelu. Zde uživatel zaškrtnutím vybere příslušný typ souboru a následně zadá cestu k němu. Dále už probíhá samotný import v několika vláknech – podle typu souboru, resp. jejich počtu.

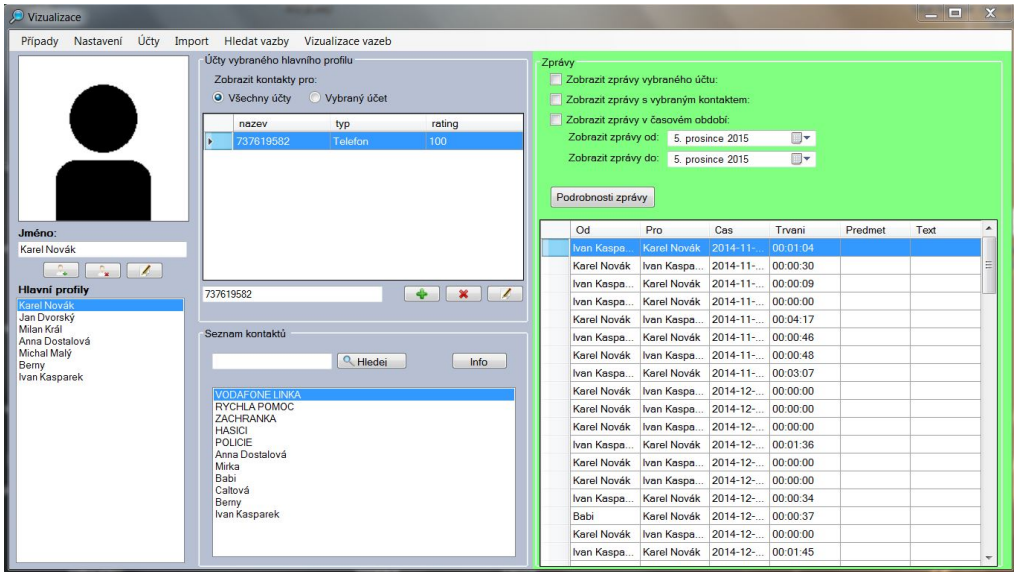
O dosavadním průběhu importu je uživatel informován pomocí *progressBaru,* který se zobrazí ve spodní části formuláře po stisknutí tlačítka *Začít import*. Jeho maximální hodnota je nastavena na hodnotu, kterou vrátí metoda returnMax z tříd pro zpracování souborů a jeho stav tedy odpovídá průběhu zpracování a importu.



Obrázek 5.5, Grafické rozhraní importu

5.7 Hlavní panel – zobrazení kontaktů a komunikací

Po vybrání případu se uživateli zobrazí hlavní formulář aplikace *MainForm*. Ten zajišťuje funkce pro prohlížení účtů, kontaktů účtů a zpráv hlavních profilů. Dále se z něj uživatel dostane na další panely sloužící k importu, hledání a vizualizaci vazeb.



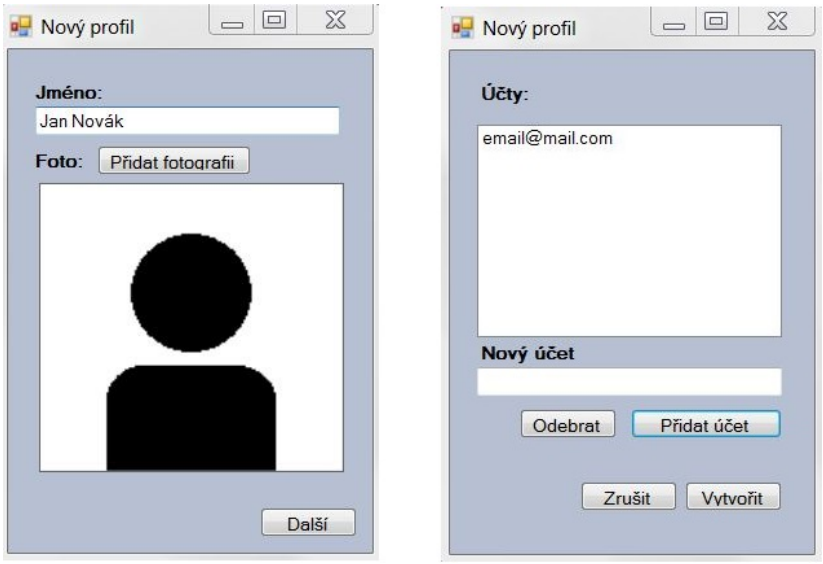
Obrázek 5.6, Grafické rozhraní - Hlavní panel

5.7.1 Hlavní profily

V levém sloupci uživatel vidí hlavní profily případu, pro vybraný hlavní profil je zobrazena fotografie (je-li k dispozici). Dále jsou zde tři tlačítka pro vytvoření, smazání a přejmenování profilu.

Vytvoření hlavního profilu

Pro vytvoření hlavního profilu byl implementován formulář *NewProfileForm*, ve kterém uživatel vyplní jméno profilu a přidá fotografii (je-li k dispozici). Na druhém panelu pak musí uživatel přidat alespoň jeden účet tomuto profilu. Více o vytváření účtu v kapitole Účty hlavního profilu.



Obrázek 5.7, Grafické rozhraní tvorby profilu

Smazání hlavního profilu

Po stisknutí tlačítka smazat, je uživatel vyzván pomocí *MessageBoxu* k potvrzení smazání. Před samotným smazáním profilu je nutná kontrola, zda-li mazaný profil není ještě v jiném případu. V takové situaci je pouze smazán záznam z tabulky *case_profile* a profil je odstraněn pouze z aktuálního případu. V opačném případě je profil smazán i se všemi účty, seznamy kontaktů a zprávami.

Přejmenování hlavního profilu

Uživatel má možnost přejmenovat vybraný hlavní profil přepsáním jeho jména v textBoxu a následným stisknutím tlačítka pro přejmenování. Během editace je změněna barva pozadí pomocí *eventu KeyPress*, původní barva je navrácena po tzv. opuštění focusu pomocí *eventu Leave*. Přejmenování profilu je implementováno tak, že je stejně jako v předchozím případě potvrzení uživatelem pomocí *DialogResult* viz. následující ukázka kódu.

```
if (listBoxMainProfiles.DataSource != null)
  {
       profile selected = context.profile.Single(x => x.ID_pk ==
       ((profile)listBoxMainProfiles.SelectedItem).ID_pk);
           if (selected != null)
           {
```

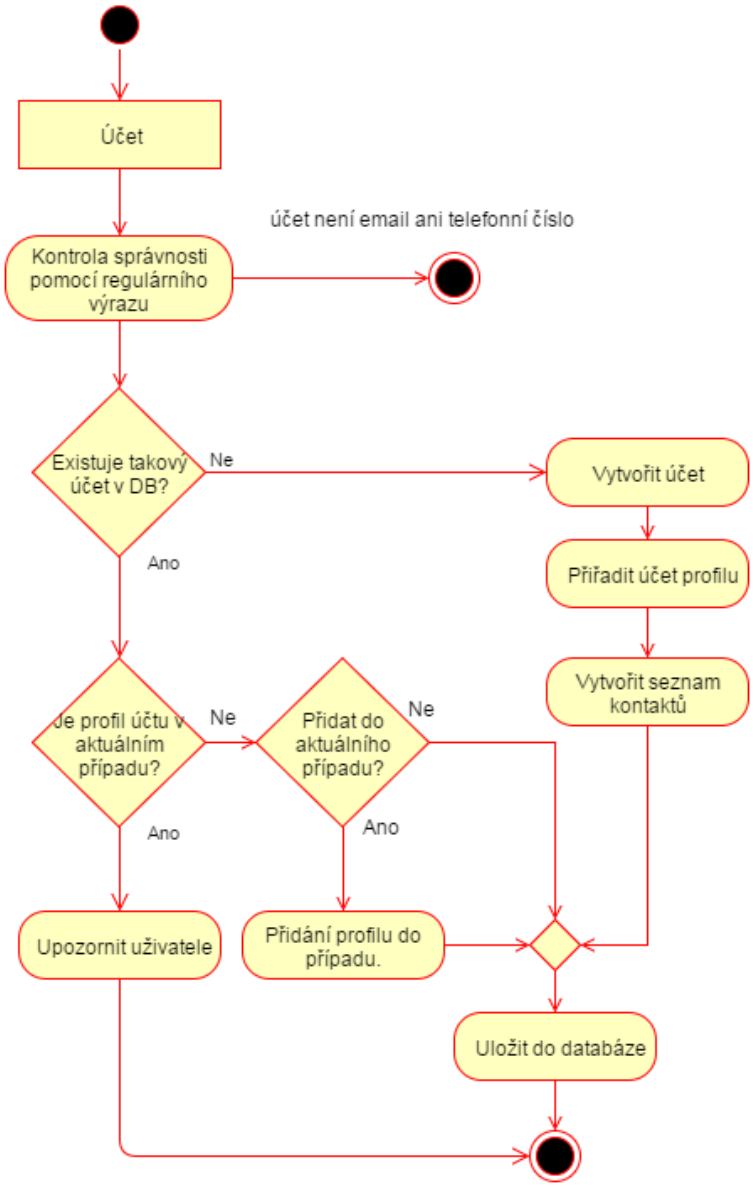
```
            DialogResult dlg = MessageBox.Show("Opravdu chcete přejmenovat
            profil " + selected.name + " na " + textBoxMainProfileName.Text,
            "Přejmenovat profil?", MessageBoxButtons.YesNo);
                   if (dlg == DialogResult.Yes)
                   {
                      selected.name = textBoxMainProfileName.Text;
                      context.SaveChanges();
                      this.restartForm();
                   }
                    else textBoxMainProfileName.Text = selected.name;
          }
}
```

5.7.2 Účty hlavního profilu

V groupBoxu *Účty vybraného hlavního profilu* má uživatel možnost prohlížet účty vybraného hlavního profilu. Pro účet je zobrazeno jméno, typ účtu a jeho rating. Zaškrtnutím jednoho ze dvou radioButtonů (*Všechny účty* nebo *Vybraný účet*) má pak uživatel možnost zvolit zobrazení kontaktů v groupBoxu Seznam kontaktů. Jsou buď zobrazeny všechny kontakty ze všech účtů, nebo jen kontakty z vybraného účtu. Dále má uživatel, stejně jako u profilů, možnost přidat, odstranit a přejmenovat účet.

Vytvoření účtu

Vytvoření účtu je znázorněno v následujícím diagramu aktivit.



Obrázek 5.8, Activity diagram vytvoření účtu

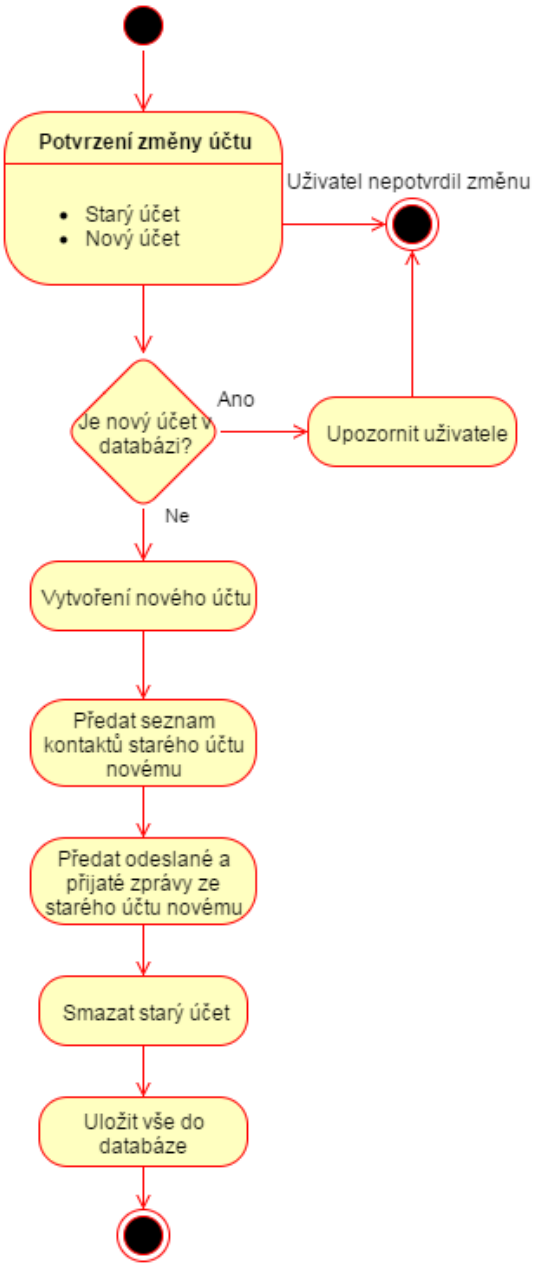
Smazání účtu

V následující ukázce je demonstrován kód, který po stisknutí tlačítka odebere vybraný účet.

```
if (dataGridViewAccounts.SelectedRows.Count <= 0) return;
       DialogResult dlg = MessageBox.Show("Opravdu smazat
       účet?","",MessageBoxButtons.YesNo);
       if (dlg == DialogResult.Yes)
        {
      context.account.Remove(((AccountsViewWrapper)dataGridViewAccounts.Select
edRows
      [0].DataBoundItem).acc);
        context.SaveChanges();
        restartForm();
        }
}
```

Přejmenování účtu

Přejmenování účtu v pravém slova smyslu není možné. Jeho jméno je zároveň i primárním klíčem a je tak zajištěno, aby účet byl v databázi právě jen jednou. Aby uživatel mohl přejmenovat účet, došlo-li například k překlepu, kterého si nevšiml během vytváření profilu a jeho účtů, byla implementována metoda, která je znázorněna v následujícím diagramu.



Obrázek 5.9, Activity diagram přejmenování účtu

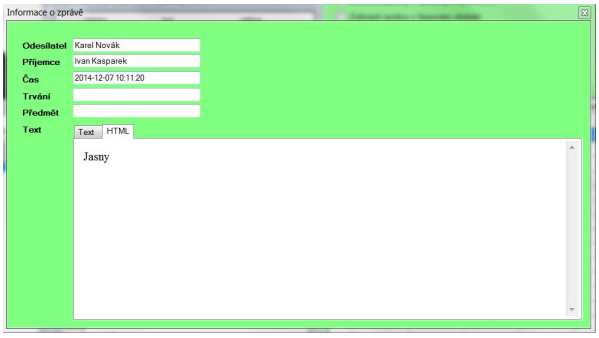
5.7.3 Zprávy

V groupBoxu *Zprávy* je pak uživateli k dispozici prohlížení e-mailů, zpráv a volání. Pro zobrazení zpráv jsou přítomny následující filtry:

- **Zobrazit zprávy vybraného účtu** – Zobrazí pouze zprávy, které přísluší vybranému účtu hlavního profilu. Např. pouze zprávy a volání z příslušného telefonního čísla.
- **Zobrazit zprávy s vybraným kontaktem** – Zobrazí zprávy pouze s vybraným kontaktem ze seznamu kontaktů.
- **Zobrazit zprávy v časovém období** – Zaškrtnutím možnosti časového filtru a výběrem konkrétního data má uživatel možnost zobrazit zprávy nebo hovory, které proběhly během vybraného časového období.

Zaškrtnutím příslušných *checkBoxů* dojde k zobrazení zpráv dle vybraných filtrů. Není-li zaškrtnutý žádný *checkBox*, jsou po stisknutí tlačítka vypsány všechny zprávy, e-maily a volání vybraného hlavního profilu.

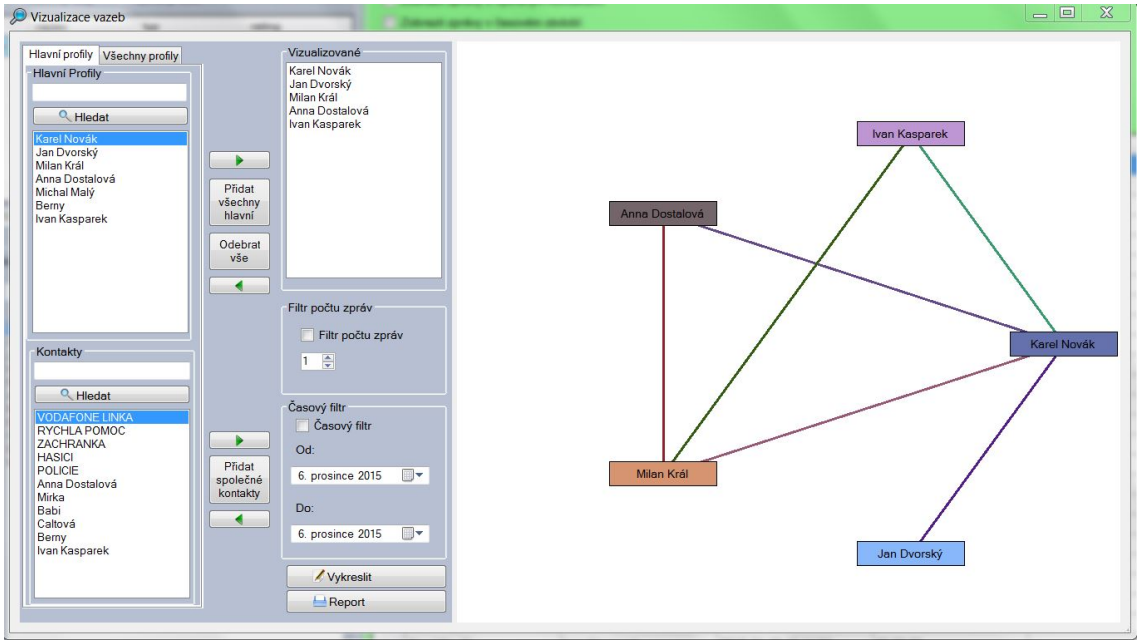
Tlačítko *Podrobnosti zprávy*, *dvojklik* nebo výběrem z *kontextového menu* po stisknutí *pravého tlačítka myši* na vybraný řádek v seznamu zpráv zobrazí okno s celou zprávou, které bylo implementováno jako formulář *MessageInfo*. Zde jsou, kromě údajů o zprávě nebo volání, také k dispozici dva náhledy (implementovány jako *tabControl*) na text zprávy. Možné je jak textové zobrazení, tak zobrazení pomocí komponenty *WebBrowser*. Tato možnost byla implementována pro případ zobrazení e-mailů (nejčastěji obchodních sdělení), které obsahují HTML kód. Díky WebBrowseru je text zprávy zobrazen tak, jak je možné jej vidět v prohlížeči. Viz obrázek.



Obrázek 5.10, Grafické rozhraní Informace o zprávě

5.8 Vizualizace vazeb

K realizaci funkce vizualizace byla implementována třída *Relations.cs* a formulář grafického rozhraní *RelationsForm.cs.* Dále autor popisuje implementaci podle toho, jakým způsobem bude tato funkcionalita používána uživatelem aplikace.



Obrázek 5.11, Grafické rozhraní Vizualizace vazeb

Na formuláři *RelationsForm* jsou k dispozici dva panely – *PanelRight* a *PanelLeft.* Levý panel slouží uživateli k výběru profilů, pro které budou nalezeny a vykresleny vazby. Pravý panel slouží k samotnému grafickému výstupu – vizualizaci vazeb tak, jak bylo popsáno v kapitole Návrh vizualizace vazeb.

Na levém panelu má uživatel k dispozici ovládací prvek *TabControl* se dvěma stránkami *tabPages.* Zde je možný výběr dle hlavních profilů a jejich kontaktů nebo výběr ze všech profilů v rámci případu. Tyto seznamy jsou implementovány jako ovládací prvky *listBox.* V listBoxech je možné hledat požadovaný profil pomocí *textBoxu*, kam uživatel může vepsat část hledaného jména. Po stisku tlačítka *Hledat* jsou pak zobrazeny jen profily, jejichž jména obsahují požadovaný výraz. Po vymazání výrazu z textBoxu a následném stisknutí tlačítka *Hledat* jsou zobrazeny opět všechny příslušné profily.

ListBox *Vizualized* umístěný v groupBoxu *Vizualizované* pak slouží jako seznam těch profilů, mezi kterými budou nalezeny a zobrazeny vazby. Výběrem v příslušném

listBoxu a stisknutím tlačítka se *šipkou doprava* je do listBoxu *Vizualized* přidán vybraný profil. Stiskem tlačítka *Přidat všechny hlavní* dojde k přidání všech hlavních profilů případu. Před každým přidáním proběhne kontrola, zda se již vybraný profil v listBoxu *Vizualized* nenachází a pokud ano, není již přidán. K odebrání profilů Vizualized pak slouží tlačítka s *šipkou vlevo* respektive tlačítko *Odebrat vše*, které vymaže celý listBox *Vizualized*.

Vazby jsou tvořeny pomocí třídy *Relation.* Ta vytváří vazbu mezi dvěma profily p1 a p2 na základě dotazu do databáze. Jsou procházeny všechny účty obou profilů. Je-li v profilu p1 účet, jehož kontaktem nebo příjemcem či odesílatelem alespoň jedné zprávy je účet profilu p2, je vytvořena vazba. Dotaz je položen i opačně (p1 zaměněno za p2), aby se eliminovala možnost, že pro profil p1 nebudou k dispozici účty, které by v seznamu kontaktů nebo zprávách měly záznam o profilu p2. Viz. ukázka kódu.

```
foreach (account acc1 in profil1.account)
{
  foreach (account acc2 in profil2.account)
    {
    List<message>temp = context.message.Where(x => (x.account.name_pk ==
    acc1.name_pk && x.account1.name_pk == acc2.name_pk) || (x.account1.name_pk
    == acc1.name_pk && x.account.name_pk == acc2.name_pk)).ToList<message>();
    zpravy.AddRange(temp);

    var query_from_p1 = from x in context.account
            where x.name_pk == acc1.name_pk
            join y in context.contactlist on x.name_pk equals y.account_fk
            join z in context.profil_seznam on y.ID_pk equals z.contactlist_fk
            where z.profile_fk == x2
            select z;


      var query_from_p2 = from x in context.account
            where x.name_pk == acc2.name_pk
            join y in context.contactlist on x.name_pk equals y.account_fk
            join z in context.profil_seznam on y.ID_pk equals z.contactlist_fk
            where z.profile_fk == x1
            select z;

             if (query_from_p1.Count() + query_from_p2.Count() > 0)
             {
               asponKontakt = true;
             }
    }
}
```

Takto jsou vazby vytvořeny mezi všemi profily v listBoxu *Vizulalized* na formuláři *RelationsForm* a to pomocí metody createRelations(). Před vykreslením vazeb mezi vybranými profily jsou uživateli k dispozici dva filtry:

- **Časový filtr** – Vazby jsou tvořeny a následně zobrazeny na základě *zpráv* v zadaném časovém období.

- **Filtr počtu zpráv** – Vazby jsou tvořeny a následně zobrazeny v případě, že je počet *zpráv* mezi profily větší nebo roven nastavené hodnotě (prahu).

```
private void createRelations()
{
   int pocetProfilu = listBoxVizualized.Items.Count;
   for (int i = 0; i < pocetProfilu; i++)
   {
     for (int j = 1 + i; j < pocetProfilu; j++)
     {
         profile p1 = (profile)listBoxVizualized.Items[i];
         profile p2 = (profile)listBoxVizualized.Items[j];
         Relation vazba = new Relation(p1.ID_pk, p2.ID_pk, context);

    // vazba - je v kontaktech a nebo jsou zprávy
    if (!checkBoxMessageCount.Checked && !checkBoxTimeFilter.Checked &&
         (vazba.AsponKontakt || vazba.Zpravy.Count > 0))
       {
           relations.Add(vazba);
       }
    // vazba - casovy filtr a filtr poctu zprav
      else if (checkBoxTimeFilter.Checked && checkBoxMessageCount.Checked &&
                vazba.getBoth(dateTimeOd.Value, dateTimeDo.Value,
                Convert.ToInt32(numericUpDownMessageCount.Value)))
       {
           relations.Add(vazba);
       }
     // vazba - pocet zprav
      else if (!checkBoxTimeFilter.Checked && checkBoxMessageCount.Checked &&
vazba.getCountCondition(Convert.ToInt32(numericUpDownMessageCount.Value)))
       {
           relations.Add(vazba);
       }
       // vazba - casovy filtr
       else if (checkBoxTimeFilter.Checked && !checkBoxMessageCount.Checked)
       {
          if (vazba.getTimeCondition(dateTimeOd.Value, dateTimeDo.Value))
             {
             relations.Add(vazba);
             }}}}
```

Stisknutím tlačítka *Vykreslit* dojde k samotnému výše popsanému vytvoření vazeb a následně jsou vykresleny na pravém panelu. Vykreslení jednotlivých profilů je implementováno jako vytvoření a zobrazení prvků *Button*, tedy tlačítek Windows Forms. Ta jsou vykreslena do kruhu v úhlu, který je vypočítán jako podíl obvodu kruhu a počtu vykreslovaných profilů. Poloměr kruhu je vypočítán jako třetina z šířky *RightPanelu*. Samotnému tlačítku reprezentujícímu profil je pak přiřazena vypočtená pozice na panelu, nastavena velikost, náhodná barva a jako text je mu nastaveno jméno profilu. Dále je objekt tlačítka přidán do slovníku *Dictionary<Button, profiles>* map. Ten slouží pro funkci, kdy je po stisknutí tlačítka zobrazen informační panel o profilu (je předán jako parametr konstruktoru formuláře *ProfileInfo).*

Vazby, které jsou reprezentovány jako přímky mezi jednotlivými tlačítky reprezentujícími příslušné profily, jsou tvořeny pomocí třídy *Graphics*, konkrétně metodou *DrawLine()*. K samotnému vykreslení všech přímek byla implementována metoda *DrawLinks()*.

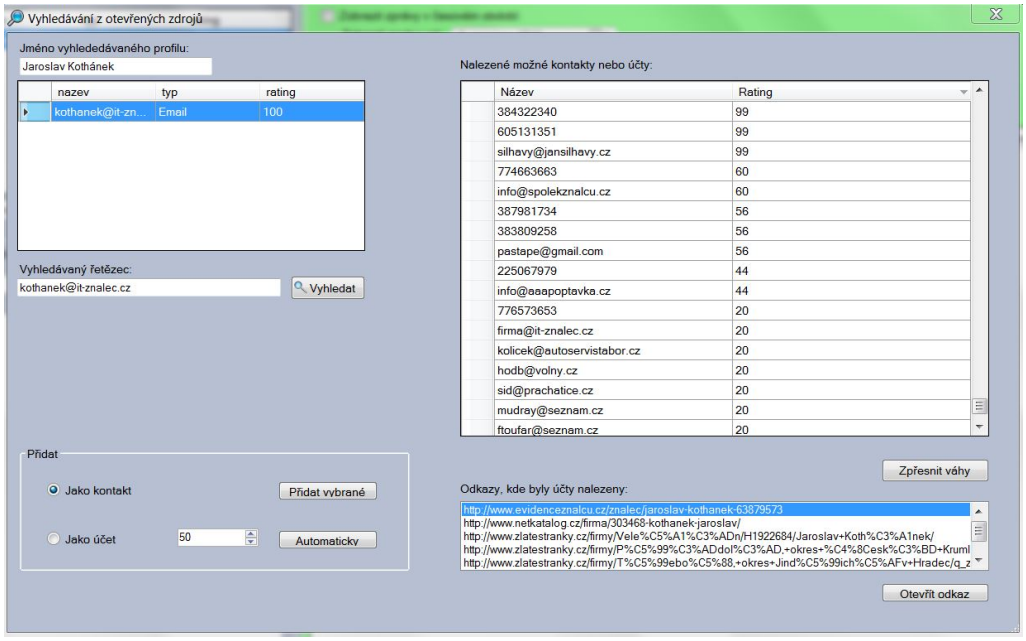
```
private void DrawLinks(PaintEventArgs e)
{
    Graphics graphics = e.Graphics;
    graphics.Clear(Color.White);
    foreach (Relation item in relations)
    {
      Color color = getRandomDarkColor(); // náhodná tmavá barva
      Pen p = new Pen(color);
      p.Width = 3;
// Body pro vykreslení. Map Value je rovna  profilX v relations - klíč je
//button - získána jeho pozice

    int x1 = map.Single(x => x.Value == item.Profil1).Key.Location.X + 65;
    int y1 = map.Single(x => x.Value == item.Profil1).Key.Location.Y + 15;

    int x2 = map.Single(x => x.Value == item.Profil2).Key.Location.X + 65;
    int y2 = map.Single(x => x.Value == item.Profil2).Key.Location.Y + 15;
             // pricteno 15 nebo 65 – do stredu tlacitka
    graphics.DrawLine(p, x1, y1, x2, y2); // vykreslit

     }
}
```

Touto metodou jsou vykresleny přímky pro každou dříve vytvořenou vazbu mezi profily.

Stisknutím tlačítka *Report* dojde k otevření saveFileDialogu a po zadání cesty a potvrzení uživatelem je vytvořen report. Samotný report je vytvořen jako HTML soubor a stejnojmenná složka, ve které je uložen obrázek ve formátu JPEG. Ukázka vygenerovaného reportu je součástí příloh.

5.9 Hledání informací z otevřených zdrojů

Kapitola popisuje implementaci funkce hledání dalších možných kontaktů nebo účtů vybraného profilu. Princip a důvod pro výběr technologie je popsán v kapitole Návrh hledání dalších informací z otevřených zdrojů. Popsány jsou vybrané metody, které jsou dle autora práce nejdůležitější.



Obrázek 5.12, Grafické rozhraní funkce Hledání informací z otevřených zdrojů

V prvním kroku byla implementována třída *SearchQuery*, která používá ke zpracování webu *Html Agility Pack*. Tato třída reprezentuje dotaz položený vyhledávači Seznam.cz a obsahuje metodu *parseSeznam*.

Metoda *parseSeznam* načte *HtmlDocument doc* z URL, jejíž součástí je proměnná *searchEngine* (ta obsahuje řetězec search.seznam.cz), dále obsahuje proměnou *stringParam*, do které je pomocí konstruktoru třídy vložen hledaný řetězec. Dalšími parametry je *filetype*, který je nastaven tak, aby byly hledány pouze HTML stránky, parametr *count=20* udává maximální počet vyhledávačem nalezených odkazů. Hodnota 20 je maximální možný počet výsledků na jedné stránce. Proto je použit ještě parametr *from*, ke kterému je v cyklu přičtena hodnota 20 a jsou načteny odkazy z dalších stránek, dokud není dosaženo prahu *treshold*.

```
public void parseSeznam()
{
    int i = 0;
```

```
    HtmlNodeCollection nodes = null;
    do
    {
       // načte stránku seznamu s příslušnými parametry, která obsahuje
          nalezené výsledky - fileType:HTML count 20

      HtmlDocument doc = new HtmlWeb().Load("http://" + searchEngine +
      "/?q=" + searchParam + "+filetype%3Ahtml" + "&count=20" + "&from=" + i);
      nodes = doc.DocumentNode.SelectNodes("//div[@class='modText'][@id]");

         // kontrola - pokud neni nic nalezeno
         if (nodes == null || nodes.Count == 0)
          {
           return;
          }

         // nalezene odkazy ulozeny do listu results
         foreach (HtmlNode node in nodes)
          {
           String link = node.SelectSingleNode("./h3/a")
           .Attributes["href"].Value;
           results.Add(link);
          }
           i = i + 20;
     } while (nodes.Count == 20 && i<treshold);
}
```

Takto vrácené nalezené URL adresy stránek, které byly vyhledávačem Seznam.cz vráceny pro výraz zadaný v proměnné *searchParam*, jsou následně uloženy do listu *results*.

Výsledky uložené v listu *results* jsou dále předány třídě *WebParser* pomocí parametrů jejího konstruktoru i s vyhledávaným řetězcem. Třída WebParser pomocí metody *SearchForAccounts* načítá pomocí *Html Agility Packu* všechny webové stránky (jejich obsah), jejichž URL jsou v listu *results* a prohledává je pomocí regulárních výrazů.

```
Regex tel = new Regex(@"((\+420)? ?[0-9]{3} [0-9]{3} [0-9]{3})
|((\+420) ?[0-9]{3} ?[0-9]{3} ?[0-9]{3})");
Regex mail = new Regex(@"[a-zA-Z0-9._-]+@[a-zA-Z0-9.-]+\.[a-zA-
Z]{2,4}");
```

Pomocí těchto regulárních výrazů jsou nalezeny všechny e-mailové adresy a telefonní čísla. Telefonní číslo musí odpovídat formátu, kdy je na webové stránce zapsáno s nebo bez mezinárodní předvolby +420, ale obsahuje mezery mezi trojicemi čísel. Druhou možností je, že číslo musí obsahovat předvolbu a nemusí obsahovat mezery mezi trojicemi čísel. Takto navržený regulární výraz byl zvolen z důvodu, že v případě, kdy bylo umožněno hledání pouze devíti po sobě jdoucích čísel, docházelo k nalezení i několika stovek čísel, které v drtivé většině případů nebyly telefonními

čísly. Toto hledání probíhá ve dvou vláknech. První vlákno hledá telefonní čísla, zatímco druhé hledá e-maily.

Takto nalezené e-maily a telefonní účty jsou uloženy do *Dictionary<String, Lists<String>> urlAndAccounts*, kde klíčem je URL stránky, na které byly výrazy nalezeny, a hodnotou je pak list, obsahující na stránce nalezené e-maily a telefonní čísla. Do tohoto listu jsou uloženy nalezené e-maily a telefonní čísla pouze v případě, že je již tento list neobsahuje. Je tím zamezeno situaci, kdy je na jedné stránce telefonní číslo nebo e-mail nalezen vícekrát. V takovémto případě by totiž došlo ke zkreslení *ratingu*, který je počítaný z počtu výskytů na nalezených webových stránkách a hodnocení URL, na které byl nalezen.

Následuje spočítání *ratingu* pro každý nalezený e-mail nebo číslo pomocí metody getAccountRating. Rating je počítán dle počtu a hodnocení stránek, na kterých byl nalezen. K hodnocení stránek je k dispozici uživateli soubor sources.txt, kam si může přidat URL jednotlivých webů a jejich hodnocení od 1 do 10. Čím vyšší hodnocení URL, tím vyšší bude i rating telefonních čísel a e-mailů, které jsou na dané stránce nalezeny. Dále je možné přidat pouze výrazy s příslušným hodnocením, které by URL adresa měla obsahovat.

```
public double getAccountRating(List<String> vyskyty)
 {
     double hitThr = 5; // pocet nalezenych, kdy je rating nejvyssi
     double suma = 0; // suma ratingu jednotlivych stran

     String line = "";
     int rating = 0;
     int toReturn = 0;

     // list do ktereho nactu resources - zname a hodnocene weby
     Dictionary<String, int> fileDict = new Dictionary<string, int>();


     System.IO.StreamReader file = new
     System.IO.StreamReader("Files\\sources.txt");

            while ((line = file.ReadLine()) != null)
            {
                String url = line.Substring(0, line.IndexOf(" "));
                rating = Int32.Parse(line.Substring(line.IndexOf(" ")));

                fileDict.Add(url, rating);
            }

            // pro kazdy stranku v listu vyskyty(tam kde se ucet nachazel)
            foreach (String site in vyskyty)
            {
                 // spocitam rating jako  suma 1/5 * (rating / 3.3 +1)
                int siteRating = 0;
```

```
                // strany co jsou v souboru resources
                foreach (String key in fileDict.Keys)
                {
                    //!!!
                    if (site.Contains(key))
                    {
                     siteRating = fileDict[key]; // kdyz tam stranka je
                         rating je rating ze souboru resources
                        // siteRating = fileDict[key]
                        break;
                    }
                    else
                    {
                        siteRating = 0; // jinak rating 0
                    }
                }
                    double y = (siteRating / 3.33) + 1; // suma dle vzorce
                    double podilNalezenych = 1 / hitThr;
                    double dilciVysledek = podilNalezenych * y;
                    suma = suma + dilciVysledek;

            }

            suma = suma * 100; //  pro rating od 0 do 100
            if (suma > 99) suma = 99; // maximalni hodnota ratingu
            toReturn = Int32.Parse(Math.Round(suma,0).ToString());
                                    // zaokrouhleni
        return toReturn;
}
```

Po spočítání ratingů následuje zobrazení všech nalezených účtů (telefonních čísel a e-mailů) s příslušným ratingem na formuláři *AccountSearch*.

Všechny nalezené účty jsou uživateli zobrazeny s vypočítanými ratingy. Tyto účty pak může uživatel přidat hledanému jako kontakt nebo jako vlastní účet. Výběr v tomto případě nechává autor na uživateli. Dále má uživatel možnost přidat jako kontakt nebo účet všechny účty, jejichž hodnota ratingu je větší než zadaná hodnota. K ověření a rozhodnutí, jak naložit s vybraným účtem má uživatel k dispozici seznam stránek, na kterých byl účet nalezen. Stisknutím tlačítka *Otevřít odkaz* je vybraná stránka otevřena ve webovém prohlížeči.

Stisknutím tlačítka *Zpřesnit váhy* dojde u vybraného nalezeného účtu k novému hledání. V tomto případě je hledaným parametrem *searchParam* vybraný nalezený účet a, stejně jako v předchozím případě, jsou pro něj vyhledány stránky, na kterých je účet nalezen. Následně už nejsou hledány všechny účty, ale pouze účet, podle kterého byl nalezen nalezený účet. Je-li nalezen, je stránka přidána mezi ty, na kterých byl hledaný účet nalezen a je přepočítán rating.

5.9.1 Rizika funkce Hledání informací z otevřených zdrojů

Na základě dosažených výsledků, které byly získány během testování, lze dojít k závěru, že tato funkce přináší časovou úsporu při hledání možných účtů nebo kontaktů vybraného profilu. Je však nutné brát v potaz, že hodnota ratingu nalezeného účtu je závislá na počtu webových stránek a jejich hodnocení. Je na uživateli aplikace, aby posoudil, zda-li nalezený e-mail nebo telefonní číslo opravdu může příslušet vybranému subjektu. Uživateli aplikace je k dispozici možnost nahlédnout na jednotlivé zdroje a je na něm a jeho zkušenostech, aby posoudil a zhodnotil nalezené skutečnosti.

6. Testování

Pro zjištění a odstranění nedostatků bylo provedeno testování pomocí testovacích scénářů a následně proběhlo i uživatelské testování bez použití scénářů. Testování se účastnili jeden dobrovolník a autor. Dobrovolník nejprve použil vytvořené testovací scénáře a následně procházel a zkoušel program dle vlastního uvážení. Testovací scénáře a záznam o provedeném testování dle testovacích scénářů je k dispozici v příloze

Díky testování byly odhaleny nedostatky v importu, kdy jména některých profilů importovaných z XRY, MPEP i UFED obsahovala ve jméně „/M“, dále byly odhaleny problémy při importu z XRY, kde byla některá čísla uložena i s pomlčkami mezi trojicemi čísel, což mělo za následek, že v databázi bylo číslo uloženo dvakrát – jednou s pomlčkami, podruhé bez nich.

Další nedostatky, které byly nahlášeny, se týkaly grafického rozhraní. Zde se jednalo zejména o designové nedostatky, jako špatně zarovnané ovládací prvky či jinak zvolené barvy pozadí. Všechny zjištěné nedostatky byly autorem práce opraveny.

7. Budoucí rozvoj aplikace

V této kapitole budou v krátkosti popsána možná řešení, která by v budoucnu mohla být dále implementována.

Největší prostor pro další rozvoj aplikace spatřuje autor v rozšíření nejen importem podporovaných formátů, ale zároveň i v rozšíření portfolia podporovaných forenzních nástrojů. Zde je však nutné, alespoň po dobu vývoje, zajistit potřebné licence, aby mohly být získány výstupy z těchto nástrojů.

Další prostor pro rozvoj aplikace autor spatřuje v oblasti hledání dalších informací z internetu. Zde je jistě prostor pro využití dalších vyhledávačů. Možné je i použití specializovaných webů jako např. kdovolal.cz, které by mohly být v některých případech nápomocné pro odfiltrování některých čísel. Samostatnou kapitolou by mohlo být i vyhledávání dalších informací týkajících se nejen fyzických, ale i právnických osob, např. IČO, DIČ, či hledání v živnostenském rejstříku, rejstříku dlužníků atp.

8. Závěr

Cílem této diplomové práce byl návrh a tvorba analytického softwarového prostředku určeného k analýze komunikace mezi rozdílnými komunikačními prostředky. Tato komunikace probíhá prostřednictvím mobilních telefonů, smartphonů, tabletů a osobních počítačů. Navrhovaný softwarový analytický prostředek bude umožňovat zjištění vazeb mezi komunikujícími subjekty, musí proto analyzovat data získaná z rozdílných forenzních nástrojů. Tato data se liší jak obsahem a rozsahem, tak formátem a strukturou. Analytický prostředek umožní agregaci rozdílných dat z různých zdrojů a jejich následné zkoumání. Dále umožní hledat nové informace o dalších možných vazbách na další subjekty a následně tyto vazby zobrazí.

V teoretické části byly popsány postupy forenzní analýzy digitálních dat, metody užívané při zjišťování a dokazování trestných činů. Předmětem zkoumání je obsah komunikace více osob, která probíhá za pomoci více typů komunikační techniky. Bylo pojednáno o postupu a jednotlivých krocích této metody zkoumání digitálních dat, které se liší podle druhu užité komunikační techniky.

V analytické části jsou popsány vybrané forenzní nástroje, používané v rámci digitální forenzní analýzy. Nejprve je pojednáno o nejpoužívanějším nástroji pro zajišťování dat z osobních počítačů, dále pak o nástrojích, které se užívají k forenzní analýze digitálních dat zajištěných z mobilní komunikační techniky. U popsaných forenzních nástrojů jsou podrobně popsány jejich výstupy.

Kapitola Návrh popisuje jednotlivé funkce aplikace, tedy to, co by měla uživateli umožňovat. Zejména se jedná o import dat, jejich zobrazení a vizualizaci, a následné hledání vazeb mezi komunikujícími osobami. Dále je pojednáno o výběru vhodných softwarových nástrojů a technologií, které byly použity při implementaci vyvíjené aplikace.

Kapitola Implementace reflektuje předcházející kapitolu a popisuje implementaci jednotlivých bloků aplikace v závislosti na požadavcích definovaných v rámci jejího návrhu. V závislosti na použitých nástrojích a technologiích popisuje import dat, jejich zpracování a grafické znázornění. Zaměřuje se na nejdůležitější funkce aplikace, zejména na možnost vizualizace vazeb mezi subjekty komunikace. Je pojednáno též o možnosti hledání informací z otevřených zdrojů.

V návaznosti na návrh a implementaci byla navržená aplikace testována za účelem zjištění a odstranění nedostatků. Testování se účastnil, vyjma autora, i jeden dobrovolník a probíhalo podle vytvořených scénářů, tak dle uvážení dobrovolníka. Scénáře zahrnovaly vytvoření případu, import souborů a následné vyzkoušení funkcí aplikace. Pozornost byla věnována zejména vizualizaci vazeb mezi komunikujícími subjekty a hledání informací z otevřených zdrojů. Na základě průběhu testování byly autorem provedeny úpravy týkající se importu dat a přehlednosti a funkčnosti grafického rozhraní.

Výsledkem této práce je funkční aplikace, která umožňuje v rámci forenzní analýzy digitálních dat použít novou kombinaci funkcí. Jde o funkce, které nejsou dostupné při použití jednotlivých, běžně používaných forenzních nástrojů. Vyvinutá aplikace umožňuje import výstupů z více zdrojů. Dále umožňuje nalézt další možné kontakty nebo účty vybraných komunikačních technologií, které mohly být použity v rámci komunikace mezi vybranými subjekty. Na závěr umožňuje zobrazit vazby mezi jednotlivými osobami.

9. Literatura

[1] SVETLÍK, Marián. Digitální forenzní analýza a bezpečnost informací. *Digital Security Magazine*. 2010, č. 1, s. 4. [cit. 2015-02-02]. Dostupné z: http://www.rac.cz/RAC/homepage.nsf/CZ/Clanky/$FILE/DSMDigit%C3%A1ln%C3%AD%20forenzn%C3%AD%20anal%C3%BDza-01-2010.pdf

[2] CALOYANNIDES, Michael A. *Computer forensics and privacy*. Boston, MA: Artech House, 2001, xvii, 392 p. ISBN 15-805-3283-7.

[3] FORMÁNEK, Martin. *Forenzní analýza digitálních nosičů dat pro počítače*. Praha, 2008. Dostupné z: https://dip.felk.cvut.cz/browse/pdfcache/formam1_2008bach.pdf. Bakalářská práce. ČVUT. Vedoucí práce Doc. Ing. Róbert Lórencz, CSc.

[4] KADLEC, Josef. *Forenzní analýza unixových systémů*. Hradec Králové, 2006. Dostupné z: http://i.iinfo.cz/files/root/k/Digitalni_forensni_analyza_unixovych_systemu.pdf. Diplomová práce. Univerzita Hradec Králové. Vedoucí práce Ing. Miloslav Feltl.

[5] ACCESSDATA. *Forensic Toolkit (FTK)* [online]. [cit. 2015-02-12]. Dostupné z: http://accessdata.com/solutions/digital-forensics/forensic-toolkit-ftk

[6] CELLEBRITE. *UFED Pro Series* [online]. [cit. 2015-02-16]. Dostupné z: http://www.cellebrite.com/Mobile-Forensics/Solutions/ufed-pro-series

[7] ACCESSDATA. *Mobile Phone Examiner Plus* [online]. [cit. 2015-02-16]. Dostupné z: http://accessdata.com/solutions/digital-forensics/mpe

[8] MSAB. *What is XRY?* [online]. [cit. 2015-02-20]. Dostupné z: https://www.msab.com/products/xry/

[9] NAGEL, Christian. *Professional C# 2005*. Indianapolis, IN: Wiley, c2006, xlvi, 1540 p. ISBN 978-076-4575-341.

[10] HORVÁTH, Tomáš. .*NET Framework* [online]. [cit. 2015-02-22]. Dostupné z: http://programujte.com/clanek/2008120700-net-framework/

[11] ZOBEC, Michal. Omezení Microsoft SQL Server 2012 Express. [online]. [cit. 2015-02-23]. Dostupné z: http://www.michalzobec.cz/omezeni-microsoft-sql-server-2012-express-2630

[12] DB-ENGINES. *DB-Engines-ranking* [online]. [cit. 2015-02-24]. Dostupné z: http://db-engines.com/en/ranking

[13] *Policista*. Praha: Ministerstvo vnitra, 2003, roč. 2003, č. 7. ISSN 1211-7943.

[14] VACHTOVÁ, Jitka. *Databáze zdarma pro komerční použití*. [online]. 3.7.2014. [cit. 2015-04-16]. Dostupné z: http://www.vachtova.cz/article/show/248

[15] LINQ (Language-Integrated Query). *MSDN - Microsoft Developer Network* [online]. [cit. 2015-04-16]. Dostupné z: https://msdn.microsoft.com/cs-cz/library/bb397926.aspx

[16] Entity Framework. *MSDN - Microsoft Developer Network* [online]. [cit. 2015-04-16]. Dostupné z: https://msdn.microsoft.com/en-us/data/ef.aspx

10. Seznam použitých obrázků

11. Seznam použitých zkratek

- CRUD - Create, Read, Update, Delete
- CLR - Common Language Runtime
- DFA - Digitální Forenzní Analýza
- DTD - Document Type Definition
- EF- Entity Framework
- EXIF- Exchangeable Image File Format
- ICCID - Integrated Circuit Card Identifier
- IMEI - International Mobile Equipment Identity
- IMSI - International Mobile Subscriber Identity
- LINQ - Language Integrated Query
- MAC - Media Access Control
- MPEP - Mobile Phone Examiner Plus
- MSIL - Microsoft Intermediate Language
- ORM - Object-Relational Mapping
- RDBMS - Relational Database Management systém
- SIM - Subscriber Identity Module
- SMS - Short Message Service
- SŘBD - Systém Řízení Báze Dat = RDBMS
- SQL - Structured Query Language
- URL - Unique Resource Locator
- WF - Windows Forms
- WPF - Windows Presentation Foundation
- XML -Extensible Markup Language

12. Přílohy

Příloha A

Instalační příručka

1) Instalace XAMPP.
2) Zabezpečení XAMPP.
   a) Nastavení přihlašovacích údajů Apache.
   b) Nastavení hesla MySQL, uživatelské jméno je defaultně *root.* Heslo MySQL zadává uživatel při spuštění aplikace.
3) Přihlášení se do phpMyAdmin a import databáze Vizualizace ze souboru na přiloženém DVD.
4) Instalace DipCom.
5) Spuštění aplikace pomocí DipCom.exe nebo ikony na ploše.

Systémové požadavky

- OS Windows 7/8.1/10
- .Net Framework 4.5 – V případě, že není nainstalován, dojde k jeho instalaci během instalace programu DipCom (je nutné připojení k internetu).
- Rozlišení obrazovky 1366x768 nebo 1920x1080

Příloha B

Uživatelská příručka

1. Přihlášení

Po prvním spuštění zadá uživatel následující údaje pro připojení k databázi a následně stiskne tlačítko *Login*.

- Adresa serveru - Při použití XAMPP dle instalační příručky je adresa serveru *localhost.*
- Port – defaultní port při použití XAMPP je 3306.
- Uživatel – Při použití XAMPP dle instalační příručky je defaultní uživatelské jméno *root.*
- Heslo – Uživatelem zvolené heslo do MySQL. V případě, že po instalaci XAMPP uživatel heslo pro MySQL nezadal, toto pole zůstane prázdné.

Při správném zadání je vytvořeno připojení a přihlašovací údaje jsou uloženy do souboru ve složce uživatele, do složky Appdata\Local\DipCom\Files. Následně je uživateli zobrazeno okno pro správu případů.

2. Správa případů

Zde má uživatel možnost:

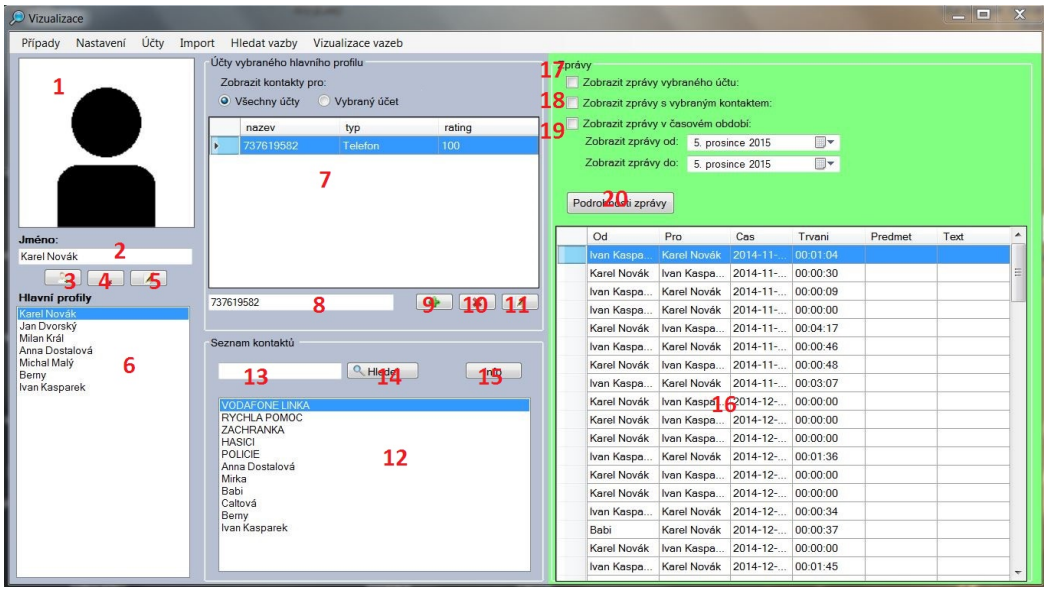
- Vytvořit případ
    - Stisknout *Vytvořit*
    - Vyplnit informace o případu
    - Potvrdit stisknutím *Uložit*

Po výběru ze seznamu případů:

- Změnit vybraný případ
    - Stisknout *Změnit*
    - Možnost přejmenování, změna evidenčního čísla a popisu
    - Potvrdit stisknutím *Uložit*
- Smazat vybraný případ
    - Stisknout *Smazat*
    - Potvrdit smazání

Tlačítkem *Vybrat* je zobrazeno hlavní okno aplikace.

3. Hlavní okno



V levém sloupci se nachází seznam všech hlavních profilů případu (6). Vybráním hlavního profilu je zobrazena jeho fotografie (1), je-li dostupná. Dále jsou v prostředním sloupci zobrazena telefonního čísla a e-maily vybraného profilu (7). Pod nimi pak kontakty těchto telefonních čísel a emailů (12).

Dvojitým kliknutím na obrázek se otevře dialogové okno, které slouží pro výběr a následné přidání fotografie profilu. Dále jsou pod jménem vybraného profilu tlačítka pro vytvoření, smazání a přejmenování profilu.

Stisknutím tlačítka pro vytvoření profilu (3), je zobrazeno okno pro vytvoření hlavního profilu. V něm je uživatel vyzván k zadání jména, popř. fotografie. Po stisknutí tlačítka *Další* je uživatel vyzván k zadání telefonního čísla nebo e-mailu, který přísluší vytvářenému profilu. Vytvářenému profilu je třeba přidat alespoň jeden e-mail nebo telefonní číslo.

Po stisknutí tlačítka pro smazání profilu (4) je uživatel dotázán, zda-li opravdu chce smazat profil vybraný v seznamu hlavních profilů (6). Po potvrzení je profil smazán. Pro přejmenování profilu vybraného v seznamu hlavních profilů (6) stačí, aby uživatel přepsal jméno v textovém poli (2), které při editaci změní barvu, a následně stiskl tlačítko pro přejmenování (5). Po potvrzení je profil přejmenován.

Výše vypsané možnosti jsou k dispozici i po stisknutí pravého tlačítka nad profilem vybraným v seznamu hlavních profilů (6).

V seznamu *Účty vybraného hlavního profilu* (7) jsou zobrazeny účty (e-maily a telefonní čísla) hlavního profilu včetně typu a ratingu. Dle výběru *Zobrazit kontakty a zprávy pro*:

- **Všechny účty**
- **Vybraný účet**

jsou zobrazeny buď všechny kontakty a následně i zprávy všech účtů, nebo jen vybraného účtu.

Stejně jako pro úpravu profilů jsou zde stejná tlačítka pro přidání (9), smazání (10) a přejmenování (11) účtu vybraného hlavního profilu (1,2,6). Pro přidání dalšího účtu (telefonního čísla nebo e-mailu) smaže uživatel obsah textového pole (8) a napíše do něj příslušný e-mail nebo telefonní číslo. Po stisknutí tlačítka pro přidání účtu (9) je účet přidán.

Stisknutím tlačítka pro odstranění účtu (10) a následném potvrzení je účet vybraný účet (7, 8) smazán. Přejmenování probíhá stejně jako v případě profilu. Vybraný účet je přepsán v textovém poli (8) a přejmenování je potvrzeno tlačítkem pro přejmenování (11).

Vyplněním části nebo celého jména do textového pole (13) a stisknutím tlačítka *Hledej* (14), jsou v seznamu (12) zobrazeny pouze kontakty, jejichž jméno odpovídá zadanému řetězci. Vymazáním řádku pro hledání a následným stiskem tlačítka *Hledej* jsou opět zobrazeny všechny kontakty. Stisknutím tlačítka *Info* (15) nebo *dvojitým klinutím*, popřípadě výběrem v kontextovém menu po stisku pravého tlačítka na jméno kontaktu je zobrazeno okno s podrobnostmi profilu.

V seznamu *Zprávy* (16) jsou zobrazeny zprávy (zprávy, hovory a e-maily) vybraného hlavního profilu (1, 2, 6). Je zde také možnost filtrovat zobrazené zprávy pomocí filtrů, které je možné kombinovat (17, 18, 19).

- **Zobrazit zprávy vybraného účtu** – Jsou zobrazeny pouze zprávy vybraného účtu ze seznamu účtů hlavního profilu.
- **Zobrazit zprávy vybraného kontaktu** – Jsou zobrazeny zprávy hlavního profilu pouze s vybraným kontaktem ze seznamu kontaktů.

- **Zobrazit zprávy v časovém období** – Jsou zobrazeny zprávy pouze ve vybraném časovém období.

Stiskem tlačítka *Podrobnosti zprávy* (20) nebo *dvojitým kliknutím* na vybraný řádek (zprávu) je zobrazeno okno se všemi informacemi o zprávě. K dispozici je i možnost zobrazení html (jako v prohlížeči) pro případ, je-li zpráva e-mail a obsahuje html kód.

Hlavní menu

Případy

Otevře okno pro Správu případů.

Účty

Otevře okno *Správa účtů*. Zde má uživatel možnost změnit majitele účtu. Tato funkcionalita je zde pro případ, dojde-li k záměně profilu majitele účtu.

Nastavení – Přihlášení

Otevře okno pro přihlášení do databáze.

Import

Otevře okno *Import*, kde je možné importovat výstupní soubory z forenzních nástrojů.

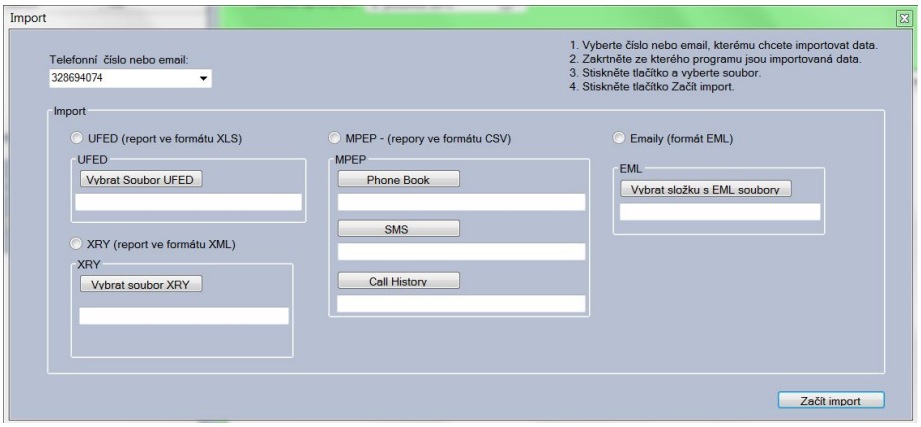
Vizualizace vazeb

Otevře okno *Vizualizace vazeb*.

Hledat vazby

Otevře okno *Hledání informací z otevřených zdrojů* pro vybraný hlavní profil.

4. Import



Import probíhá vždy pro vybraný hlavní profil a jeho vybraný účet. Uživatel na *hlavním panelu* v *seznamu hlavních profilů* vybere profil a po stisknutí tlačítka *Import v hlavním menu* nebo po výběru položky *Import z kontextového menu*, které je zobrazeno po stisknutí pravého tlačítka myši nad vybraným hlavním profilem, je uživateli zobrazeno okno pro import.

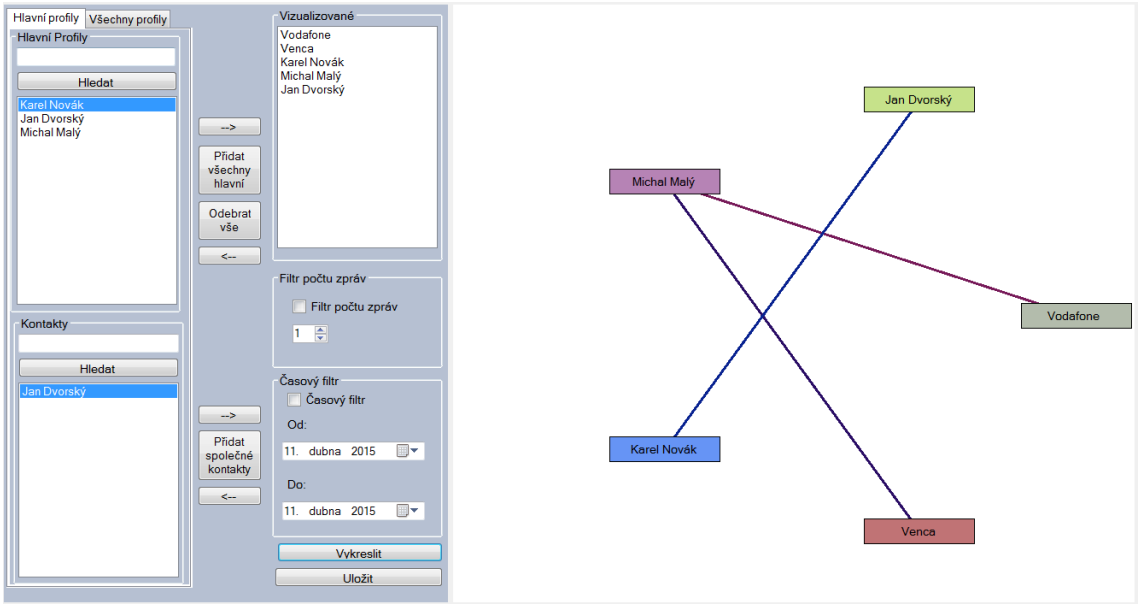
Zde v horním pravém rohu může vybrat jeden z uživatelových účtů, kterému budou importována data. Dále uživatel zaškrtne typ souboru, ze kterého bude probíhat import, a stisknutím příslušného tlačítka zadá cestu souboru.

Po výběru souboru uživatel stiskne tlačítko *Začít import*, tím začne samotný import. O jeho průběhu je uživatel informován pomocí progressBaru. Po skončení importu se okno samo zavře a uživatel na hlavním panelu může prohlížet importovaná data.

Možnost nastavení

Uživatel má možnost pro soubory XLS z nástroje UFED a pro soubory CSV z nástroje MPEP reagovat na menší změny ve strukturách generovaných reportů. Je možné vybrat sloupce, ze kterých jsou importována data, a zároveň řádek, od kterého začne import. Toto lze nastavit změnou souborů ufedconfig.txt a mpepconfig.txt. Je nutné, aby struktura souborů zůstala zachována.

5. Vizualizace Vazeb



Výběrem položky *Hlavní profily* jsou zobrazeny hlavní profily a po výběru jednotlivého hlavního profilu jsou zobrazeny jeho kontakty. Výběrem *Všechny profily* je zobrazen seznam všech profilů případu (hlavní profily a profily jejich kontaktů).

Pomocí tlačítka se *šipkou* směrem k seznamu *Vizualizované* jsou do seznamu přidávány vybrané profily, popřípadě kontakty hlavních profilů. Tlačítkem *Přidat všechny hlavní* jsou do seznamu *Vizualizované* přidány všechny hlavní profily. Tlačítkem *Odebrat vše* je celý seznam *Vizualizované* smazán. Tlačítkem *Přidat společné kontakty* jsou přidány všechny kontakty společné pro profily v seznamu *Vizualizované.*

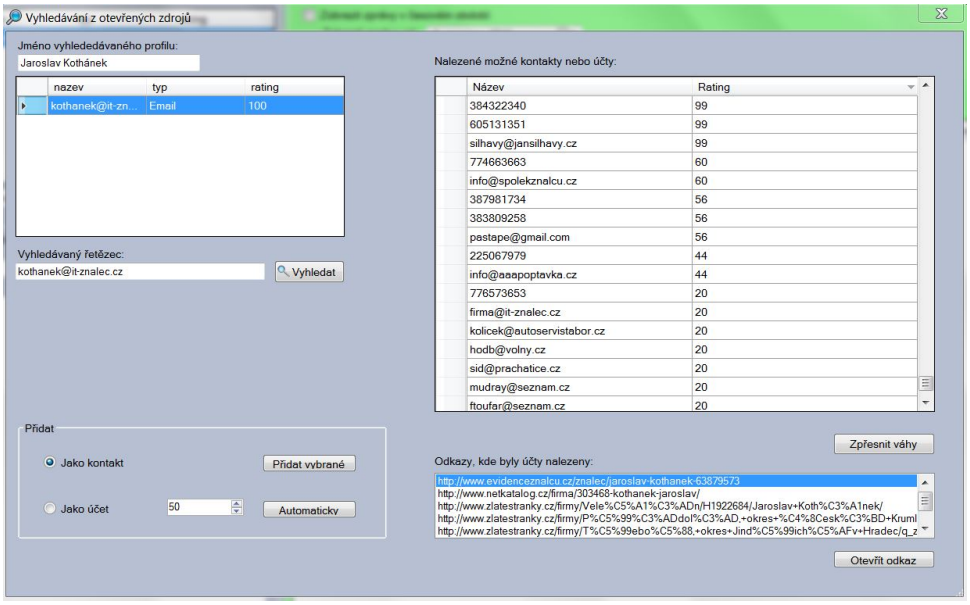
Seznam *Vizualizované* je seznam všech profilů, mezi kterými budou zjišťovány a zobrazeny vazby. Pokud nejsou zaškrtnuty filtry, je vazba vytvořena jak v případě, kdy mezi profily došlo ke komunikaci, tak v případě, že byl profil v seznamu kontaktů druhého profilu. Je-li zaškrtnutý jeden nebo oba filtry, je vazba vytvořena jen na základě proběhlé komunikace, a to v případě, jsou-li splněny podmínky filtru.

- **Časový filtr** – Vazba je vytvořena jen v případě, došlo-li ke komunikaci v zadaném časovém období.

- **Filtr počtu zpráv** - Vazba je vytvořena jen v případě, došlo-li ke komunikaci, kdy počet zpráv (hovorů, zpráv, e-mailů) byl roven nebo přesáhl nastavenou hodnotu.

Stisknutím tlačítka *Vykreslit* dojde k vykreslení vazeb, které splňují podmínky filtrů, mezi vybranými profily seznamu Vizualizované. Kliknutím na vykreslený *obdélník* se jménem profilu je možné zobrazit informace o příslušném profilu. Stiskem tlačítka *Report*, je uživatel vyzván k nastavení cesty pro uložení reportu ve formátu html, který obsahuje obrázek s vykreslenými vazbami, dále pak vazby mezi jednotlivými profily a záznamy o komunikaci, která mezi nimi proběhla.

6. Hledání informací z otevřených zdrojů



Výběrem hlavního profilu v *Hlavním okně* a stisknutím tlačítka *Hledání* v hlavním menu aplikace se zobrazí okno *Hledání informací z otevřených zdrojů*. To slouží k hledání dalších možných kontaktů nebo účtů vybraného hlavního profilu.

V levém horním rohu je zobrazeno jméno vybraného hlavního profilu a jeho účty. Kliknutím na jméno je do řádku *Vyhledat* vloženo jméno profilu, kliknutím na účet v seznamu účtů profilu je do řádku *Vyhledat* vložen vybraný účet (telefonní číslo nebo e-mail). Do řádku *Vyhledat* může uživatel vložit i jakýkoliv výraz, který by mohl souviset s vybraným profilem, například přezdívku atp. Po stisknutí *Vyhledat* jsou pro vyhledaný výraz nalezeny stránky, které tento výraz obsahují a následně jsou uživateli

zobrazeny všechny e-maily a telefonní čísla, která byla na těchto stránkách nalezena, a to včetně *Ratingu*. Čím vyšší je číslo ratingu, tím vyšší je pravděpodobnost souvislosti mezi hledaným výrazem a nalezeným účtem.

Po výběru jednotlivého nalezeného telefonního čísla nebo e-mailu je uživateli k dispozici seznam webových stránek, na kterých byly nalezeny. Tlačítkem *Otevřít odkaz* je vybraná webová stránka otevřena ve výchozím prohlížeči uživatele.

Dále jsou uživateli k dispozici tlačítka *Přidat vybrané* a *Automaticky*, a to v kombinaci s možnostmi *Jako kontakt* a *Jako účet.*

- **Jako kontakt**- Telefonní číslo nebo e-mail je přidán jako kontakt hlavního profilu.
- **Jako účet -** Telefonní číslo nebo e-mail je přidán jako účet hlavního profilu.

Výběrem jednoho či více (pomocí klávesy *Ctrl*) nalezených telefonních čísel nebo e-mailů a stiskem tlačítka *Přidat vybrané* jsou vybrané účty přiřazeny hlavnímu profilu, a to výše vybraným způsobem.

Stisknutím tlačítka *Automaticky* jsou přidána všechna telefonní čísla a e-maily, jejichž rating je roven nebo větší něž zvolená hodnota. Všechna čísla nebo e-maily splňující podmínku jsou přidána buď jako kontakt, nebo jako účet, dle výběru uživatele.

Možnost nastavení

Pro uživatele existuje možnost nastavit si rating jednotlivých webových stránek (URL), popřípadě výrazů, které daná URL obsahuje. Nastavení je možné přidáním do souboru sources.txt v adresáři Files, který se nachází v adresáři programu. Je nutné aby struktura souboru zůstala zachována.

Příloha C

Struktura a popis výstupů z forenzních nástrojů

1. UFED

Samotná závěrečná zpráva je strukturována do několika kapitol. Každá kapitola obsahuje určitý okruh informací. Při samotném vytváření závěrečné zprávy bylo zvoleno takové nastavení, aby obsahovala kompletní informace, které je schopen UFED Physical Analyzer z připojeného zařízení získat. Jak bylo uvedeno, samotná zpráva je exportována do jednoho nebo více z volitelných formátů. Nezávisle na zvoleném formátu je struktura kapitol následující:

- **Souhrn** – Obsahuje informace o samotném zajišťování (extrakci) mobilního zařízení, jako je výrobce zařízení, datum začátku a konce extrakce, informace o případu, jméno vyšetřovatele a místo extrakce (jsou-li vyplněny) a identifikátor zařízení UFED.
- **Informace o zařízení –** Obsahuje informace o zajišťovaném zařízení, jako je Android ID, země a jazyk, čas aktivace telefonu, čísla ICCID a IMSI a časovou zónu.
- **Podrobnosti o datech hash obrázku –** Název této kapitoly je zřejmě mylně přeložen, obsahuje informace v zajištěném obraze – bitové kopii.
- **Zásuvné moduly** – Obsahuje informace o zásuvných modulech a jejich verzích, které byly použity pro extrakci – byly nahrány do telefonu, aby pomocí nich byla získána data.
- **Obsah** – Jedná se o obsah samotné zprávy, který zobrazuje počty nalezených položek, a to včetně počtu nalezených smazaných položek.
- **Bezdrátové sítě -** Obsahuje informace o wifi sítích, ke kterým byl telefon připojován. V tabulce jsou SSID, režimy zabezpečení (např .WPA-PSK) a data posledních připojení k sítím.
- **Cookies** – Obsahuje informace o cookies z webových prohlížečů jako je jméno, hodnota, doména, čas vytvoření a vypršení.

- **Časová osa** - Jedná se většinou o nejrozsáhlejší kapitolu. Obsahuje tabulku, ve které je chronologický přehled všech událostí, např. o příjmu zpráv, informace o voláních, e-mailech atd.
- **E-maily** - Obsahuje informace o e-mailových zprávách nalezených v zařízení. V tabulce jsou zaznamenány: časové razítko zprávy, adresa příjemce a odesílatele, předmět a text zprávy, priority a kontrolní součet MD5 zprávy.
- **Hesla** – Obsahuje informace o heslech v zařízení. Při vytváření výsledného reportu však tabulka s hesly obsahovala ve „správné“ nehashované podobě pouze heslo k WiFi. Pro účet Google tabulka obsahovala hash hesla, ačkoliv v programu Physical Analyzer bylo heslo nalezeno v nehashované podobě.
- **Historie webu** – Obsahuje informace o historii z webových prohlížečů. Tabulka obsahuje název webu, URL, datum návštěvy, počet návštěv a zdroj – tedy prohlížeč, ze kterého byla stránka navštívena.
- **Hledané položky** – Obsahuje informace o hledaných položkách a čase hledání v aplikacích jako je Youtube nebo Play Market (v případě OS Android).
- **Chaty** - V případě použití aplikací jako Skype, Hangouts nebo Faceboook obsahuje informace o komunikaci včetně textů zpráv a dat odeslání nebo příjmu.
- **Instalované aplikace –** V případě chytrých telefonů obsahuje informace o nainstalovaných aplikacích. Tabulka obsahuje jméno, verzi, čas instalace aplikace. V případě OS Android obsahuje také informace o povoleních, kterými aplikace disponuje (např. přístup k paměti, účtům, polohám atd.).
- **Kontakty** – Obsahuje informace o seznamu kontaktů v zařízení. Tabulka obsahuje jméno, typ kontaktu, telefonní číslo, e-mail nebo jiný název účtu a informaci, zda byl kontakt smazán v případě, je-li takový nalezen.
- **MMS zprávy –** Obsahuje informace o MMS zprávách v zařízení. Tabulka obsahuje časové razítko, příjemce nebo odesílatele, předmět zprávy a odkaz na přílohu (obrázek). Dále také obsahuje kontrolní součet MD5.
- **GPS** – Obsahuje obrázek – mapu s vyznačenými polohami, kde se zařízení nacházelo.
- **Polohy** - Obsahuje tabulku s informacemi o jednotlivých GPS polohách zařízení. Ta obsahuje souřadnice a čas, kdy se zařízení nacházelo na příslušných

souřadnicích. Další podstatnou informací je jméno aplikace, která daný záznam pořídila.

- **Slovník uživatele –** Obsahuje slova, která byla přidána do uživatelova slovníku**.**
- **SMS zprávy** – Obsahuje informace o SMS zprávách v zajištěném zařízení. Tabulka obsahuje informace, zda byla zpráva příchozí či odeslaná, čas příjmu nebo odeslání a samotný text zprávy. Dále je zde informace, zda se jedná o smazanou zprávu.
- **Uživatelské účty –** Obsahuje informace o dalších uživatelských účtech, které byly v zařízení nalezeny, popřípadě smazány. Tabulka obsahuje název účtu a jeho případný identifikátor.
- **Webové záložky –** Obsahuje informace o záložkách uložených ve webových prohlížečích. Tabulka obsahuje název webu, URL, počet návštěv a datum poslední návštěvy. Dále pak čas vytvoření a informaci, zda byla smazána.
- **Záznamy o hovorech** – Obsahuje informace o proběhlých hovorech. Tabulka obsahuje informaci, zda byl hovor příchozí nebo odchozí, číslo a popřípadě i jméno volaného respektive volajícího, čas hovoru, délku hovoru a informaci, zda byl záznam o hovoru smazán, byl-li nalezen smazaný.
- **Záznamy v kalendáři –** Obsahuje informace o jednotlivých záznamech v kalendářích. Tabulka obsahuje informace o čase zahájení a ukončení události, její předmět, místo, podrobnosti a e-maily dalších účastníků. Záznam *Kategorie* určuje, zda se jedná o účet kalendáře pouze v telefonu nebo např. služby Google. Dále jsou zobrazeny informace o připomenutí a intervalu jeho opakování.
- **Datové soubory** – Kapitola je rozdělena do čtyř podkapitol (Aplikace, Databáze, Dokumenty a Text). Obsahují informace o souborech aplikací a jejich případných databázích (včetně počtu řádků), dále pak informace o textových souborech uložených v zařízení. Zde se jedná o seznam souborů s příponami XML, TXT, DOC, atd.
- **Analytika aktivit –** Obsahuje seznam všech kontaktů a použitých čísel. K nim jsou zobrazeny informace o počtu událostí. Událostí je v tomto případě rozuměn kontakt – volání, SMS, e-mail atp.

- **Analytika telefony –** Obsahuje seznam SMS a volání u všech kontaktů. Tabulka obsahuje informace o hovorech a SMS zprávách s jednotlivým kontaktem. Jsou zde zaznamenány celkové i dílčí počty volání (příchozích, odchozích, zmeškaných), totéž platí i pro SMS zprávy.
- **Analytika e-maily -** Obsahuje seznam e-mailů u všech kontaktů. Tabulka obsahuje informace o počtech e-mailů s jednotlivými kontakty**.**
- **Skype** – Obsahuje informace o kontaktech účtu Skype a počtu hovorů a zpráv.

2. Mobile Phone Examiner Plus

Struktura samotné zprávy, kdy byly vybrány všechny kategorie dat, vypadá pro zkušební zařízení následovně:

- **Investigator Information** – Obsahuje informace o vyšetřovateli, popřípadě o případu, které byly vyplněny vyšetřovatelem.
- **Device Information** - Obsahuje informace o zkoumaném zařízení. Jedná se o číslo verze firmwaru, informace o SIM, výrobci, číslo IMEI, typ základní desky, časovou zónu, jméno operátora atd.
- **Bookmarks -** Obsahuje informace o záložkách z webových prohlížečů. Na rozdíl od UFED nejsou k dispozici data o počtu návštěv a datu poslední návštěvy.
- **Call History** – Obsahuje informace o uskutečněných hovorech. Pro každé volání obsahuje tabulku, kde je zaznamenána délka hovoru, čas hovoru, jméno kontaktu a jeho číslo.
- **Contact -** Obsahuje informace o jednotlivých kontaktech v seznamu kontaktů. Pro každý kontakt obsahuje vlastní tabulku s typem, e-mailem, telefonním číslem, jménem, datem posledního kontaktu.
- **Media** – Obsahuje informace ke všem mediálním souborům (audio soubory, video soubory a obrazové soubory). V této kapitole jsou však informace pouze o umístění souboru, jeho jménu a velikosti. Podrobné informace k jednotlivým souborům jsou pak v následujících kapitolách (Audio Files, Images, Video Files).

- **MMS** - Obsahuje informace o odeslaných a přijatých MMS zprávách. Pro každou MMS závěrečná zpráva obsahuje vlastní tabulku s informací, zda byla MMS přečtena, informace o odesílateli nebo příjemci, text zprávy a čas přijetí nebo odeslání.
- **Text Messages** - Obsahuje informace o textových správách SMS. Tak jako ve všech kapitolách i zde je pro každou jednotlivou SMS jedna tabulka obsahující telefonní číslo příjemce nebo odesílatele. Bohužel, k telefonnímu číslu není zobrazeno jméno i pokud je uloženo v seznamu kontaktů, což nepřispívá k přehlednosti závěrečné zprávy. Dále tabulka obsahuje informaci o tom, zda byla příchozí zpráva přečtena, čas přijetí respektive odeslání a samotný text zprávy.
- **Andorid Packages** – V případě operačního systému Android obsahuje informace o jednotlivých aplikacích (balíčcích), které jsou přítomny v zajišťovaném telefonu. Tabulky obsahují informace o jménu a verzi balíčku, jménu aplikace a verzi kódu.
- **WiFi Hotspots** – Obsahuje tabulku pro každou WiFi síť, která je uložena v zařízení. Na rozdíl od UFED nejsou k dispozici hesla ani data připojení. Tabulky obsahují pouze status (je-li povolené připojení), SSID a prioritu připojení.
- **Audio Files** – Kapitola obsahuje podrobné informace o jednotlivých audio souborech, které byly nalezeny v zařízení. Zde je obsah tabulek, které jsou vytvořeny pro každý soubor zvlášť, poměrně podrobný. Lze nalézt informace o velikosti, typu a umístění souboru. Dále pak délku skladby, název alba a skladby, autora a rok vydání. Pro každý soubor tabulka obsahuje kontrolní součty SHA-256 a MD5.
- **Images** - Kapitola obsahuje podrobné informace o jednotlivých obrazových souborech, které byly nalezeny v zařízení, následně je zobrazen i samotný obrázek. Tabulky obsahují informace o názvu, velikosti a datu přidání jednotlivého souboru, cestu k souboru v zařízení, popřípadě popis – je-li k dispozici. K dispozici jsou i kontrolní součty SHA-256 a MD5 pro jednotlivé soubory.

- **Video Files -** Obsahuje informace o video souborech v zařízení. Opět je pro každý soubor vytvořena vlastní tabulka, která obsahuje informace o datu vytvoření respektive přidání, cestu k souboru, velikost a typ souboru. Dále pak délku a rozlišení videa. Posledními informacemi jsou pak kontrolní součty MD5 a SHA-256.

3. XRY

Závěrečný report je složen z následujících kapitol (opět byla zvolena možnost, aby obsahoval kompletní dostupné informace):

- **Souhrn** – Kapitola obsahuje souhrn informací o zajišťování – datum extrakce, verzi XRY a informace o vyšetřovateli a případu. Dále obsahuje tabulku s počty zjištěných informací – počet kontaktů, zpráv, záznamů v kalendáři a počet nalezených souborů.
- **Údaje** – Kapitola obsahuje informace o případu a vyšetřovateli – v případě, že byly vyplněny.
- **Zařízení** – Tato kapitola je rozdělena do čtyř podkapitol:
    - **Základní informace** – Obsahuje informace o zařízení jako je jméno výrobce a typ telefonu, čísla IMSI , IMEI a ICCID, jméno operátora a MAC adresu WiFi modulu.
    - **Protokol událostí** – Obsahuje záznam o datu nabootování zařízení.
    - **Installed Apps** – Kapitola obsahuje záznam o aplikacích nainstalovaných v zařízení a zda jsou umístěny v zařízení nebo na paměťové kartě.
    - **Paměť klávesnice** - Obsahuje seznam slov přidaných uživatelem do slovníku.
- **Kontakty –** Obsahuje seznam kontaktů s jejich telefonními čísly nebo e-maily, dále je k dispozici informace o místě, kde byl kontakt uložen.
- **Volání –** Obsahuje informace o uskutečněných hovorech včetně času a trvání, dále pak o kontaktu, byl-li uložen v seznamu kontaktů.
- **Kalendář událostí -** Obsahuje informace o záznamech v kalendáři (předmět, umístění, od kdy do kdy událost trvala a její umístění v paměti).

- **Zprávy –** Kapitola je rozdělena do dvou podkapitol a obsahuje informace o zprávách (časy odeslání nebo přijetí, text zprávy a informace o kontaktu).
    - **SMS**
    - **MMS**
- **Web -** Kapitola osahuje podkapitolu Záložky. U nich lze najít webovou adresu, jméno a počítadlo přístupů.
- **Soubory**
    - **Obrázky –** Kapitola obsahuje informace o všech obrazových souborech v zařízení. Zajímavostí je, že součástí zprávy jsou veškerá dostupná metadata (EXIF) k jednotlivým souborům.
    - **Zvuky –** Obsahuje informace o audio souborech v zařízení, jako je jméno, velikost a typ souboru, umístění v paměti respektive na paměťové kartě.
    - **Videa** - Obsahuje informace o video souborech v zařízení. Tabulka strukturou odpovídá té, která byla v kapitole zvuky.
    - **Dokumenty –** Kapitola obsahuje seznam dokumentů, které byly v zařízení a to včetně jmen, velikostí a umístění.
    - **Archiv –** Obsahuje seznam archivů, které jsou v zařízení a informace o jejich velikosti a umístění. Byly zde zobrazeny jak archivy APK tak ZIP.
    - **Databases –** Obsahuje seznam databázových souborů v zařízení, jejich umístění a velikost. Ve zkušebním zařízení se jednalo o soubory SQLite.
    - **Nerozpoznané soubory –** Kapitola obsahuje seznam nerozpoznaných souborů.
- **XRY Sytém**
    - **Log –** Kapitola obsahuje kompletní seznam úkonů provedených při extrakci dat. Obsahuje jméno modulu, status (byl-li úkon úspěšný nebo zakázaný) a čas provedení samotného úkonu.

Příloha D

Testovací scénáře

Název	Případy
Cíl	Ověření funkcí formuláře Správa případů.
Postup	1) Vytvořit nový případ a) Vyplnit údaje o případu b) Uložit 2) Vybrat vytvořený případ a změnit údaje o případu vytvořený případ a) Změnit všechny/některé údaje b) Uložit pomocí tlačítka Uložit 3) Vybrat vytvořený případ a smazat pomocí tlačítka Smazat a) Stisknout tlačítko smazat b) Potvrdit

Název	Profily a účty
Cíl	Ověření funkcí pro vytvoření, editaci a mazání profilů a účtů.
Postup	1) Vytvořit nový případ (testovací scénář případy) 2) Vytvořit hlavní profil a) Pomocí tlačítka pro přidání nového hlavního profilu b) Pomocí kontextového menu c) Při vytváření přidat jeden/více účtů 3) Vytvořený profil přejmenovat 4) Přidat další účet

	5) Přejmenovat vybraný účet 6) Smazat vybraný účet 7) Smazat vybraný profil

Název	Import
Cíl	Ověřit funkce importu.
Postup	1) Vytvořit případ 2) Vytvořit čtyři profily, každý s jedním účtem 3) Pro každý profil a jeho účet importovat data 1. profil data z UFED 2. profil data z XRY 3. profil data z MPEP 4. profil soubory EML

Název	Hlavní panel
Cíl	Ověřit funkce Hlavního panelu. Navazuje na předešlý testovací scénář (Import).
Postup	1) Ověření zobrazení účtů, kontaktů a zpráv pro vybraný profil a účet 2) Ověřit funkci vyhledání kontaktů dle jména 3) Ověřit filtry pro zobrazení zpráv

Název	Hledání vazeb
Cíl	Ověřit funkce panelu pro hledání vazeb. Navazuje na předešlý scénář (Hlavní panel).
Postup	1) Vybrat jeden z hlavních profilů 2) Zobrazit panel Hledat vazby a) Stisknutím tlačítka na hlavním panelu b) Výběrem položky Hledat vazby z kontextového menu hlavních profilů 3) Vybranému profilu najít vazby 4) Otestovat otevření odkazu v prohlížeči 5) Přidat vybrané nalezené účty nebo kontakty a) Jako kontakty b) Jako účty 6) Zkontrolovat na hlavním panelu

Název	Vizualizace vazeb
Cíl	Ověřit funkce panelu Vizualizace vazeb. Scénář navazuje na scénář Import.
Postup	1) Ověření funkcí hledání profilů a kontaktů 2) Ověření přidávání, odebírání a přidávání společných kontaktů do seznamu vizualizovaných 3) Ověření filtrů (samostatně i společně) 4) Ověření funkce vykreslení 5) Ověření funkce generování reportu 6) Ověření samotného vytvořeného reportu

Příloha E

Výsledky testování

Tester	Martin Dušek
Kontakt	dusekmartin@hotmail.com

Průběh testování

- Testování proběhlo pomocí předem připravených testovacích scénářů, které mi dodal autor práce. Dále jsem měl k dispozici soubory pro import. (Jednalo se o reálná data získaná ze zařízení popsaného v kapitole 3.6 – pozn. aut.)
- Postup a pořadí testů bylo dodrženo, v některých případech jsem se snažil i o alternativní postup, kdy jsem některé kroky přerušil, abych viděl chování programu při zrušení nějakého postupu. Například při vytváření profilů, změně případu atd.
- Samotné testování proběhlo ve dvou kolech. Po prvním kole jsem autorovi předal seznam nalezených chyb a nedostatků. Po druhém kole byla již většina nalezených chyb opravena, nicméně jsem nalezl další. Jejich seznam jsem opět předal autorovi.

Nalezené nedostatky a chyby byly po druhém kole testování autorem odstraněny.