VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ
DEPARTMENT OF COMPUTER SYSTEMS

REGRESNÍ TESTOVÁNÍ V TELEKOMUNIKACÍCH

REGRESS TESTING IN THE TELCO AREA

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **MARTIN PAMÁNEK**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. MARTIN KRČMA**
SUPERVISOR

BRNO 2017

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav počítačových systémů Akademický rok 2016/2017

Zadání bakalářské práce

Řešitel: **Pamánek Martin**
Obor: Informační technologie
Téma: **Regresní testování v telekomunikacích**
Regress Testing in the Telco Area
Kategorie: Analýza a testování softwaru

Pokyny:

1. Seznamte se s problematikou regresního testování.
2. Nastudujte specifika regresního testování v oblasti telekomunikací.
3. Navrhněte obecný nástroj pro provádění regresních testů v této oblasti. Nástroj bude mít příkazové rozhraní.
4. Navržený nástroj implementujte.
5. Vytvořte ukázkový test pro jeden Comptel EventLink stream.
6. Demonstrujte běh nástroje a testu.
7. Navrhněte další rozvoj práce.

Literatura:

- Dle pokynů vedoucího.

Podrobné závazné pokyny pro vypracování bakalářské práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva bakalářské práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap (20 až 30% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Krčma Martin, Ing.**, UPSY FIT VUT
Datum zadání: 1. listopadu 2016
Datum odevzdání: 17. května 2017

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav počítačových systémů a sítí
612 66 Brno, Božetěchova 2

L.S.

prof. Ing. Lukáš Sekanina, Ph.D.
vedoucí ústavu

Abstrakt

Práce se zabývá problematikou telekomunikačních sítí. Diplomová práce popisuje základní prvky telekomunikačních sítí a zaměřuje se na jejich vzájemnou komunikaci neboli *mediaci*. Software zajišťující přenos dat mezi prvky telekomunikačního systému podléhá častým změnám, údržba a provoz tohoto typu softwaru si žádá velké množství zdrojů. Tato práce se snaží zmírnit náklady na údržbu softwaru za pomocí automatizace a regresního testování.

Abstract

This bachelor's thesis deals with the problematics of the telecommunication. It describes basic elements of telecommunication networks and focuses on the communication using different interafces between the elements. This process is also known as mediation. Software tools responsible for data transfers between the network elements are rapidly evolving due to a growing functionality of the telecommunication networks. The maintenance costs are considerable and growing as well. This thesis aims to reduce these costs using automation of regression testing processes in order to produce more reliable products.

Klíčová slova

Regresní testování, Telekomunikace, Mediace

Keywords

Regression testing, Telecommunications, Mediation

Citace

PAMÁNEK, Martin. *Regresní testování v telekomunikacích*. Brno, 2017. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Krčma Martin.

Regresní testování v telekomunikacích

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Martina Krčmy. Další informace mi poskytl odborný konzultant Ing. Petr Štráfelda. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Martin Pamánek
15. května 2017

Poděkování

Tímto bych chtěl poděkovat vedoucímu práce Ing. Martinu Krčmovi za všechny poskytnuté rady, odborné vedení a podnětné připomínky k podobě této práce. Svým objektivním a ochotným přístupem ke studentům jde příkladem. Můj dík si také zaslouží odborný konzultant Ing. Petr Štráfelda za veškerou pomoc při návrhu a implementaci aplikace. Jeho trefné připomínky a široké znalosti byly velikou pomocí.

Obsah

Seznam obrázků

Kapitola 1

Úvod

Rozvoj výpočetní techniky ovlivňuje i vývoj v oblasti telekomunikačních sítí. Telekomunikační sítě však neslouží pouze pro potřeby člověka a jeho dorozumívání se především prostřednictvím telefonních služeb, ale jejich využití spočívá i v přenosu ostatních dat. V dnešní době je přítomna vysoká úroveň propojení klasické telefonie a počítačových datových sítí. Důsledkem je tak vytvoření univerzální telekomunikační sítě, jež dokáže přenášet data pro různé typy služeb [15]. Pro všechny služby (například textové zprávy nebo videa) jsou využívány obdobné přenosové prostředky. Nejznámější sítí pro výměnu informací je Internet. Jedná se o vzájemně propojené telekomunikační sítě jednotlivých operátorů. Součástí Internetu je však více - datová úložiště, výpočetní centra, servery. Telekomunikační sítě operátorů se mohou překrývat, ale vždy jsou od sebe odděleny. Propojeny jsou v takzvaných propojovacích bodech, kterých je na světě přibližně 156.

Poskytovatelé telekomunikačních služeb musí zajistit dostupnost a kvalitu nabízených služeb, velice důležité je ale i správné vyúčtování za tyto služby. Regulace Evropské Unie, států nebo politika samotných poskytovatelů vyžaduje velkou dynamiku v podobě poskytovaných služeb. Software zajišťující potřebnou funkcionalitu tak velice často prochází změnami. Nedílnou součástí při implementaci změn je důsledné otestování přidané funkcionality systému. Neméně důležité je také zjištění, zda-li stávající funkčnost systému nebyla změněna nežádoucím způsobem při přidání nebo odebrání žádané funkce. K tomu se využívá regresní testování.

Cílem této práce je vytvoření nástroje regresně testujícího software, který spojuje rozhraní mezi jednotlivými součástmi telekomunikační sítě. Comptel EventLink je příkladem softwaru sloužícího jako prostředník mezi prvky sítě, tento proces se nazývá také mediace. Protože nástroje pro regresní testování Eventlinku jsou obtížně dostupné, je cílem této práce takový prostředek vytvořit.

První část práce se věnuje základní problematice telekomunikačních sítí, především oboru telefonie. Tato část vysvětluje funkci a zařazuje do kontextu software EventLink od společnosti Comptel. Dále jsou zde rozebrány komponenty EventLinku a jejich využití. Druhá sekce se věnuje návrhu aplikace pro regresní testování EventLinku. Obsahem je také seznámení s podobou aplikace, její rozhraní a funkce, které aplikace umožňuje. Třetí část rozšiřuje předchozí bod a věnuje se aplikaci pro regresivní testování detailněji. Je zde popsána implementace klíčových částí a principy fungování aplikace. Poslední část je věnována využití této aplikace. Je zde podrobné seznámení s podobou výsledků, které jsou dosaženy s využitím aplikace. Pro demonstraci jsou použity reálné příklady, které mohou nastat v praxi.

Kapitola 2

Telekomunikační technologie

Telekomunikační síť je kolekcí spojovacích uzlů (stanice sloužící k přepojování datových telefonních služeb), kanálů (ty mohou být fyzického druhu, v tom případě je kanál tvořen metalickým vodičem, nebo je spojení zajištěno nehmotně, příkladem je rádiový kanál) a terminálů (koncová zařízení, typicky se jedná o mobilní telefony). Tyto prvky jsou navzájem propojeny tak, aby zajistili telekomunikaci mezi terminály. Síť může využívat především dvě technologie komunikace. První z technologií, přepojování okruhů, využívá předem sestaveného okruhu. Výhodou této komunikace je nízká režie takového obvodu. Druhá technologie (přepojování paketů) využívá přepínače pro směrování [1]. Každý terminál má unikátní síťovou adresu (jednoznačný identifikátor síťového zařízení, například pro počítačové sítě se používá MAC adresa - Media Access Control).

Pro účely této práce je důležitá především síť telefonní. Existuje více druhů telefonních sítí, zde jsou uvedeny dvě nejdůležitější. A to za prvé veřejná telefonní síť, kde jsou účastníci propojeni k telefonní ústředně pomocí fyzického vedení. Za druhé je to mobilní telefonní síť. Tento typ sítě využívá pro připojení rádiových vln. Mobilní a veřejné telefonní sítě vlastní veřejní telefonní operátoři *Public Telephone Operators* (PTO). Virtuální síťoví operátoři *Virtual Network Operators* (VNO) si pronajímají kapacity od PTO a následně prodávají veřejnosti telefonní služby [6].

2.1 Mobilní sítě

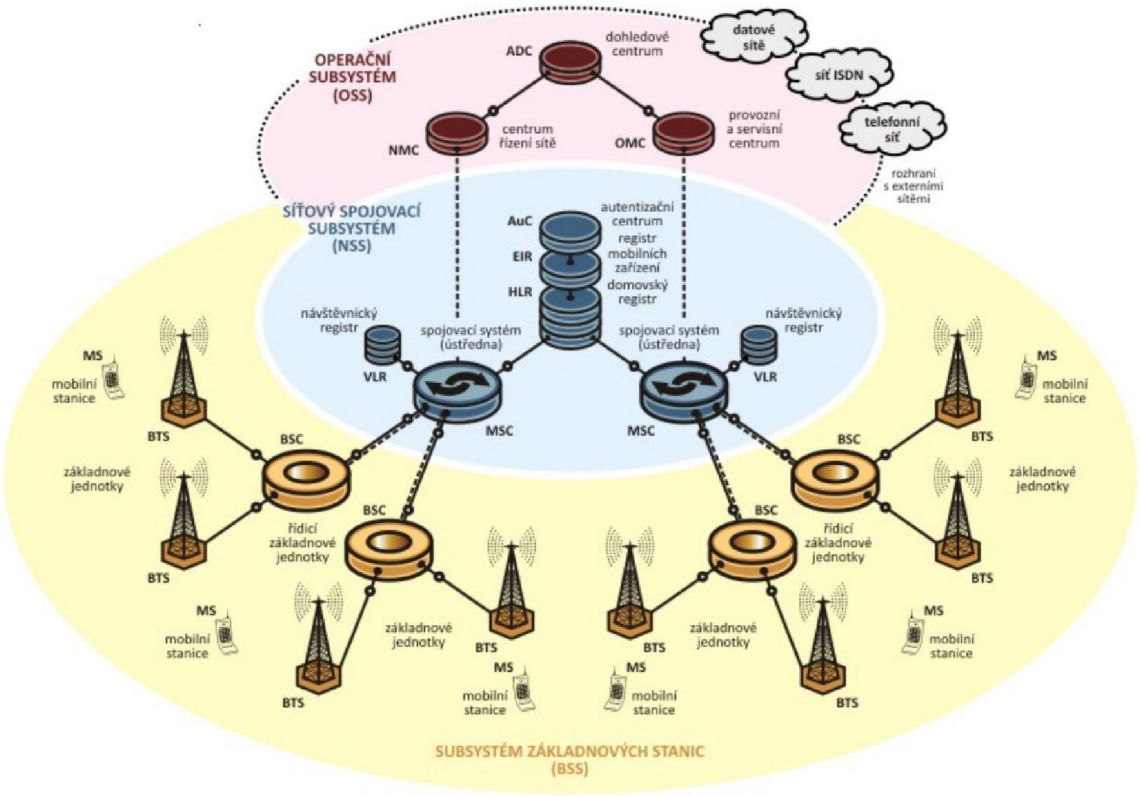
Začátky mobilních sítí sahají do počátků osmdesátých let, kdy byly zprovozněny první 1G sítě. Jednalo se o analogové radiotelefonní mobilní systémy, využívající rádiového rozhraní. Služby byly přidělovány pomocí principu FDMA (*Frequency Division Multiple Access*), kde je frekvenční pásmo rozděleno na určitý počet kanálů a účastník sítě má přidělený právě jeden kanál [5]. Nástupcem 1G sítě se stala GSM neboli Global System for Mobile Communication (2G) síť, která je založena na digitálním přenosu dat. V jiných částech světa než je Evropa nemusí být 2G síť vždy GSM, nicméně tento typ je nejrozšířenější. Digitální přenos obecně zajišťuje nízkou ztrátu kvality přenášených dat. Nyní se používají náročnější standardy než 2G síť. Zatím nejnovějším standardem je 5G, jehož použití se plánuje na rok 2020. [8].Tato nová generace by měla přinést vylepšené pokrytí a razantně snížit zpoždění. Přenosové rychlosti by měly dosahovat až sto megabitů za sekundu v metropolích.

Pro objasnění základních principů se nadále bude práce věnovat především základnímu modelu 2G. Na rozdíl od sítí první generace používá GSM síť pro přístup do sítě TDMA (*Time Division Multiple Access*). TDMA je založeno na přidělování celého frekvenčního ka-

nálu po krátkých časových úsecích. Každý uživatel má tedy po nějakém čase přidělen krátký časový slot. Mezi služby, které 2G poskytuje patří: telefonie (včetně tísňového volání a to i v zahraničí), krátké textové zprávy neboli Short Message Services (SMS), hlasová schránka, e-mail, bankovní služby, informační služby, mezinárodní roaming (poskytování služeb účastníkovi v jiné zemi, než kde má účastník zaregistrované svoje telekomunikační služby). Mezi přenosové služby pak lze zařadit asynchronní (synchronizace přenosu je zajištěna pomocí vysílání synchronizačních znaků v rámci přenášených dat) duplexní přenos dat o přenosových rychlostech až 9600 bit/s. Je zde také možnost synchronního (zde je potřeba, aby zdroj vysílal data konstantní přenosovou rychlostí) duplexního přenosu dat a to také s přenosovou rychlostí až 9600 bit/s [11].

2.1.1 Struktura 2G sítě

Strukturu 2G můžeme rozdělit na tři základní části. Strukturu demonstruje obrázek 2.1.



Obrázek 2.1: Příklad architektury mobilní sítě [16]

Base Station System (Systém základnových stanic) se skládá z mobilních stanic *Mobile Station* (MS) jež komunikují se základnovými stanicemi *Base Transceiver Station* (BTS). Několika základnovým stanicím je přiřazena jedna řídící základnová jednotka *Base Station Controler* (BSC). Mobilní stanice vybere BTS s nejlepším signálem.

Network Switching System (NSS) je část GSM sítě, jež uskutečňuje propojování hovorů. Umožňuje mobilním telefonům komunikovat s ostatními zařízeními v síti. Mezi hlavní prvky patří především:

- *Home Location Register* (HLR) - Jedná se o centrální databázi mobilní sítě. Obsahuje informace o každé SIM kartě (subscriber identity module) vydané mobilním operátorem. SIM karta má unikátní identifikátor zvaný IMSI (international mobile subscriber identity), jež je primárním klíčem každého záznamu HLR. Další pojem spjatý se SIM je MSISDN (mobile subscriber integrated services digital network number). Jedná se o telefonní číslo, které je používáno pro příchozí a odchozí hovory. Nicméně SIM může mít i další MSISDN, které je použito pro fax a datové spojení. MSISDN je taktéž primárním klíčem databáze [10].

- *Mobile Switching Centre* (MSC) - Jejími úkoly jsou spojování telekonferencí, faxových a datových hovorů, účast na přenosu SMS, autentizace uživatelů. Komunikace s VLR, HLR apod.

- *Visitor Location Register* (VLR) - Obsahuje databázi všech účastníků mobilní sítě pro danou MSC. VLR je často integrována přímo do MSC. VLR informuje HLR v případě, že se účastník dostal do spádové oblasti.

- *Equipment Identity Register* (EIR) - Je to databáze čísel IMEI (International Mobile Equipment Identity), neboli unikátní číslo přidělené výrobcem telefonu. Využívá se pro blokování přístupu uživatelů, kteří jsou na černé listině. Není to nutná součást sítě.

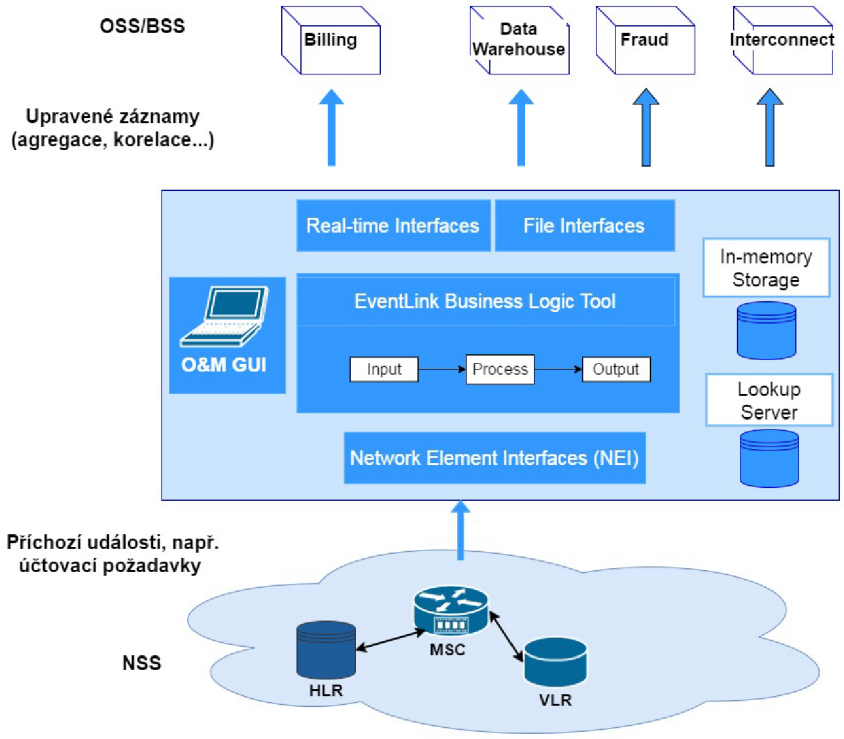
Operations Support System (OSS) zabezpečuje provoz BSS a NSS systémů. OSS ma za úkol udržovat telekomunikační síť v chodu, tedy stará se o konfiguraci a údržbu zařízení a dat, opravu chyb či zajištění služeb. Nedílnou součástí je také *monitoring* a *reporting*. *Business Support System* (BSS) má za úkol komunikaci se zákazníky. BSS zpracovává požadavky zákazníků, vytváří faktury nebo vybírá platby za telekomunikační služby. Rozdíl mezi OSS a BSS je minimální (vychází především z historie, kdy se jednalo o nezávislé systémy), protože není jasně definováno, jaké funkce konkrétní systém musí podporovat. Lze však říci, že OSS je více hardwarově orientován, zatímco BSS má blíže k softwaru [12]. Tyto dva systémy jsou často označovány jako jeden celek - OSS/BSS.

Administrative Centre (ADC) má na starosti administrativní úkoly jako např. správa poplatků a vyúčtování. *Network Management Centre* (NMC) zajišťuje řízení toku informací v síti. *Operation and Maintenance Centre* (OMC) provádí údržbu a provoz sítě.

2.2 Comptel EventLink

Vzhledem ke komplexnosti dnešních telekomunikačních sítí, jsou rozhraní mezi OSS/BSS a prvky sítě (NSS) často odlišné. Zde je zapotřebí *mediace*. Pojem mediace tedy označuje software, který přenáší data mezi dvěma odlišnými rozhraními. Mezi firmy nabízející mediaci patří např. CSG (Anglie), Enterest (Německo), Digital Route(Švédsko) či Comptel (Finsko). V rámci této práce bude vždy brána v potaz mediace od firmy Comptel. Pro znázornění mediace viz obrázek 2.2.

Fraud monitoruje chování uživatelů a hledá známky nestandardního či podvodného chování. *Interconnect* je mechanismus sloužící k účtování služeb mezi jinými operátory. *Billing* sbírá informace o uživateli poskytnutých službách. Na základě předdefinovaných

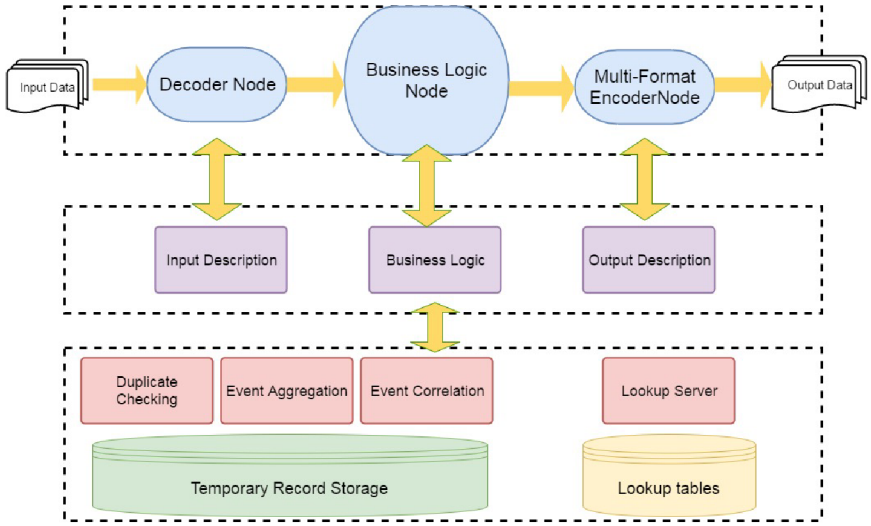


Obrázek 2.2: Mediace mezi OSS/BSS a NSS v případě EventLinku

pravidel pak generuje zákazníkům vyúčtování služeb. *Data Warehouse* je subsystém pro uchování a zpracování dat, se kterými mobilní síť pracuje [17].

Comptel EventLink je příkladem mediačního softwaru. EventLink je schopen sbírat data z *IP Multimedia Subsystem* (IMS), což je GSM/UMTS rozšířené o multimediální přenosy. Dále pak *Next-generation network* (NGN), založeno na zabalení dat do IP paketů. Podpora 2G-4G sítí, IP, pevných linek, cloudů nebo satelitů. EventLink následně upraví nasbíraná data tak, aby odpovídala rozhraní cílových systémů (příkladem budiž OSS/BSS elementy předchozího obrázku. Základní podobu prvků EventLink platformy ilustruje obrázek 2.3.

Základním prvkem EventLinku je *node* neboli uzel. Jedná se o nejmenší funkční jednotku. Uzly mohou pracovat nezávisle na sobě, nicméně nemohou dostávat data z předešlých uzlů, které aktuálně nepracují. Přenos dat mezi jednotlivými uzly je implementován pomocí bufferů, kdy node má vstupní i výstupní buffer. Uspořádaná množina uzlů vytváří *Stream*. Stream se skládá z neomezeného množství uzlů (v praxi množství uzlů závisí na cílové platformě). Návaznost jednotlivých uzlů je dána konfigurací streamu. Stream pracuje v jednom ze dvou režimů: offline (přenos souborů pomocí FTP), nebo online (pro zajištění minimální latence). Správce uzlů *Node manager* je zodpovědný za chod uzlů - monitoruje jejich chod, restartuje uzly, ukládá auditní data (informace o chování a problémech, ke kterým v uzlu došlo). Dalším důležitým elementem je *Lookup Server*. Lookup Server slouží k vyhledávání dat v systémové databázi.Využíván je především pro modifikaci streamem zpracovávaných dat. *In-memory Storage* slouží jako úložiště dat. Toho se využívá při dočasném uložení záznamů, korelaci, agregaci, kontrole duplikátů apod.



Obrázek 2.3: Prvky EventLinku

2.3 Stream

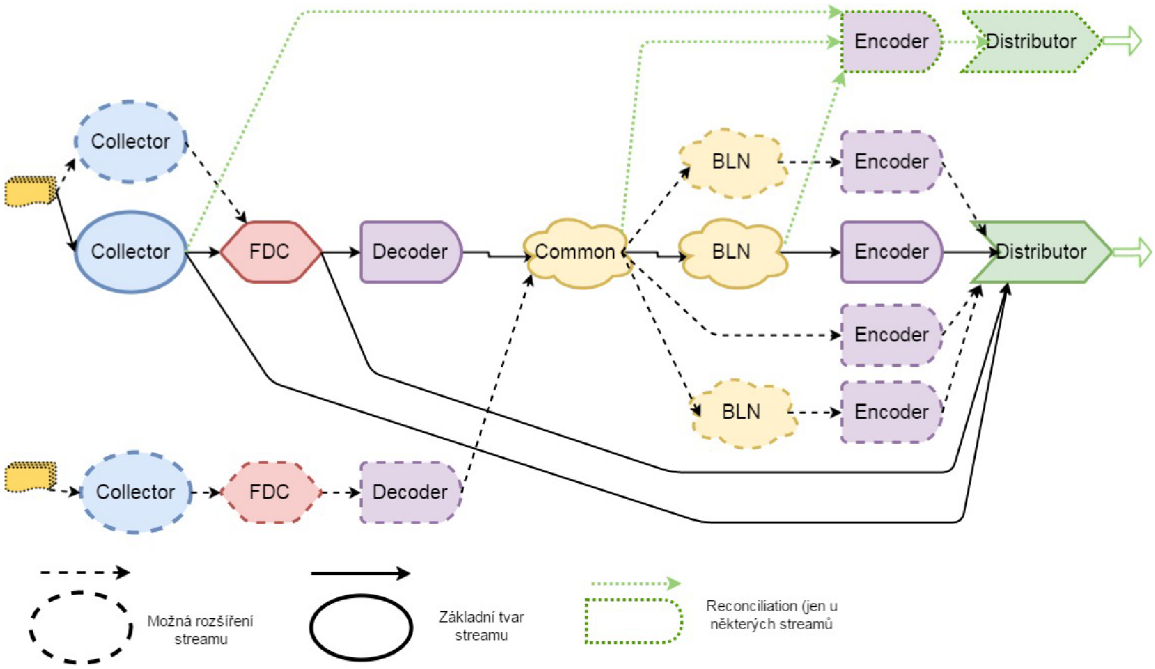
Tato podkapitola se věnuje streamu a jeho základním elementům. Stream může obsahovat teoreticky neomezené množství uzlů. Každý uzel má jednu nebo více mediačních funkcí. Uzly lze rozdělit do několika kategorií, které budou popsány níže. Jednoduché schéma je znázorněno na obrázku 2.4. Prvky ohraničené plnou čarou značí minimální podobu streamu. Minimální podobou je myšlen stream v základní podobě, která se v praxi používá. Nicméně logika streamu je často složitější a obsahuje více uzlů dané kategorie. Tento případ je naznačen uzly s přerušovanou čarou. Dále pak uzly mohou generovat rekonciliaci, viz kap. 2.3.7.

2.3.1 Kolektor

Kolektor uzel slouží ke sběru dat, které má stream za úkol zpracovat. Rozhraní kolektoru umožňuje sběr dat např. pomocí mobilních sítí (2G až 4G), IP, pevná linka, cloud aj.. Kolektor podporuje FTP, SFTP, TCP/IP, GTP, FTAM protokoly. Sběr dat ze sítě je prováděn pomocí online sběru dat, nebo offline - sběr souborů. Soubory lze sbírat kontinuálně nebo jejich sběr nastavit na určitý čas. Mezi funkcionalitu kolektoru dále patří bezpečný přenos (potvrzení úspěšného přenosu a kontrolní součet) či možnost přejmenovat soubory. Kolektory také podporují zálohu dat, která může být použita ke znovuzpracování.

2.3.2 FDC

File Duplicate Checker kontroluje přítomnost duplicitních záznamů, aby nedošlo ke zpracování některého požadavku vícekrát. Kontroly jsou prováděny na základě obsahu, jména, velikosti nebo zdrojového id průchozích souborů. Ověřuje také pořadí záznamů. To zajistí, že záznamy se do systému dostanou ve správném pořadí, nedojde k jejich ztrátě nebo zpoždění a nedojde k vícenásobnému zpracování dat.



Obrázek 2.4: Kategorie uzlů ve streamu

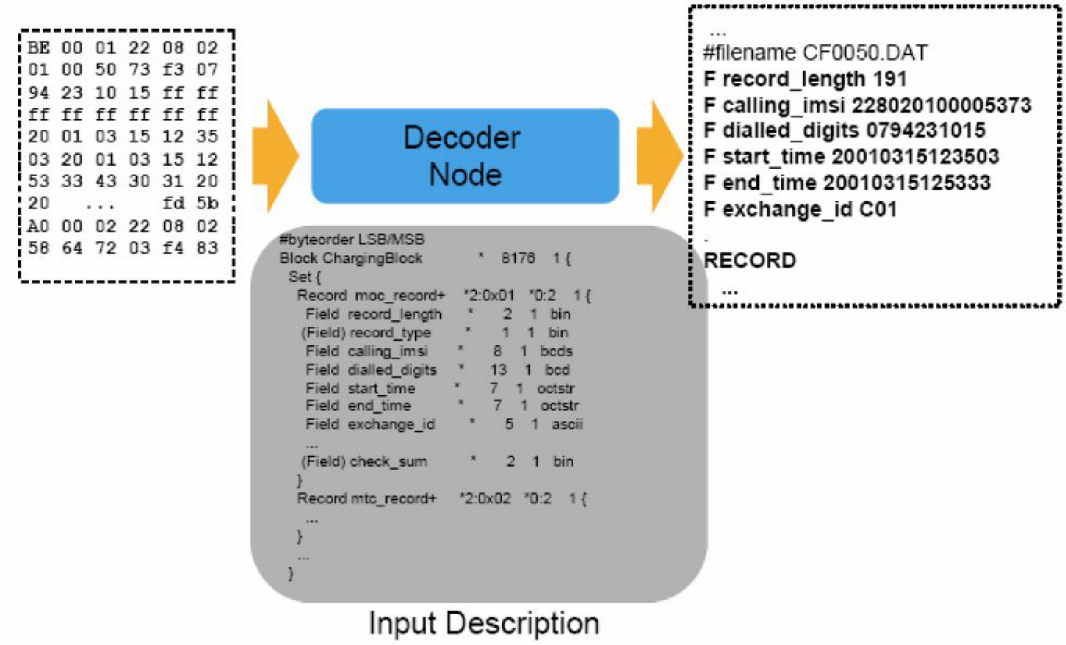
2.3.3 Dekodér

Sebraná data jsou dekodérem dekódována do interního formátu EventLinku pro další zpracování. Dekodér je tvořen aplikační částí a konfigurační částí. Jsou podporována data splňující pravidla ASN.1. neboli *Abstract Syntax Notation One*. ASN.1 [4] poskytuje soubor formálních pravidel umožňujících popsat strukturu objektů způsobem nezávislým na konkrétním hardwarovém řešení. Podporováno je také XML, nebo data ve formátu .txt. Data jsou dekódována na základě specifické konfigurace (gramatiky) obsažené v každém dekodéru. Data, jež nesplňují požadavky dané gramatikou, jsou zahozena. Schéma dekodéru lze vidět na obrázku 2.5.

2.3.4 Byznys logika

Byznys logika pracuje se záznamy v interním formátu EvetLinku. Záznamy, CDRs (*Call Data Records*) jsou informace, které mobilní zařízení sbírají kdykoli využíváme telekomunikačních služeb. Záznamy obsahují obrovské množství informací o tom jak, s kým a kdy komunikujeme. Proto se jimi zabývá množství výzkumů [18]. Byznys logika je zpravidla uzel s největší funkcionalitou. Uzel definuje pravidla, která mění zpracovávaná data podle přání zákazníka. Nejčastější operace v rámci byznys logiky jsou:

- *Validace* - Je kontrolována přítomnost a formát položek záznamu. Pokud je potřeba, jsou vytvořeny položky nové nebo upraveny položky stávající. Příkladem např. kontrola data a času.
- *Filtrace* - Záznamy nesplňující určitá kritéria jsou vynechány a nejsou dále zpracovány, jsou zahozeny.
- *Rozšíření záznamu* - Zpracovávané záznamy často neobsahují všechna potřebná data, je tedy zapotřebí získat data z externích systémů. Z dostupných dat jsou vytvořeny



Obrázek 2.5: Schéma dekodéru [2]

vyhledávací klíče, které jsou použity k vyhledání potřebné hodnoty v databázi za pomoci lookup serveru.

- *Agregace* - Používá se například u dlouho trvajících hovorů, kdy je vytvořeno více částečných záznamů. Vznikne tak jeden souhrnný záznam.
- *Korelace* - Používá se taktéž ke slučování záznamů, nicméně zde se slučují záznamy z různých platforem a v různých formátech.

Pozn.: Byznys uzly se mohou větvit (např. z důvodu vnitrostátní a roamingové služby). Každý uzel tedy může zpracovávat odlišná data. Byznys uzel s názvem *common* provádí operace nad daty, které jsou společné všem BLN.

2.3.5 Enkodér

Před distribucí dat do OSS/BSS systémů je potřeba převést data do formátu, který tyto systémy používají. K tomu slouží právě enkodér. Enkodér podporuje generování dat ve formátu ASCII (*American Standard Code for Information Interchange*), binárního, ASN.1 či XML formátu.

2.3.6 Distributor

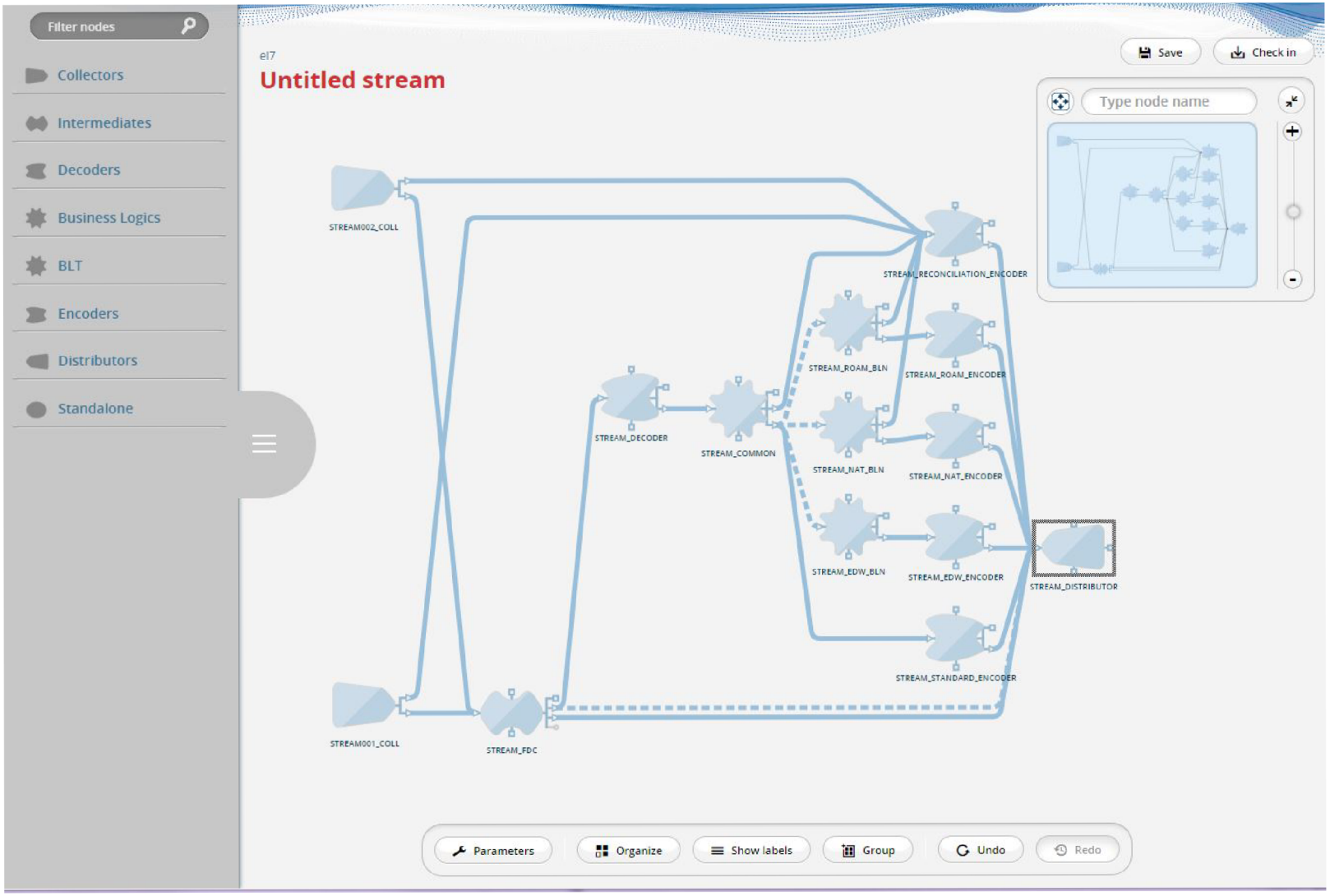
Distributor přenáší data do OSS/BSS systémů. Přenos dat u distributoru má obdobné možnosti jako přenos dat u kolektoru, tedy online protokol nebo pomocí balíků souborů. Distribuce probíhá okamžitě nebo lze přenášet data v určitý čas.

2.3.7 Rekonciliace

Rekonciliace (Reconciliation) slouží ke kontrole správné funkce uzlů. Pokud je v parametrech uzlů nastaveno generování rekonciliace, jsou daným uzlem generována rekonciliační data. Tyto data obsahují informace o množství záznamů, které byly zpracovány uzlem, počtu zahozených záznamů, počtu úspěšně zpracovaných záznamů. Slouží tedy ke kontrole funkce uzlu. Lze zde zjistit, zda se v uzlech neztrácejí záznamy.

2.3.8 GUI

GUI (Graphical User Interface) je nedílnou součástí mediačního produktu Comptelu EventLink. Ačkoliv jednotlivé uzly streamu můžeme naprogramovat a otestovat bez GUI, pro samotnou tvorbu a propojení jednotlivých uzlů je nejlepší využít možností GUI. Dále umožňuje monitoring, tedy sledovat zda streamy aktuálně běží nebo ne. Grafické uživatelské rozhraní umožňuje zobrazit statistiku průchozích dat jednotlivými uzly nebo GUI upozorní uživatele na neočekávané chování varováním. Uživatel zde má možnost přidávat a upravovat lookup servery, případně je vypínat a zapínat. Obrázek 2.6 znázorňuje, jak vypadá stream v grafickém uživatelském prostředí.



Obrázek 2.6: Příklad streamu v GUI

Kapitola 3

Návrh

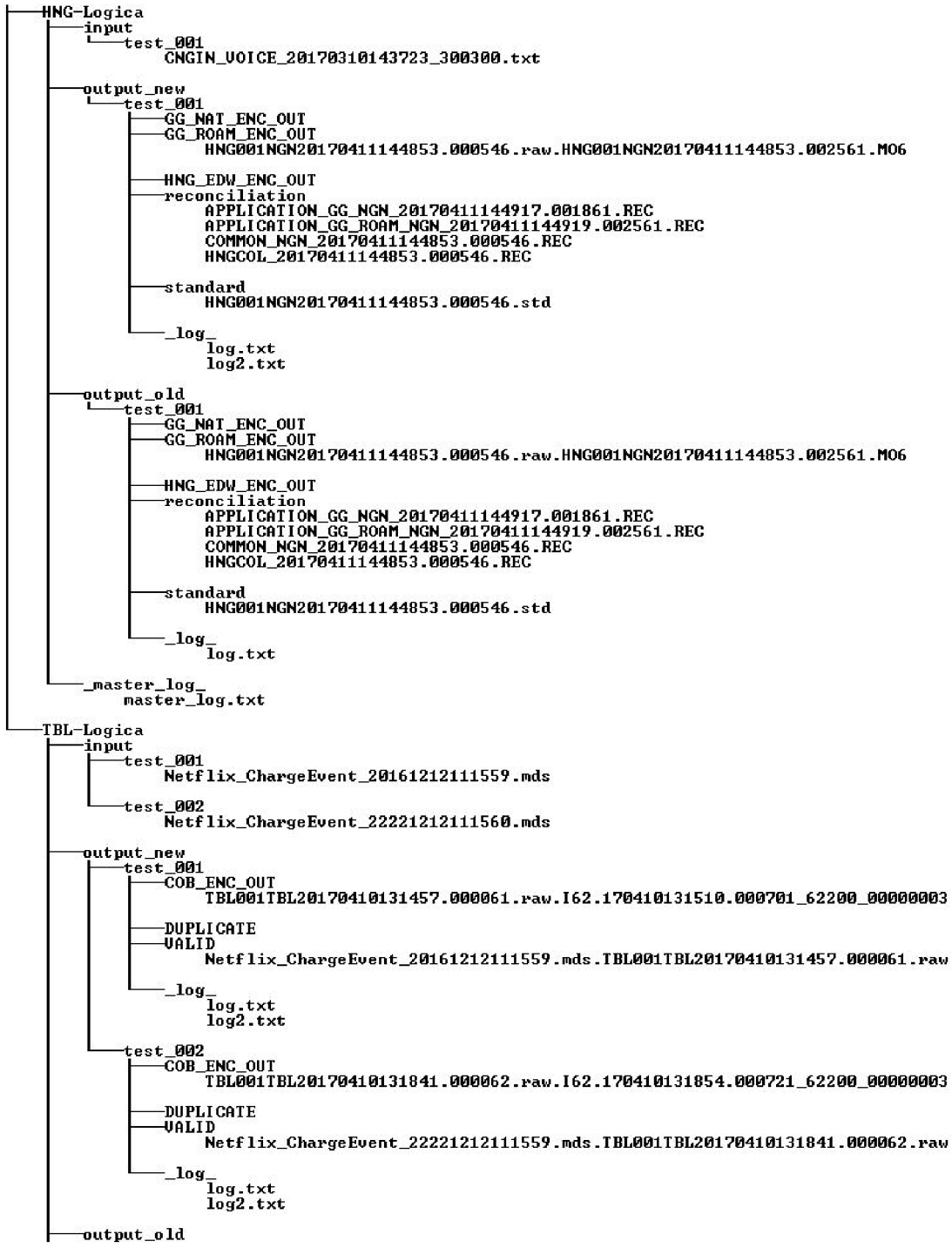
Specifikace rozhraní je následující. Aplikace pro regresní testování streamů má konzolovou podobu. Interakce tak probíhá pomocí příkazové řádky při spuštění aplikace s parametry. Výstup aplikace je tvořen soubory generovanými na základě odlišností předešlých a současných výstupních dat testovaného streamu. Průběh aplikace je možné částečně monitorovat pomocí EventLink GUI. Konkrétně je možné sledovat stav streamu a jeho uzlů (vypnuté/zapnuté). Pod pojmem monitorovat si lze představit např. stav streamu a jeho uzlů, množství dat proudících streamem, počet zahozených souborů, generování varovných hlášení a podobně.

3.1 Rozhraní

Testovací data mají následující hierarchii. Všechna data náležící jednomu streamu jsou oddělena do samostatných složek (včetně dat výstupních a výsledků). Složky mají jméno podle svého streamu. Podsložka *HNG-Logica/ input* obsahuje všechny dílčí testovací skupiny. *HNG-Logica* zastupuje jméno streamu, *input* značí složku, která obsahuje vstupní testovací data. Podsložka s názvem skupiny testů (například *test_001*) obsahuje testovací soubor (nebo skupinu souborů). Strukturu testovacích souborů demonstruje obrázek 3.1. Pro stream HNG je jedním z testovacích souborů: *HNG-Logica/ input/test_001/-CNGIN_-VOICE_20170310143723_300300.txt.*

Výstupní data budou mít dvě verze. První z nich jsou data referenční, na obrázku se jedná o složku *output_old*. Tato složka obsahuje data zpracovaná některým z předchozích běhů aplikace. Jak název napovídá, složka obsahuje data referenční, tedy ta která jsou považována za správná, a se kterými se poté porovnávají data nově získaná. Jak lze vidět na obrázku 3.1, vychází se ze struktury vstupních dat, ale s tím rozdílem že jednotlivé podsložky obsahující sady testů (například *test_001*) obsahují ještě složku s názvem výstupu ze streamu. Tyto výstupy jsou vlastně linky, které jsou distributorem posílány dále do systémů. Do streamů proudí velice často více druhů záznamů. Příkladem může být vnitrostátní a domácí hovor. Záznamy pro každého z nich se od sebe budou lišit. Protože je jejich obsah odlišný, budou se také jinak zpracovávat v BSS. Link je tedy kanál pro jeden druh záznamů, které mají stejný formát a jsou zpracovány stejnou částí BSS systému.

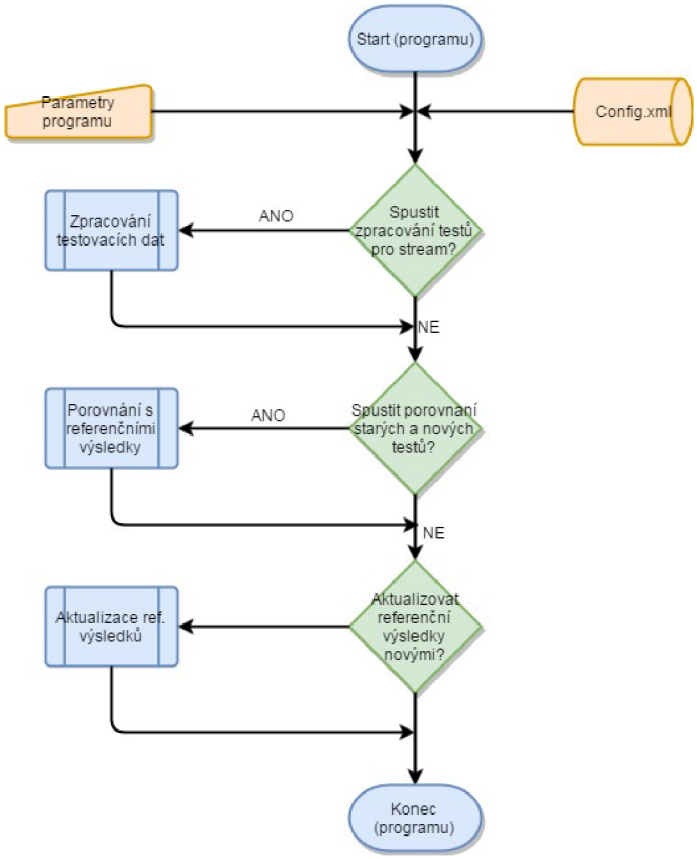
Druhá sada výstupních dat jsou ta, která aplikace za pomoci streamu zpracovala jako poslední. Strukturou jsou shodná s daty referenčními. Dále je zde podsložka *HNG-Logica/ output_new/_log_*, kde jsou dva logovací soubory. Tyto soubory obsahují výsledky porovnání nových a referenčních výstupů. Jeden soubor obsahuje podrobné výsledky, druhý



Obrázek 3.1: Struktura testovacích souborů

potom výsledky souhrnné. V referenční složce je výsledek testů také, nicméně zde se jedná o porovnání výstupů referenčních s ještě dřívějšími daty. Je tak zajištěna historie testování. Podsložka *_master_log_* pak obsahuje souhrnné informace všech testovacích případů.

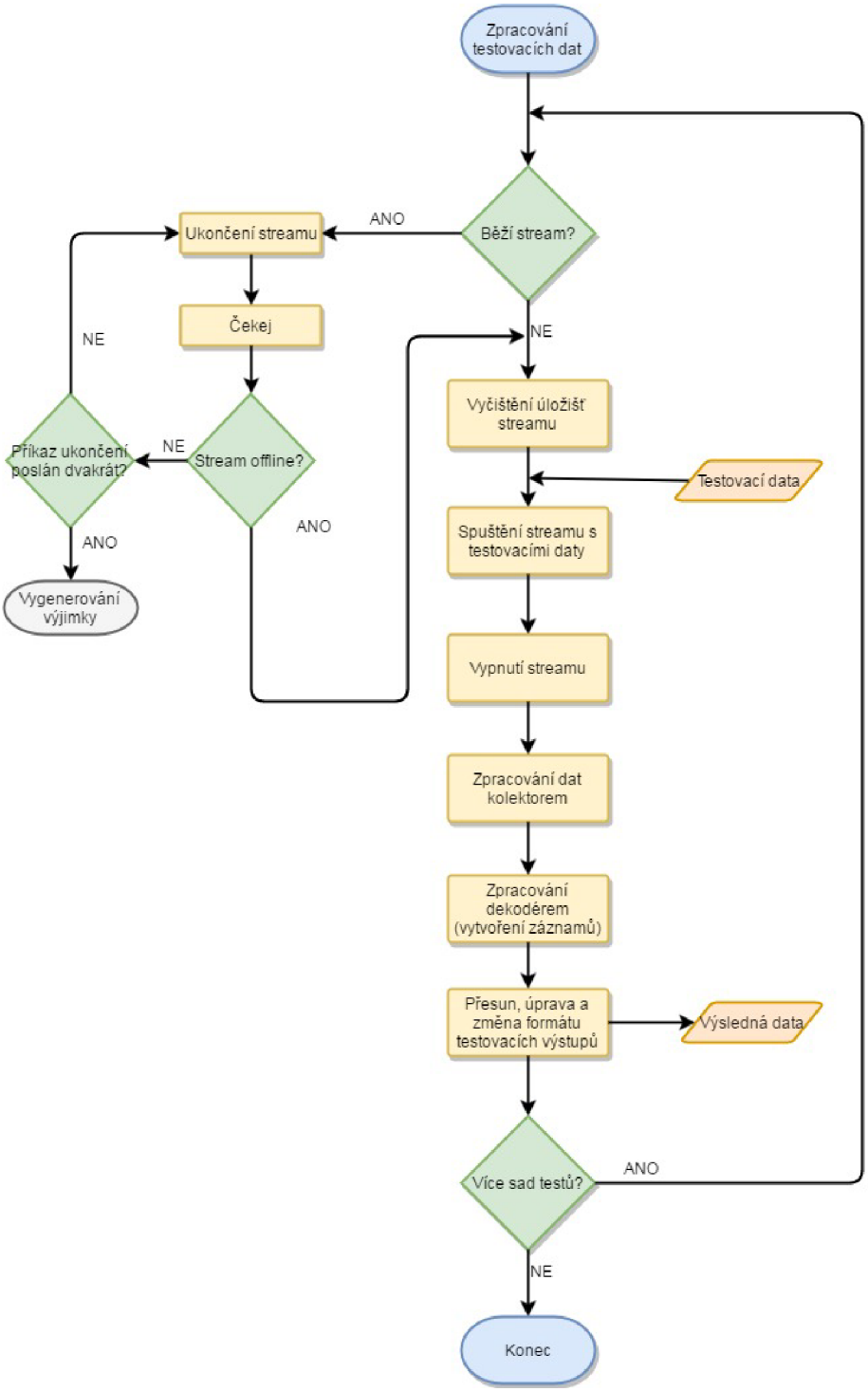
3.2 Aplikace



Obrázek 3.2: Tok programu

Aplikace má tři nezávislé moduly. Základní tok programu znázorňuje obrázek 3.2. První část aplikace má na starosti zpracování testovacích dat zvoleným streamem. Program zjistí stav streamu v prostředí EventLinku. Pokud je stream online, zajistí se jeho vypnutí. Děje se tak proto, že počáteční stav streamu musí být nulový. Nyní je třeba ještě vyčistit vnitřní úložiště, jež náleží našemu streamu. V tuto chvíli je stream ve výchozí pozici a lze pomocí něho zpracovávat nová data (v případě běžícího streamu, nebo nevyčištěných úložišť by mohlo dojít ke špatným výsledkům v důsledku zpracovávaných dat streamem, která nesouvisí s naším testováním). Na vstup jsou přivedena testovací data a je zapnut stream. Ve chvíli, kdy stream dokončí zpracování vstupních dat, je ukončena jeho činnost. Protože data změněná streamem jsou určena pro další systémy a nejsou člověkem snadno čitelná, je třeba je upravit. Data jsou zpracována samostatnými uzly v podobě kolektoru a dekodéru s konfigurací specifickou pro každý výstupní link streamu. Dekodér vytvoří z dat snadno čitelné záznamy. Takto vytvořené soubory skládající se ze záznamů jsou přeneseny na určené místo ve stromové struktuře dat, jak ukazuje obrázek 3.2. Soubory jsou ještě převedeny z interního formátu Eventlinku do textové podoby a záznamy jsou seřazeny. Stream totiž negarantuje

vždy stejné pořadí záznamů v souborech. Postup je opakován pro všechny skupiny testů. Flow diagram této části návrhu zobrazuje obrázek 3.3.



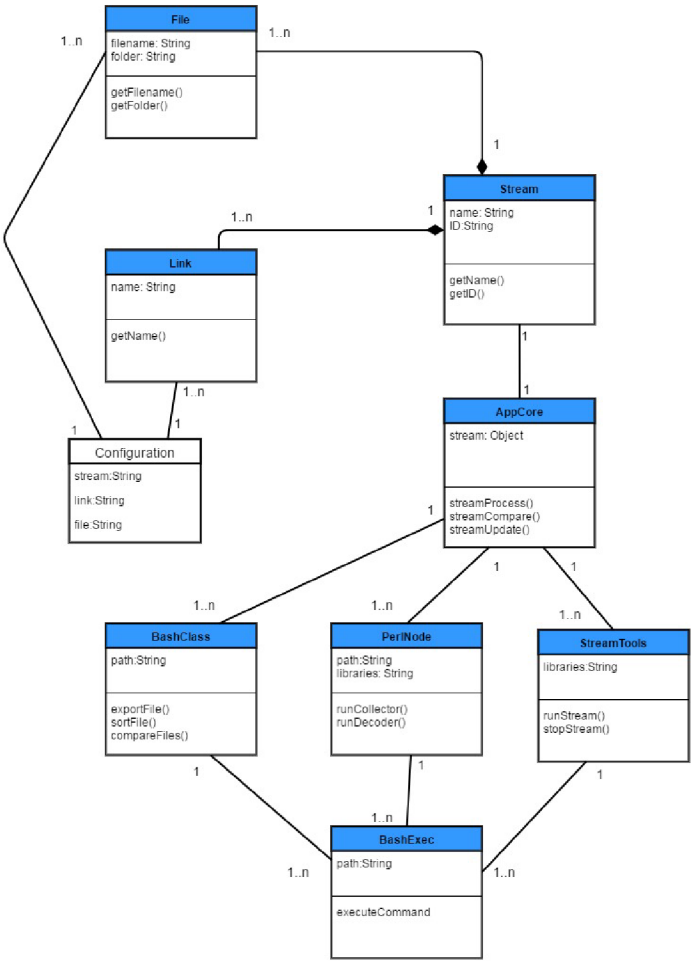
Obrázek 3.3: Tok programu v případě zpracování dat streamem

Pokud je zvolen i parametr pro vytvoření porovnaní nejnovějších a referenčních dat, aplikace začne postupně porovnávat jednotlivé složky výstupních dat. Pokud se ve složce

nachází více souborů, jsou seřazeny podle velikosti. Následně aplikace s pomocí bashovských příkazů začne porovnávat nová a referenční data. (V potaz je brán Bash z unixových shellů, jež interpretuje příkazový řádek, dalšími interprety jsou například *Korn shell - ksh* či *C shell - csh.*) [7] Výsledek je generován do logovacích souborů a dále upravován. V případě úspěšných výsledků v souladu se specifikací požadovaných změn, lze pomocí této aplikace nahradit referenční soubory novými.

Protože aplikace používá software třetích stran, jsou zde možná selhání aplikace. Jedná se především o fungování *Node Managera*, řídícího průběh fungování streamu. Odstranění tohoto problému musí být provedeno manuálně. Po odstranění problému lze aplikaci znovu spustit.

Návrh z pohledu třídního diagramu znázorňuje obrázek 3.4. *Configuration* je objekt, který obsahuje statická a perzistentní data. Jedná se o XML soubor uchovávající informace jaké testovací případy mají být spuštěny, případně jaké výstupní linky mají být testovány. Jsou zde uloženy informace o testovaných streamech - seznam testovacích souborů určených pro daný stream a pak také seznam výstupních linků streamů (výstup streamu je často rozdělen podle charakteru dat na více typů, každý typ má svůj link). Třída *File* obsahuje všechny informace o vstupních testovacích souborech, jejich jméno a umístění. Podobně třída *Link* obsahuje informace o každém linku, který stream používá. Třída *Stream* slouží k uchování informací o streamu, jako je jeho jméno, id, apod. Také jsou zde uloženy cesty pro často používané skripty či uchovány informace o tom, jaké operace má aplikace provést. Jádro aplikace *AppCore* obsahuje kostru aplikace a má řídící funkci. Vyjadřuje základní úroveň aplikace. *Bashclass* je třída implementující operace, které jsou prováděny příkazy bashe. *StreamTools* třída implementuje metody potřebné ke komunikaci se streamem. Využívá skripty interpretovaného programovacího jazyka Perl dodaných společností Comptel. *PerlNode* má velice podobnou funkci jako StreamTools, nicméně tato třída místo komunikace se streamem řeší komunikaci pouze s jednotlivými uzly, využívá také další skripty. Konečně třída *BashExec* tvoří mezivrstvu mezi třídami aplikace a jakýmkoliv jiným softwarem (nejčastěji se jedná bashovské či perlovské skripty).



Obrázek 3.4: Třídní diagram aplikace

Kapitola 4

Implementace

Tato kapitola se zaměřuje na implementaci vybraných částí aplikace. Jako programovací jazyk je použita Java. Aplikace běží na verzi "1.7.0_60". Podporovány jsou i vyšší verze. V aplikaci je také použit skriptovací jazyk Perl, konkrétně verze "5.8.9". Software je používán a testován na operačním systému GNU/Linux s 64 bitovou architekturou.

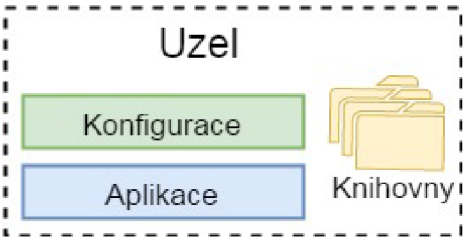
Aplikace lze rozdělit hierarchicky na více úrovní. První úroveň aplikace je implementována v Javě. Software využívá konfiguračního souboru *config.xml*. Tento soubor obsahuje informace o jednotlivých streamech, uloženy jsou zde seznamy linků, které se testují. Dále konfigurační soubor obsahuje seznam testovacích souborů, které se mají streamem zpracovat. Na základě nastavení tohoto souboru se pak aplikací zpracují požadovaná data. Data jsou poslána do vstupního bufferu kolektoru odpovídajícího streamu. Následně je zapnut stream, který začne data zpracovávat. Zapnutí streamu je provedeno pomocí nástroje *tools.pl*, jedná se o multifunkční nástroj, jež dokáže např. exportovat konfigurace jednotlivých uzlů či konfiguraci celého streamu. Lze skrze něho spouštět či vypínat streamy a také je monitorovat. Obrázek 4.1 zjednodušeně demonstruje potřebné nastavení pro využití nástroje tools.pl. Na prvním řádku obrázku lze vidět podobu příkazu a parametrů nutných pro spuštění streamu. Pro spouštění externího procesu je použita funkce *ProcessBuilder*.

```
// streamCommand = /home/el7/elink/EventLink/base/bin/tools.pl -c startstream -n
public static String RunProc(List<String> streamCommand) throws IOException, InterruptedException {
        ProcessBuilder pb = new ProcessBuilder().command(streamCommand);

        pb.environment().put("EVENTLINK_HOME", "/home/el7/elink/EventLink/base");
        pb.environment().put("JAVA_HOME", "/home/el7/elink/java_jre/current");
        pb.environment().put("ORACLE_HOME", "/home/el7/oracle/app/el7/product/11.2.0/client_1");
        pb.environment().put("RCFILE", "/home/el7/elink/.mds.rc");
        pb.environment().put("PERL5LIB", "/home/el7/oracle/app/el7/product/11.2.0/client_1/perl/lib/5.10.0");
        pb.environment().put("LD_LIBRARY_PATH", "/home/el7/elink/EventLink/base/lib:"
                + "/home/el7/elink/common/perl/lib/5.8.9/x86_64-linux-thread-multi/CORE");

        // process start
        Process process = pb.start();
        // wait for the end of the process
        process.waitFor();
        String result = output(process.getInputStream());
        return result;
    }
```

Obrázek 4.1: Použití nástroje pro řízení streamů

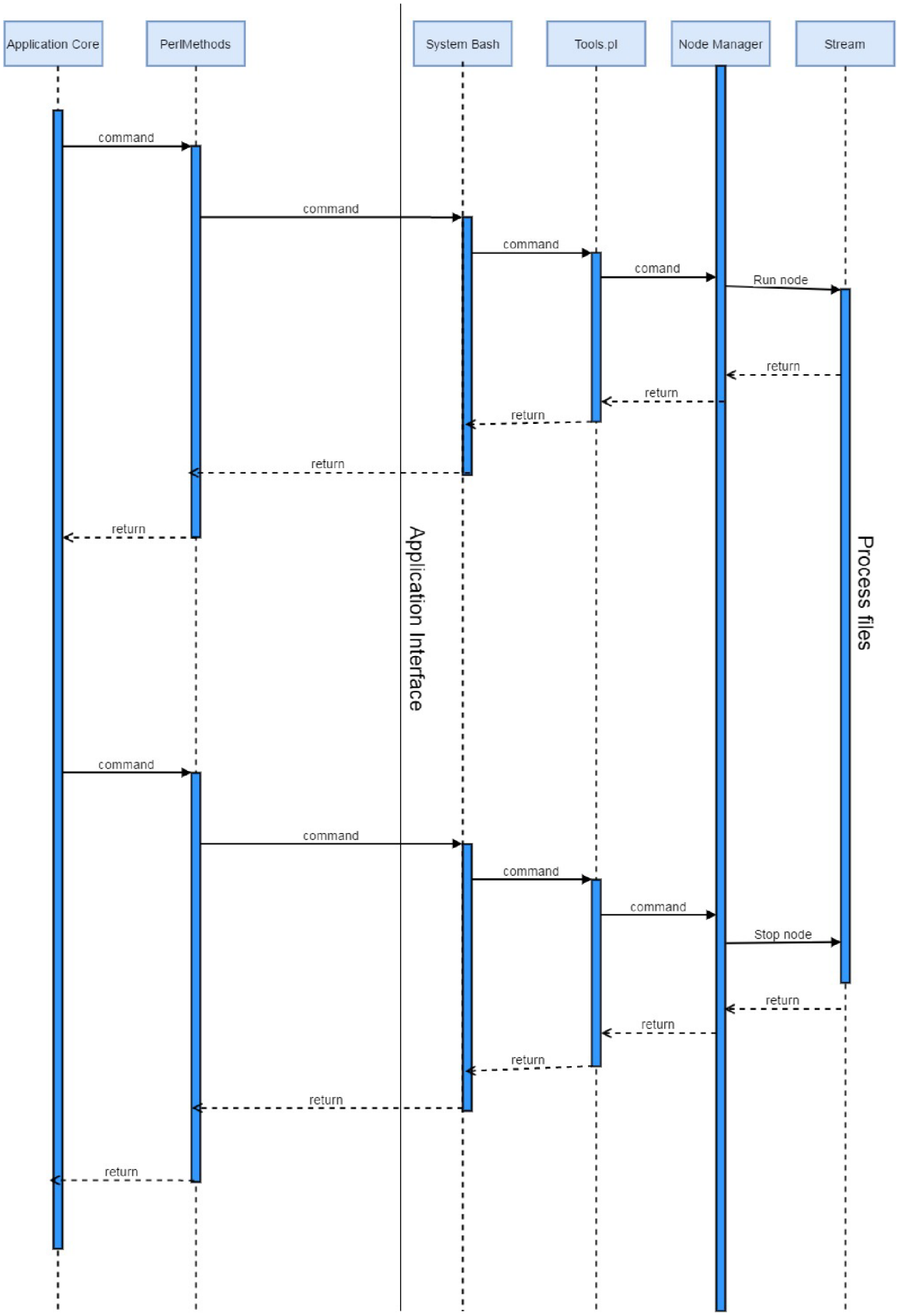
Jako ilustrace volání mezi částmi programu a dalším softwarem poslouží sekvenční diagram na obrázku 4.1. Třída *PerlMethods* je zodpovědná za řízení a komunikaci s externími elementy aplikace. Režii tedy má v momentě, kdy je třeba změnit stav streamu. Příkazy jsou pomocí subprocesů vytvořených pomocí bashe realizovány voláním nástroje *tools.pl*.

Příkazy jsou následně předány manažeru uzlů (Node manager), který řídí chování streamů. Node manager běží po celou dobu. Pokud byl příkaz úspěšně vykonán, node manager pošle výsledek operace volajícímu a obdobně se chovají i vyšší úrovně.



Obrázek 4.2: Přehled zdrojových souborů potřebných pro funkci uzlu

Hierarchicky druhá úroveň aplikace je tvořena především skripty volanými z hlavní aplikace. Jak již bylo zmíněno dříve, data zpracovaná streamem je třeba upravit do podoby a formátu, se kterým bude možno dále pracovat (výstup streamu je v podobě sekvencí znaků bez mezer a jakéhokoliv označení funkce dat, což není vhodné pro další zpracování aplikací). Pro tento účel je vhodné využít již existující možnosti daného prostředí. Proto je zde využit uzel s funkcí dekodéru. Strukturu uzlu demonstruje obrázek 4.2. Protože pro každý výstupní link z distributoru je obsah dat rozdílný, je třeba pro každý tento link samostatný dekodér. Každý z dekodérů má vlastní konfigurační část. Pro ilustraci dat, které dekodér zpracovává, slouží obrázek 4.4. Podobu dat změněných pomocí dekodéru ukazuje obrázek 4.5. Část kódu dekodéru je potom možné vidět na obrázku 4.6. Lze vidět, že framework uzlů pro EventLink nám umožňuje širokou funkčnost. Hlavní funkce programu *node_process()* provádí načtení dat (*nb_get_input_binary()*) a dekóduje každý řádek souboru. Funkce *nb_new_record()* vytvoří nový záznam a *o_add_field()* přidá položku do záznamu. Nakonec, *nb_write_record()* zapíše celý záznam do souboru.



Obrázek 4.3: Sekvenční diagram znázorňující vzájemné propojení mezi programem a prvky systému

```
1  HDR20170411100341014700000003MEDIATION       1.1  SMS00000196
2  SMS0320601          32477141611          32476141613          2368920170530075633          0160003247          543402443       071047000001
3  SMS0320601          32477141611          32476141613          2312520170531075633          0160003247          52559134        071047000001
4  SMS0320601          32477141611          32476141613          2312520170531075633          0160003247          52559134        071047000001
5  TRL20170411104014014700000003MEDIATION       1.1  000001960000000003000000000000000000000000000000000000000000000000000000000000
6
```

Obrázek 4.4: Příklad vstupních dat pro dekodér

```
1  RECORD
2  #input_id 1491901225x001_1
3  #output_id
4  #input_type
5  #output_type HNG_ROAM_RECORD
6  #addkey
7  #source_id ||272
8  #filename HNG001NGN20170411103838.000544.raw.HNG001NGN20170411103838.001859.sms
9  F HEADER SMS
10 F EXTERNAL_ID_TYPE 03
11 F CALLING_IMSI 20601
12 F CALLING_NUMBER 32477141611
13 F CALLED_NUMBER 32476141613
14 F USAGE_TYPE 23689
15 F MESSAGE_SUBMIT_TIME 20170530075633
16 F MESSAGE_DELIVERY_TIME
17 F NUMBER_OF_UNITS 0160
18 F COMPLETITION_STATUS 00
19 F ORIGINATED_MSC 3247
20 F TERMINATED_MSC 543402443
21 F RECORD_ID 071047000001
22
```

Obrázek 4.5: Exportovaná data po průchodu dekodérem

```
sub node_process() {
    nb_diagnostic( "APP", DIAG_LOW, "node_process(): entered the function" ) if $Diag_Low;

    my $bin = nb_get_input_binary();
    if ( $bin == -1 ) {
        nb_msg1( 'BMBDISDEC001', nb_get_original_filename() );
        nb_file_discard_input();
        return;
    }

    my $line;
    my $record_nr = 0;
    my $filesize = nb_file_size($bin);
    my $intLineLength;

    do {
        $line = read_line($bin);
        ($line) = $line =~ /([\S ]*)/;
        $intLineLength = length($line);
        if ( $intLineLength == RECORD_LENGTH  ){
            $HEADER                                 = substr($line, 1 ,  3 );   # 3
            $EXTERNAL_ID_TYPE                       = substr($line, 4 ,  2 );   # 2
            $CALLING_IMSI                           = substr($line, 6 ,  16 );  # 16
            $CALLING_NUMBER                         = substr($line, 22 ,  21 ); # 21
            $CALLED_NUMBER                          = substr($line, 43 ,  21 ); # 21
            $USAGE_TYPE                             = substr($line, 64 ,  5 );  # 5
            $MESSAGE_SUBMIT_TIME                    = substr($line, 69 ,  14 ); # 14
            $MESSAGE_DELIVERY_TIME                  = substr($line, 83 ,  14 ); # 14

            nb_new_record();
            audit_out_type("HNG_ROAM_RECORD");

            o_add_field( "HEADER", $HEADER );
            o_add_field( "EXTERNAL_ID_TYPE", $EXTERNAL_ID_TYPE );
            o_add_field( "CALLING_IMSI", $CALLING_IMSI );
            o_add_field( "CALLING_NUMBER", $CALLING_NUMBER );
            o_add_field( "CALLED_NUMBER", $CALLED_NUMBER );
            o_add_field( "USAGE_TYPE", $USAGE_TYPE );
            o_add_field( "MESSAGE_SUBMIT_TIME", $MESSAGE_SUBMIT_TIME );
            o_add_field( "MESSAGE_DELIVERY_TIME", $MESSAGE_DELIVERY_TIME );

            nb_write_record(DECOUT);
```

Obrázek 4.6: Část zdrojového kodu dekodéru

Takto vytvořená data jsou dále přesunuta a zpracována. Soubory je nutné převést z binárního formátu, zde je využit skript *r_export*, jež převede binární data na textová (Obrázek 4.5 už je v převedené formě). Soubory jsou poté upraveny skripty *r_sort*, *r_2lines*, *r_lines2cdrs*.(R_sort) skript seřadí jednotlivé položky abecedně v každém záznamu, jež soubor obsahuje. Samotné seřazení záznamů v rámci souboru je pak provedeno pomocí skriptu (r_2lines). Následné převedení souboru do vhodného formátu je provedeno pomocí (r_lines2cdrs). Příklad volání a použití těchto skriptů demonstruje obrázek v příloze A.

Kapitola 5

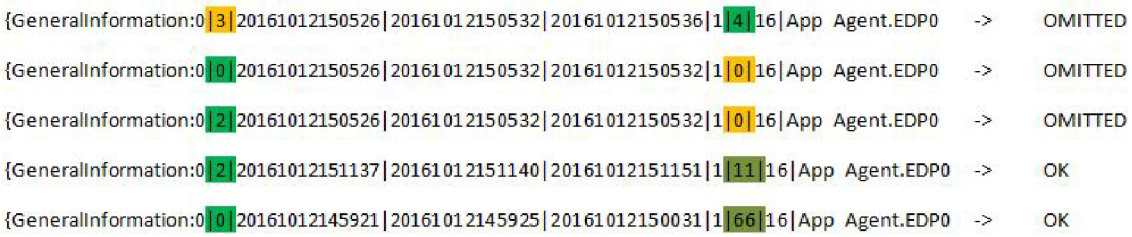
Testování

Regresní testování je pojem označující způsob testování založeném na provedení množiny testů nad aplikací, která již byla dříve otestována. Vykonání těchto testů je provedeno po implementaci změny v dané aplikaci. Cílem testů je ověřit, že nedošlo změnou programu k zanesení chyby, případně tuto chybu odhalit [3]. Regresní testování je zpravidla automatizované, probíhá pravidelně v době, kdy došlo k implementačním změnám a je dostatečný časový úsek pro provedení testů [9]. Regresní testování se skládá z testovacích scénářů, kdy každý scénář testuje určitou část funkcionality aplikace. Množství těchto testovacích scénářů přitom musí být vhodně zvoleno tak, aby splňovalo především dva požadavky. První požadavek je, že regresní testování nesmí běžet příliš dlouho, aby nenarušilo vývojový cyklus softwaru. Regresní testy se provádějí zejména přes noc, je tedy nutné, aby jejich běh skončil nejpozději v ranních hodinách, aby nedocházelo k prodlevám. Druhý požadavek je naopak založen na tom, aby testy byly natolik rozsáhlé a odhalily všechny kritické chyby [14].

Aplikace pro testování streamů v prostředí Comptel EventLink lze použít jako nástroj pro regresní testování (tedy porovnání nových výstupů aplikace s výstupy referenčními). Tento program lze však využít nejen pro regresní testování, ale i pro testování nově implementovaných změn. Doposud nebyl v daném prostředí žádný regresní ani jiný testovací systém přítomen. Také zde není žádná konzistentní databáze testovacích dat. Testování probíhalo manuálně za pomoci omezeného množství testovacích dat, která nepostihla dostatečnou funkcionalitu streamu a tím pádem docházelo k častému výskytu chyb ve vytvořeném softwaru. Nástroj popsaný v této práci si klade za úkol výrazně zvýšit kvalitu produkovaného softwaru. Toho je docíleno především pomocí vytvoření konzistentní databáze obsahující množinu testovacích případů pro každý stream. Množina těchto testů je srozumitelně zdokumentována a výsledky jsou popsány tak, aby bylo možné testy nadále modifikovat. Aplikace následně za pomocí streamu zpracuje testovací soubory a vytvoří výstupní data. Tyto data jsou pak porovnány s referenčními výstupy. Software dokáže odhalit změny provedené v nové verzi a upozorní na ně. Poskytuje také porovnání s různým stupněm detailů.

5.1 První testovací příklad

Každý stream obsahuje skupinu testovacích případů. Každý testovací případ testuje část funkčnosti streamu. Obrázek 5.1 ukazuje část jednoho testovacího souboru. Ke každému testovacímu souboru náleží jeho popis, který usnadňuje použití a údržbu testů. Vzorový



Obrázek 5.1: Část souboru určeného k testování na základě vyznačených atributů

popis ilustruje obrázek 5.2. Tento popis ukazuje reálný požadavek na změnu funkcionality streamu. Jedná se o změnu uzlu zpracovávajícího záznamy, které spadají do roamingových hlasových služeb (kvůli regulaci EU 531/2012 se roaming od 15. června nevztahuje na státy evropské unie [13] - tedy ani na tento stream). Nová verze streamu má implementovat změny, kde záznamy roamingových hovorů (*GENERAL_INFORMATION.FLOW_TYPE = '0' or '2'*) o nulové délce (*GENERAL_INFORMATION.CALL_DURATION = '0'*) mají být zahozeny. Tyto záznamy nejsou v dalších systémech nadále potřeba. V obrázku 5.1 značí první sloupec typ služby (hodnoty 0 a 2 pro roaming), druhý ze zvýrazněných sloupců pak označuje dobu trvání služby. Z obrázku vyplývá, že by mělo dojít k zahození prvních tří záznamů.

```
################################################################################################################################
test_001
        Filename:       CNGIN_VOICE_20161017150818_299170.txt
        CR:             CR2017_036_HNG
        Affected Nodes: Applicable for MO6 CDRs (HNG_GG_ROAM_BLN)
        Condition:      If ((GENERAL_INFORMATION.FLOW_TYPE = '0' or '2') and GENERAL_INFORMATION.CALL_DURATION = '0')
        Action:         OMITT record
        Desription:     Test file consists of 5 records. First 3 records should be omitted.

################################################################################################################################
test_002
```

Obrázek 5.2: Příklad popisu testovacího souboru

Data, jež vytváří stream a dále posílá distributor, ilustruje obrázek 5.3. Na obrázku je pouze část výsledných dat. Lze vidět, že ve výsledku figurují pouze dva záznamy, další informace se zde dají určit s obtížemi nebo vůbec.

```
20161012151137201610121511510    0          11          0    0                32475151230
20161012145921201610121500310    0          66          0    0                3222026630
```

Obrázek 5.3: Část dat zpracovaných streamem

Testovací aplikace tyto jinak obtížně čitelná data upraví do vhodné podoby. Na obrázku 5.4 potom vidíme přehled o výsledku testu. Výsledek zobrazuje jméno testu a jeho systémovou cestu. Pro každý výstupní link generující výstupní data, vypíše krátký přehled o zpracovaných záznamech. Je zde uveden počet výstupních záznamů. Dva vytvořené záznamy odpovídají očekávanému výsledku testu. Podle dalšího řádku se ve vytvořených záznamech neprovedla žádná změna. Na zvýrazněném řádku je pak informace, že v novém streamu je o dva záznamy méně. Link *standard* se používá pro zálohování vstupních dat. Zde nedošlo k žádné změně, což ani nebylo očekáváno. Podle tohoto testu je implementovaná změna streamu v pořádku.

Protože však předchozí informace nemusejí být dostatečné nebo je potřeba chyby lokalizovat, generuje program také detailní porovnání. Příklad takového porovnání zobrazuje obrázek 5.5. V detailním porovnání výsledků lze vidět jednotlivé položky záznamů. Na ob-

```
/home/el7/pamanekm/Regression_testing/test/HNG-Logica/input/test_001/CNGIN_VOICE_20161017150818_299170.txt

*****************************************************************************************************************
Output link: GG_ROAM_ENC_OUT
Number of Records: 2
Number of Failed Rec: 0
Number of OK Recs: 2
Record difference: -2


*****************************************************************************************************************
Output link: standard
Number of Records: 5
Number of Failed Rec: 0
Number of OK Recs: 5
Record difference: 0
```

Obrázek 5.4: Výsledek testu, porovnávající starou a novou verzi streamu

rázku jsou vidět části dvou záznamů (klíčové slovo RECORD). Zvýrazněné řádky s políčkem TOTAL_CHARGEABLE_UNITS obsahují hodnotu délky trvání hovorové služby. Je možné vidět, že záznam s nenulovou hodnotou byl zpracován také novou verzí streamu. Naopak záznam, kde je délka hovoru rovna nule, se v nové verzi streamu nevyskytuje. Na řádku 217 si lze ještě povšimnout pole RECORD_TYPE. Hodnota tohoto políčka - MO6 zastupuje roamingovou službu (dříve v textu také jako FLOW_TYPE).

```
152  F TOTAL_CALL_EVENT_DURATION 1                                  F TOTAL_CALL_EVENT_DURATION 1
153  F TOTAL_CHARGEABLE_UNITS 11                                    F TOTAL_CHARGEABLE_UNITS 11
154  F TOTAL_CHARGED_UNITS 0                                        F TOTAL_CHARGED_UNITS 0
155  F TOTAL_TAX_RATING                                             F TOTAL_TAX_RATING
156  F TRANSACTION_ID 21000001                                      F TRANSACTION_ID 21000001
157  F TRANSPARENCY_INDICATOR                                       F TRANSPARENCY_INDICATOR
158  F TYPE_OF_NUMBER 1                                             F TYPE_OF_NUMBER 1
159  F USAGE_TYPE 57155                                             F USAGE_TYPE 57155
160  .                                                              .
161  RECORD                                                     <
162  #addkey                                                    <
163  #filename HNG001NGN20170429175651.000588.raw.HNG001NGN2017042 <
164  #input_id 1493481543x001_1                                 <
165  #input_type                                                <
166  #output_id                                                 <
167  #output_type HNG_ROAM_RECORD                               <
168  #source_id ||7f2                                           <
169  F CALLED_COUNTRY_CODE   CAN                                <
170  F CALLED_NUMBER 32496768159                                <
171  F CALLED_NUMBER2 32496768159                               <
217  F RECORD_TYPE MO6                                          <
228  F THIRD_PARTY_NUMERICBER                                   <
229  F TIMEBAND                                                 <
230  F TIMESTAMP 20161012150526                                 <
231  F TIME_OFFSET   +02                                        <
232  F TOTAL_CALL_EVENT_DURATION 1                              <
233  F TOTAL_CHARGEABLE_UNITS 0                                 <
234  F TOTAL_CHARGED_UNITS 0                                    <
235  F TOTAL_TAX_RATING                                         <
236  F TRANSACTION_ID 19000001                                  <
237  F TRANSPARENCY_INDICATOR                                   <
```

Obrázek 5.5: Ukázka detailního porovnání výsledků streamu

Souhrnnou informaci naopak předkládá obrázek 5.6. Toto zobrazení je vhodné především pro rychlé zobrazení výsledků regresivních testů, kdy je přítomno větší množství testovacích případů. Tento formát výstupu slučuje výsledky ze všech testovacích případů. Tento stručný výpis obsahuje jméno a umístění vstupního testovacího souboru, název výstupního linku a informaci, zda došlo k vytvoření shodného výstupu v obou verzích streamu.

```
/home/el7/pamanekm/Regression_testing/test/HNG-Logica/input/test_001/CNGIN_VOICE_20161017150818_299170.txt
GG_ROAM_ENC_OUT
FAIL

standard
OK
```

Obrázek 5.6: Ukázka stručného porovnání výsledků streamu

GG.HNG.FILESTATUS	1
GG.HNG.METHODSTARTTIME	20170429175710
GG.HNG.FILEID	HNG001NGN20170429175651
GG.HNG.INPUTFILE1.NAME	HNG001NGN20170429175651.000588.std
GG.HNG.INPUTFILE1.FILESIZE	11821
GG.HNG.INPUTFILE1.NUMRECS	5
GG.HNG.INPUTFILE1.NUMUNITS	81
GG.HNG.NUMOUTPUTFILES	2
GG.HNG.OUTPUTFILE1.TYPE	20
GG.HNG.OUTPUTFILE1.NAME	HNG001NGN20170429175651.002981.MO6
GG.HNG.OUTPUTFILE1.NUMRECS	4
GG.HNG.OUTPUTFILE1.NUMUNITS	77
GG.HNG.OUTPUTFILE1.NUMCALLDATES	1
GG.HNG.OUTPUTFILE1.CALLDATE1	20161012
GG.HNG.OUTPUTFILE1.CALLDATE1.NUMRECS	4
GG.HNG.OUTPUTFILE1.CALLDATE1.NUMUNITS	77
GG.HNG.OUTPUTFILE2.TYPE	22
GG.HNG.OUTPUTFILE2.NAME	HNG001NGN20170429175651.002981.GG_ROAM_HNG.CV
GG.HNG.OUTPUTFILE2.NUMRECS	1
GG.HNG.OUTPUTFILE2.NUMUNITS	0
GG.HNG.OUTPUTFILE2.NUMCALLDATES	1
GG.HNG.OUTPUTFILE2.CALLDATE1	20161012
GG.HNG.OUTPUTFILE2.CALLDATE1.NUMREASONCODE	1
GG.HNG.OUTPUTFILE2.CALLDATE1.REASONCODE1.COD	UNB_USAGE_TYPE
GG.HNG.OUTPUTFILE2.CALLDATE1.REASONCODE1.NUMRECS	1
GG.HNG.OUTPUTFILE2.CALLDATE1.REASONCODE1.NUMUNIT	4
GG.HNG.METHODENDTIME	20170429175710

Obrázek 5.7: Rekonciliace uzlu z nového streamu

GG.HNG.FILESTATUS	1
GG.HNG.METHODSTARTTIME	20170430162746
GG.HNG.FILEID	HNG001NGN20170430162726
GG.HNG.INPUTFILE1.NAME	HNG001NGN20170430162726.000592.std
GG.HNG.INPUTFILE1.FILESIZE	11821
GG.HNG.INPUTFILE1.NUMRECS	5
GG.HNG.INPUTFILE1.NUMUNITS	81
GG.HNG.NUMOUTPUTFILES	2
GG.HNG.OUTPUTFILE1.TYPE	20
GG.HNG.OUTPUTFILE1.NAME	HNG001NGN20170430162726.003021.MO6
GG.HNG.OUTPUTFILE1.NUMRECS	2
GG.HNG.OUTPUTFILE1.NUMUNITS	77
GG.HNG.OUTPUTFILE1.NUMCALLDATES	1
GG.HNG.OUTPUTFILE1.CALLDATE1	20161012
GG.HNG.OUTPUTFILE1.CALLDATE1.NUMRECS	2
GG.HNG.OUTPUTFILE1.CALLDATE1.NUMUNITS	77
GG.HNG.OUTPUTFILE2.TYPE	22
GG.HNG.OUTPUTFILE2.NAME	HNG001NGN20170430162726.003021.GG_ROAM_HNG.CV
GG.HNG.OUTPUTFILE2.NUMRECS	1
GG.HNG.OUTPUTFILE2.NUMUNITS	0
GG.HNG.OUTPUTFILE2.NUMCALLDATES	1
GG.HNG.OUTPUTFILE2.CALLDATE1	20161012
GG.HNG.OUTPUTFILE2.CALLDATE1.NUMREASONCODES	1
GG.HNG.OUTPUTFILE2.CALLDATE1.REASONCODE1.CODE	UNB_USAGE_TYPE
GG.HNG.OUTPUTFILE2.CALLDATE1.REASONCODE1.NUMRECS	1
GG.HNG.OUTPUTFILE2.CALLDATE1.REASONCODE1.NUMUNITS	4
GG.HNG.OMITTED.NUMRECS	2
GG.HNG.OMITTED.NUMUNITS	0
GG.HNG.OMITTED.NUMCALLDATES	1
GG.HNG.OMITTED.CALLDATE1	20161012
GG.HNG.OMITTED.CALLDATE1.NUMREASONCODES	1
GG.HNG.OMITTED.CALLDATE1.REASONCODE1.CODE	UNB_DURATION
GG.HNG.OMITTED.CALLDATE1.REASONCODE1.NUMRECS	2
GG.HNG.OMITTED.CALLDATE1.REASONCODE1.NUMUNITS	0
GG.HNG.METHODENDTIME	20170430162746

Obrázek 5.8: Rekonciliace uzlu z původního streamu

Posledním informačním zdrojem pro určení správnosti je soubor obsahující rekonciliaci. Na obrázku 5.7 je zobrazena rekonciliace pro referenční výstup. Obrázek 5.8 naproti tomu ukazuje rekonciliaci pro novou implementaci streamu. Žlutě zvýrazněná pole představují počet záznamů, jež vstoupily do uzlu. Zelená pole obsahují hodnoty validně zpracovaných záznamů. Tyto hodnoty jsou v souladu s dosavadním vyhodnocením. Pro novou verzi přibývá modře zvýrazněné pole *OMITTED.NUMRECS*, které obsahuje počet zahozených záznamů. Červené políčko *NUMREASONCODES* vyjadřuje celkový počet důvodů, proč byly záznamy zahozeny. Následuje výčet těchto důvodů pro zahození záznamů. Šedě označené pole *REASONCODE1.CODE* obsahuje kód, který určuje proč byl záznam zahozen. Růžově zvýrazněný řádek *REASON_CODE1.NUMRECS* potom vyjadřuje počet zahozených záznamů pro daný důvod zahození. V tomto případě byly zahozeny dva záznamy. Oba záznamy měly stejný důvod zahození (*UNB_DURATION*), který v tomto případě odpovídá důvodu požadovaného ve specifikaci pro implementaci změny. Protože všechny výše zmíněné kontroly dopadly podle očekávání, lze zkonstatovat, že změny jsou správně implementovány a produkt může být puštěn do oběhu.

5.2 Druhý testovací případ

Ve druhém testu se bude simulovat zanesení chyby v implementaci konfiguračního souboru pro byznys logiku. Testovací případ bude testovat správnost mapování jednoho z políček výstupního záznamu. Důležitá bude pouze část hodnot tvořících jednotlivé záznamy. Jednou z funkcí byznys logiky je *enrichment* neboli obohacení záznamů o data, která nejsou přímo obsažena v příchozím záznamu. Přiřazení těchto hodnot podle určitých pravidel je jedním z úkolů mediace. Jedním z typů obohacení záznamů je vyhledání doplňujících dat v lookup tabulce na základě vyhledávacího klíče, složeného z prakticky libovolného množství polí daného záznamu. Příklad takové lookup tabulky je znázorněn na obrázku 5.9. Dotaz do této tabulky je složen ze sedmi hodnot. Návratovou hodnotou jsou potom hodnoty ze sloupce USAGE_TYPE (sloupeček s modrou šipkou). Tento test je zaměřen především na mapování hodnoty BUSINESS_ID, která má ze všech hodnot tvořících vyhledávací klíč nejsložitější mapování.

Stručný popis testu je stejně jako popis všech ostatních testovacích případů uveden v souhrnném dokumentu. Popis je možné vidět na obrázku 5.10. Z obrázku vyplývá, že testovací případ obsahuje osm záznamů. V případě použití tabulky z obrázku 5.9 a využití odpovídajícího nastavení vstupních záznamů streamu, by se měl test zachovat tak, že bude vytvořeno sedm záznamů. Každý z těchto sedmi záznamů by měl obsahovat odlišnou hodnotu parametru USAGE_TYPE. Poslední záznam bude zahozen z důvodu nenalezení odpovídající hodnoty v lookup tabulce.

FLOW_TYPE	ROAM_FLAG	BUSINESS_ID	NETWORK_ID	DESTINATION_ID	DESTINAT...	USAGE...	DESC...	START_D...	END_DATE
3	0	FAMCUG_ONNET	11111	*	ANY	11111	TEST	160901	990101
3	0	FAMCUG_ONNET	55555	*	ANY	11111	TEST	160901	990101
3	0	FAMCUG_OFFNET	22222	*	ANY	22222	TEST	160901	990101
3	0	PCS	33333	*	ANY	33333	TEST	160901	990101
3	0	VPN_ONNET	44444	*	ANY	44444	TEST	160901	990101
3	0	VPN_OFFNET	55555	*	ANY	55555	TEST	160901	990101
3	0	OCN	66666	*	ANY	66666	TEST	160901	990101
3	0	REGULAR_POSTPAID	77777	*	ANY	77777	TEST	160901	990101

Obrázek 5.9: Příklad lookup tabulky pro vyhledávání doplňujích dat záznamů

```
################################################################################################################
;
 Filename:       CNGIN_VOICE_20161017150818_000002.txt
 CR:             CR2017_033_HNG
 Affected Nodes: Applicable for .sms National sms CDRs (HNG_GG_NAT_BLN)
 Condition:      Rules in SOLA
 Action:         USAGE_TYPE parameter mapping test
 Desription:     Test file consists of 8 records.
                 Last record is omitted with reason code UNB_USAGE_TYPE.
                 Output contains of 7 recs.

                 SERVICE_SPECIFIC-FAMCUG-SERVICE_APPLICATION_ID |NOT NULL |NOT NULL |NOT NULL |         |         |         |
                 SERVICE_SPECIFIC-FAMCUG-DESTINATION_TYPE       |01       |02       |         |         |01       |         |01       |
                 SERVICE_SPECIFIC-VPN-CALL_TYPE                 |02       |03       |03       |03       |02       |         |04       |
                 SERVICE_SPECIFIC-VPN-SERVICE_APPLICATION_ID    |NOT NULL |NOT NULL |NOT NULL |NOT NULL |NOT NULL |NOT NULL |         |
                 SERVICE_SPECIFIC-PCS_SERVICE_APPLICATION_ID    |NOT NULL |NOT NULL |NOT NULL |         |         |         |         |
                 SERVICE_SPECIFIC-OCN-SERVICE_APPLICATION_ID    |NOT NULL |NOT NULL |NOT NULL |NOT NULL |NOT NULL |NOT NULL |         |

                 BUSSINES_ID                                    |FAMCUG_ONNET |FAMCUG_OFFNET |PCS |VPN_ONNET |VPN_OFFNET |OCN |REGULAR_POSTPAID

                 USAGE_TYPE                                     |11111        |22222         |33333 |44444     |55555      |66666 |7777777
```

Obrázek 5.10: Popis druhého testovacího případu

V popisu testu je taktéž uvedeno mapování pole BUSSINES_ID. Tomuto poli je přiřazena hodnota v závislosti na vypsaných parametrech. Žlutě zvýrazněné hodnoty jsou takové, které jsou v daném momentě rozhodující pro přiřazení správné hodnoty. Ostatní (nezvýrazněné) hodnoty parametrů by neměly mít na výsledek vliv, pokud je implementace konfigurace uzlu správná. Nicméně pokud je mapování nesprávně implementováno, díky těmto parametrům dojde ke změně výsledku a tím pádem k nalezení chyby. Tedy pokud parametr FAMCUG-SERVICE_APPLICATION_ID je nenulový a FAMCUG-DESTINATION-_TYPE je roven "01", pak je hodnota BUSSINES_ID rovna "FAMCUG_ONNET"a lookup tabulka v tomto případě najde pro parametr USAGE_TYPE hodnotu "11111", jež bude přítomna v koncovém záznamu. Pokud však podmínka není splněna, probíhá další test - zda je FAMCUG-SERVICE_APPLICATION_ID nenulový a FAMCUG-DESTINATION_TYPE roven "02". V případě splnění podmínky je BUSSINES_ID přiřazena hodnota "FAMCUG_-OFFNET"a následný dotaz na lookup tabulku vrátí hodnotu "22222"pro parametr USAGE_-ID. Pokud nedojde ke splnění této podmínky, testuje se další podmínka v pořadí. Každý z výsledných záznamů by pak měl obsahovat odlišnou hodnotu USAGE_TYPE.

Výsledek testování pak lze vidět na obrázku 5.11. Zde lze vidět, že počet průchozích záznamů odpovídá správnému výsledku testu (GG.HNG.OUTPUTFILE1.NUMRECS má hodnotu sedm). Dokonce i důvod k zahození osmého záznamu (žlutě zvýrazněný UNB_-USAGE_TYPE) má očekávanou hodnotu - UNB_USAGE_TYPE, znamenající, že v lookup tabulce nebyla nalezena shoda při hledání tohoto parametru.

```
1  GG.HNG.FILESTATUS                                    1
2  GG.HNG.METHODSTARTTIME                               20170504155338
3  GG.HNG.FILEID                                        HNG001NGN20170504155320
4  GG.HNG.INPUTFILE1.NAME                               HNG001NGN20170504155320.000620.std
5  GG.HNG.INPUTFILE1.FILESIZE                           18946
6  GG.HNG.INPUTFILE1.NUMRECS                            8
7  GG.HNG.INPUTFILE1.NUMUNITS                           0
8  GG.HNG.NUMOUTPUTFILES                                2
9  GG.HNG.OUTPUTFILE1.TYPE                              20
10 GG.HNG.OUTPUTFILE1.NAME                              HNG001NGN20170504155320.002621.sms
11 GG.HNG.OUTPUTFILE1.NUMRECS                           7
12 GG.HNG.OUTPUTFILE1.NUMUNITS                          0
13 GG.HNG.OUTPUTFILE1.NUMCALLDATES                      1
14 GG.HNG.OUTPUTFILE1.CALLDATE1                         20161012
15 GG.HNG.OUTPUTFILE1.CALLDATE1.NUMRECS                 7
16 GG.HNG.OUTPUTFILE1.CALLDATE1.NUMUNITS                0
17 GG.HNG.OUTPUTFILE2.TYPE                              22
18 GG.HNG.OUTPUTFILE2.NAME                              HNG001NGN20170504155320.002621.GG_NAT_HNG.CV
19 GG.HNG.OUTPUTFILE2.NUMRECS                           1
20 GG.HNG.OUTPUTFILE2.NUMUNITS                          0
21 GG.HNG.OUTPUTFILE2.NUMCALLDATES                      1
22 GG.HNG.OUTPUTFILE2.CALLDATE1                         20161012
23 GG.HNG.OUTPUTFILE2.CALLDATE1.NUMREASONCODES          1
24 GG.HNG.OUTPUTFILE2.CALLDATE1.REASONCODE1.CODE        UNB_USAGE_TYPE
25 GG.HNG.OUTPUTFILE2.CALLDATE1.REASONCODE1.NUMRECS     1
26 GG.HNG.OUTPUTFILE2.CALLDATE1.REASONCODE1.NUMUNITS    0
27 GG.HNG.METHODENDTIME                                 20170504155338
```

Obrázek 5.11: Rekonciliace uzlu testovaného v druhém testovacím případu

Pro rychlé vyhodnocení výsledku testování poslouží stručný výpis shrnující výsledky všech testovacích případů, který je ukázán na obrázku 5.9. Je patrné, že test z předchozí kapitoly dopadl úspěšně. Avšak testovací případ, kterému se věnuje tato podkapitola, dopadl odlišně než v předchozí verzi. Aby bylo možné určit, co je odlišné v nové verzi streamu, je nutno podívat se na detailnější výpis výsledku.

Výpis obsahující více detailů znázorňuje obrázek 5.9. Jak naznačuje žluté zvýraznění, jeden z výstupních záznamů se liší. Jedná se o rozdílnou hodnotu parametru USAGE_TYPE. Místo očekávané hodnoty "55555"obsahuje nově hodnotu "11111". Dá se předpokládat, že došlo k chybě při vyhledávání v lookup tabulce. Dotaz vedený na lookup tabulku vrátil od-

```
/home/el7/pamanekm/Regression_testing/test/HNG-Logica/input/test_001/CNGIN_VOICE_20161017150818_299170.txt
GG_ROAM_ENC_OUT
OK

standard
OK


/home/el7/pamanekm/Regression_testing/test/HNG-Logica/input/test_002/CNGIN_VOICE_20161017150818_000002.txt
GG_NAT_ENC_OUT
FAIL

GG_ROAM_ENC_OUT
OK

standard
OK
```

Obrázek 5.12: Stručný přehled výsledků druhého testovacího případu

lišnou hodnotu. To může být způsobeno více příčinami, například změnou dat přítomných v lookup tabulce nebo nekorektním sestavením klíče pro vyhledávání v tabulce. Vytvoření jiného klíče je pak důsledkem přiřazení jiných hodnot do jedné ze sedmi položek použitých pro sestavení vyhledávacího klíče.

```
/home/el7/pamanekm/Regression_testing/test/HNG-Logica/input/test_002/CNGIN_VOICE_20161017150818_000002.txt

*******************************************************************************************************************
Output link: GG_NAT_ENC_OUT
Number of Records: 7
Number of Failed Rec: 1
Number of OK Recs: 6
Record difference: 0

RECORD NUMBER: 3
    65  F USAGE_TYPE 55555                                          | F USAGE_TYPE 11111

*******************************************************************************************************************
Output link: GG_ROAM_ENC_OUT
Number of Records: 0
Number of Failed Rec: 0
Number of OK Recs: 0
Record difference: 0

*******************************************************************************************************************
Output link: standard
Number of Records: 8
Number of Failed Rec: 0
Number of OK Recs: 8
Record difference: 0
```

Obrázek 5.13: Výsledek druhého testovacího případu

V detailním zobrazení si lze prohlédnout celé záznamy, viz obrázek 5.14. Pokud se bude uvažovat obsah lookup tabulky na obrázku 5.9, pak je velice pravděpodobné, že došlo ke špatnému určení BUSINESS_ID parametru. Jedná se prakticky o jediné pole, které má odlišné hodnoty. Hodnota BUSINESS_ID pro USAGE_TYPE = "11111", je FAMCUG_ONNET. Protože hodnota FAMCUG_ONNET je přiřazena dříve než očekávaná hodnota VPN_OFFNET, lze usuzovat, že chyba bude v implementaci podmínky pro dosazení FAMCUG_ONNET jako hodnoty BUSINESS_ID parametru.

Tento testovací příklad názorně ukazuje podobu testovacího případu, pro regresní testování správné funkcionality streamu. Demonstruje podobu výsledků v případě, že došlo ke změně mapování hodnot pro výstupní záznam u nové verze streamu.

5.3 Unikátní parametry

Některé hodnoty polí ve výsledných záznamech nemusí být vždy stejné při dalším zpracování (ačkoliv vstupní data stejná jsou). Pokud se jedná o pole, která jsou společná záznamům

Obrázek 5.14: Detailní výpis druhého testovacího případu

ze všech streamů, výsledek porovnání těchto polí je ignorován v těle programu. Pokud se jedná o hodnoty polí, která jsou specifická pro každý typ záznamu, dochází k jejich ošetření při průchodu samostatným uzlem - dekodérem. Tato data jsou v případě potřeby validována, zda obsahují správný formát a následně je za ně dosazena hodnota určená k testování. Nejčastěji se jedná o pole obsahující data nebo sekvenční čísla, která jsou do záznamu přidána při zpracování testovacích dat streamem.

Kapitola 6

Závěr

Cílem práce bylo seznámení s regresním testováním v telekomunikacích. Následně na základě dostupných informací o testování a cílovém prostředí, byl vytvořen testovací nástroj, který provádí testování softwaru vytvořeného pomocí produktu EventLink. Kapitola 2 zařazuje aplikaci vytvořenou v této práci do širšího kontextu a věnuje se popisu prostředí, pro které je aplikace vyvýjena. Kapitola 3 se zabývá rozhraním a návrhem aplikace pro regresní testování. Jsou zde rozebrány jednotlivé funkční moduly aplikace a jejich využití. Kapitola 4 se věnuje detailům implementace aplikace pro regresní testování. Jsou zde vzaty v potaz specifika cílového systému a dostupných knihoven, z nichž některé z nich byly po pečlivém nastudování integrovány do těla aplikace. V kapitole 5 je pak na praktickém příkladu ukázáno vytvoření testů pro stream z platformy Eventlink. Dále je ukázán a vysvětlen výsledek těchto testů.

Telekomunikační sítě se od svého vzniku rozrůstají rychlým tempem a dá se očekávat, že v blízké budoucnosti nebude rychlost vývoje klesat. Protože množství a kvalita poskytovaných služeb stále roste, zvyšují se tím také požadavky na zlepšení komunikace a rozhraní mezi komponenty sítě. Jednou z komerčních možností pro zajištění správné komunikace je za pomoci EventLinku od firmy Comptel. Tato platforma však postrádá vestavěné prostředky určené k detailnímu testování. Tato práce si klade za úkol vytvořit univerzální nástroj určený k regresnímu testování. Nástroj obsahuje množinu testů, kdy každý z nich má za úkol otestovat část funkčnosti streamu. Shromažďuje a případně při změně udržuje testy ve své vlastní databázi založené na adresářové struktuře. Dále aplikace tyto testy automatizovaně spouští a generuje výsledky. Aplikace dokáže porovnat aktuální výsledky s referenčními (tedy takovými výsledky, které byly uznány jako korektní). Všechny odchylky jsou nalezeny a vypsány. Pomocí vyvinutého nástroje by se mělo docílit sníženého počtu chyb při zavádění produktů a tím docílit menšího množství defektů. Sníží se tak náklady a čas na údržbu softwaru.

Do budoucna se počítá s rozšířením aplikace tak, aby dokázala kromě aktuálních tří streamů otestovat všechny streamy na cílové platformě. Také by bylo vhodné vylepšit testování rekonciliačních souborů. V případě potřeby by bylo možné doplnit aplikaci o grafické uživatelské rozhraní.

Literatura

[1] Academy, C. N.: *Exploring the Modern Computer Network*. [Online; navštíveno 01.05.2017].
URL http://www.ciscopress.com/articles/article.asp?p=2158215&seqNum=6

[2] Alexandr, L.: *CGI Comptel Training*. Prosinec 2007.

[3] Andreas Spillner, H. S., Tilo Linz: *Software Testing Foundations*. Rocky Nook Inc, 2014, ISBN 978-1-937538-42-2.

[4] *ASN.1 Specification*. [Online; navštíveno 19.04.2017].
URL http://www.etsi.org/technologies-clusters/technologies/protocol-specification/asn-1?highlight=YToxOntpOjA7czo1OiJhc24uMSI7fQ==

[5] Bazala, D.: *Telekomunikace a VoIP telefonie*. BEN-Technická literatura, 2006, ISBN 80-7300-201-9.

[6] Coll, E.: *Telecom 101*. Teracom Training Institute, 2016, ISBN 978-1894887038.

[7] Free Software Foundation, I.: *GNU Bash*. [Online; navštíveno 12.05.2017].
URL https://www.gnu.org/software/bash/

[8] IEEE: *IEEE 5G Standards*. [Online; navštíveno 15.04.2017].
URL http://5g.ieee.org/standards

[9] Mauro Pezze, M. Y.: *Software Testing and Analysis*. Daniel Sayre, 2008, ISBN 978-0-471-45593-6.

[10] Mazda, F.: *Telecommunications Engineer's Reference Book*. Focal Press, 1993, ISBN 978-0750611626.

[11] Nishit Narang, S. K.: *2G Mobile Networks*. Tata McGraw-Hill Publishing Company Limited, 2007, ISBN 0-07-062106-3.

[12] Nolle, T.: *New telecom industry trends require OSS/BSS systems changes*. [Online; navštíveno 02.05.2017].
URL http://searchtelecom.techtarget.com/tip/New-telecom-industry-trends-require-OSS-BSS-systems-changes

[13] Parliament, T. E.: *Regulation (EU) 2015/2120 of the European Parliament and of the Council*. Listopad 2015, [Online; navštíveno 01.05.2017].
URL http://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:32015R2120

[14] Paul Amman, J. O.: *Introduction to Software Testing*. Cambridge University Press, 2008, ISBN 978-0-521-88038-1.

[15] Petr Jareš, J. V.: *Úvod do telekomunikačních sítí*. [Online; navštíveno 09.05.2017]. URL http://data.cedupoint.cz/oppa_e-learning/2_KME/053.pdf

[16] Pravda, I.: *Základy mobilních sítí*. [Online; navštíveno 15.04.2017]. URL http://data.cedupoint.cz/oppa_e-learning/2_KME/057.pdf

[17] Sathyan, J.: *Fundamentals of EMS, NMS and OSS/BSS*. Auerbach Publications, 2010, ISBN 978-1420085730.

[18] Vincent D Blondel, G. K., Adeine Decuyper: *EPJ Data Science*. Springer Berlin Heidelberg, 2015, doi:10.1140/epjds/s13688-015-0046-0.

Přílohy

Seznam příloh

Příloha A

Použití skriptů r__sort, r__2lines, r__lines2cdr

```
private static String Sort( String dir) throws IOException, InterruptedException {

        //     cat 1490944332x001_1_1490944344x001_1_3.exported| r_sort| r_2lines| sort| r_lines2cdrs> item
        String r_sort = "/home/el7/scripts/r_sort";
        String r_2lines = "/home/el7/scripts/r_2lines";
        String r_lines2cdrs = "/home/el7/scripts/r_lines2cdrs";
        String pipe = "| ";

        String myCommand = "ls " + dir;
        String myString = BashCommand.BashCommandReturn(myCommand);
        String[] ss = myString.split("\\s+");

        for (String item:ss) {

            // get new filename
            String comm = "cat " + item + pipe + "grep filename" + pipe + "awk '{print $2}'" + pipe + "head -1";
            ProcessBuilder pb1 = new ProcessBuilder().command("/bin/sh", "-c", comm);
            pb1.directory(new File(dir));

            // add cat, rdump etc....
            pb1.environment().put("PATH", "/bin:/usr/bin:/usr/sbin:/home/el7/elink/EventLink/base/bin:/home/el7/scripts");

            // process start
            Process process1 = pb1.start();

            // wait for the end of the process
            process1.waitFor();
            String newfilename = output(process1.getInputStream());

            if (newfilename.equals("")) {
                continue;
            }

            myCommand = "cat " + item + pipe + r_sort + pipe + r_2lines + pipe + "sort| " + r_lines2cdrs + "> " + newfilename;

            //ProcessBuilder pb = new ProcessBuilder().command("/bin/sh", "-c", myCommand);
            pb.directory(new File(dir));

            // add cat, rdump etc....
            pb.environment().put("PATH", "/bin:/usr/bin:/usr/sbin:/home/el7/elink/EventLink/base/bin:/home/el7/scripts");

            // process start
            Process process = pb.start();

            // wait for the end of the process
            process.waitFor();
            String result = output(process.getInputStream());

            // delete original file
            String file_to_delete = dir + "/"+ item;
            Path path = Paths.get(file_to_delete);
            Files.delete(path);
        }

        return result;
    }
```

Obrázek A.1: Použití skriptů pro seřazení dat v souboru

Příloha B

Obsah přiloženého CD

/Bakalarska_prace/ l složka obsahující dokumenty obsažené v textové části bakalářské práce.

/Regression_testing/ složka obsahující zdrojové kódy aplikace.

/xpaman01_RegresniTestovani.pdf PDF obsahující elektronickou verzi textové části bakalářské práce.

/README.txt návod k použití aplikace.