

**VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra telekomunikační techniky**

**Integrace Asterisku s podnikovými informačními systémy
Integration of Asterisk with Information Systems of Company**

2016

Bc. Vítězslav Miech

Zadání diplomové práce

Student: **Bc. Vítězslav Miech**

Studijní program: N2647 Informační a komunikační technologie

Studijní obor: 2612T059 Mobilní technologie

Téma: **Integrace Asterisku s podnikovými informačními systémy**
Integration of Asterisk with Information Systems of Company

Jazyk vypracování: čeština

Zásady pro vypracování:

Cílem diplomové práce je realizovat propojení Asterisku s informačním systémem tak, aby uživatel v IS byl identifikován na základě čísla volajícího, Asterisk směroval volání konkrétnímu uživateli dle informací v IS a poskytl mu informace o volajícím.

1. CRM systémy a jejich funkce.
2. Asterisk a možnosti integrace se CRM.
3. Návrh a realizace propojení Asterisku s databází o zákaznících.
4. Zhodnocení dosažených výsledků.

Seznam doporučené odborné literatury:

F. Piepiorra, *Vtiger C.R.M. Users and Administration Manual for v5.4.0*. Publisher: lulu.com, October 2012, ISBN-13: 978-1300260776.

R. Bryant, L. Madsen, J. Meggelen. *Asterisk: The Definitive Guide, 4th Edition, The Future of Telephony Is Now*. O'Reilly Media, 2013, 846 pages.

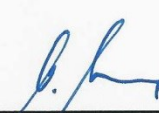
Formální náležitosti a rozsah diplomové práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí diplomové práce: **doc. Ing. Miroslav Vozňák, Ph.D.**


Datum zadání: 01.09.2014

Datum odevzdání: 29.04.2016





doc. Ing. Miroslav Vozňák, Ph.D.
vedoucí katedry



prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Prohlášení studenta

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě dne: *26. dubna 2016*



.....
podpis studenta

Poděkování

Rád bych poděkoval doc. Ing. Miroslavu Vozňákovi, Ph.D. za odbornou pomoc a konzultaci při vytváření této diplomové práce.

Abstrakt

Tato diplomová práce se zabývá integrací pobočkové ústředny Asterisk s informačními podnikovými systémy. Cílem je navrhnout a implementovat dialplán hovoru na support infolinku fiktivního podniku operující s Vtiger CRM systémem a komunikující přes VoIP technologii se softwarovou pobočkovou ústřednou Asterisk. Dále zobrazit uživateli CRM více informací při příchozím hovoru, analyzovat a testovat hovory a webovou aplikaci Vtiger Asterisk Connector. Teoretická část popisuje CRM systémy a porovnává tři vybrané open-source CRM systémy. Druhým stěžejným tématem je Asterisk a s ním popis jeho rozhraní pro komunikaci s externími aplikacemi a modulem pro ODBC umožňující integraci s relačními databázemi.

V praktické části je popsána veškerá konfigurace a implementace Vtiger CRM, PBX Asterisk, Vtiger Asterisk Connector a ODBC konektoru. Na základě exploračního modelu databáze je navržen a naimplementován dialplán, který přeměňuje hovory pomocí SQL dotazů nad databází informačního systému. Praktická část obsahuje i modifikaci notifikačního okna s přidáním informací z databáze. Na závěr je v práci popsán průběh testování a analýza hovorů.

Klíčová slova

Asterisk, AMI, AGI, CRM systémy, ODBC, relační databáze, SQL, VoIP, Vtiger CRM

Abstract

This master thesis is about integration of an Asterisk IP private branch exchange with the information systems of a company. The goal of this thesis is to design and implement an information line dialplan within a sample company operating a Vtiger CRM system using VoIP technology with their Asterisk IP PBX. Then display more information in incoming call notifier to user of the CRM, analyse and test of calls and Vtiger Asterisk Connector web application. The theoretical part of the thesis describes CRM systems and compares three chosen open-source CRM systems. The next main topic is about Asterisk and description of its interfaces for communication with external applications and module for ODBC connector which allows relational database integration.

The practical part describes the configuration and implementation of Vtiger CRM, PBX Asterisk, Vtiger Asterisk Connector and ODBC connector. The dialplan is designed and implemented based on the findings of database data model. The dialplan redirects calls using SQL queries on the database of information systems. The practical part also contains modification of notification window with added information from database. On the end of the thesis is described process of testing and analysis of calls.

Key words

Asterisk, AMI, AGI, CRM systems, ODBC, relational database, SQL, VoIP, Vtiger CRM

Obsah

Úvod.....	- 1 -
1 CRM systémy.....	- 3 -
1.1 Architektura CRM systémů.....	- 5 -
1.2 Funkce CRM systémů.....	- 5 -
1.3 Open-source řešení.....	- 6 -
1.3.1 Vtiger CRM.....	- 7 -
1.3.2 Zoho CRM.....	- 8 -
1.3.3 SuiteCRM.....	- 9 -
1.3.4 Výhody a nevýhody.....	- 10 -
2 Asterisk.....	- 11 -
2.1 Dialplán.....	- 11 -
2.2 SIP.....	- 12 -
2.3 Rozhraní AMI.....	- 13 -
2.4 Rozhraní AGI.....	- 14 -
2.4.1 AGI varianty.....	- 14 -
2.5 Rozhraní ARI.....	- 15 -
2.6 Integrace Asterisku s relačními databázemi.....	- 16 -
2.6.1 ODBC a dialplán funkce func_odbc.....	- 16 -
2.7 Integrace Asterisku se CRM.....	- 17 -
3 Integrace Asterisku s Vtiger CRM.....	- 19 -
3.1 Experimentální topologie.....	- 19 -
3.2 Implementace Vtiger CRM systému.....	- 20 -
3.3 Konfigurace Asterisk serveru.....	- 22 -
3.4 Implementace Vtiger Asterisk Connector.....	- 24 -
3.5 Integrace Asterisku s databází Vtiger CRM.....	- 25 -
3.6 Návrh a implementace dialplánu.....	- 28 -
3.7 Modifikace notifikace příchozího hovoru.....	- 35 -
4 Testování a analýza hovorů.....	- 39 -
Závěr.....	- 50 -
Použitá literatura.....	- 51 -
Seznam příloh.....	- 53 -

Seznam použitých symbolů

Symbol	Jednotky	Význam symbolu
t	s	Čas

Seznam použitých zkratek

Zkratka	Význam
AGI	Asterisk Gateway Interface
AJAM	Asynchronous Javascript Asterisk Manager
AJAX	Asynchronous JavaScript and XML
AMI	Asterisk Manager Interface
API	Application Programming Interface
ARI	Asterisk REST Interface
CLI	Command Line Interface
CRM	Customer Relationship Management
CTI	Computer Telephony Integration
DBMS	Database Management System
EAI	Enterprise Application Integration
ERP	Enterprise Resource Planning
HTML	HyperText Markup Language
HTTP	Hyper Text Transfer Protocol
ID	Identification
IP	Internet Protocol
ISDN	Integrated Services Digital Network
IAX	Inter-Asterisk Exchange Protocol
IT	Informační technologie
JDK	Java Development Kit
JSON	JavaScript Object Notation
NAT	Network Address Translation
ODBC	Open Database Connectivity
PBX	Private Branch eXchange
PHP	Hypertext Preprocessor
REST	Representational State Transfer
RPC	Remote Procedure Call
RTP	Real-time Transport Protocol

Seznam použitých zkratk

SaaS	Software as a Service
SDP	Session Description Protocol
SIP	Session Initiation Protocol
SMTP	Simple Mail Transfer Protocol
SOAP	Simple Object Access Protocol
SQL	Structured Query Language
SW	Software
tar	tape archive
TCP	Transmission Control Protocol
TDM	Time-Division Multiplexing
TLS	Transport Layer Security
UDP	User Datagram Protocol
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
VoIP	Voice over Internet Protocol
VPN	Virtual Private Network
XML	Extensible Markup Language

Seznam ilustrací a seznam tabulek

Číslo ilustrace	Název ilustrace	Číslo stránky
1.1	Graf procentuální spokojenosti s CRM systémem v podnicích	3
1.2	Graf procentuální problematiky CRM systémů v podnicích	4
1.3	Architektura CRM systému	5
1.4	Procentuální graf investic do CRM oblastí	6
1.5	Graf procentuálního využití CRM na určitých platformách	6
1.6	Konfigurace Asterisku ve Vtiger Phone Calls modulu	7
1.7	Konfigurace Asterisku v PBXManager modulu	8
1.8	Ukázka domovské stránky Zoho CRM	9
1.9	Ukázka domovské stránky SuiteCRM	10
2.1	Vztah mezi konfiguračními soubory ODBC pro Asterisk	17
3.1	Experimentální topologie	19
3.2	Systémová konfigurace při instalaci Vtiger CRM	21
3.3	Konfigurace Asterisk serveru ve Vtiger CRM	22
3.4	Vtiger Asterisk Connector startovní stránka běžící aplikace	25
3.5	Diagram hovoru zákazníka na support infolinku	29
3.6	Vztahy mezi entitami databáze vtigercrm	29
3.7	Defaultní notifikační okno příchozího hovoru ve Vtiger CRM	35
3.8	Modifikované notifikační okno příchozího hovoru ve Vtiger CRM	48
4.1	Vtiger CRM s notifikací o příchozím hovoru	46
4.2	Notifikační okno při příchodu neznámého hovoru ve Vtiger CRM	46
4.3	Ukázka Record List ve Vtiger CRM	47
4.4	Detail hovoru v záznamu Records List	47
4.5	Vytváření reportu Proběhlé hovory	48

Seznam ilustrací a seznam tabulek

4.6	Graf reportu Proběhlé hovory	48
Číslo tabulky	Název tabulky	Číslo stránky
2.1	AMI frameworky	14

Úvod

Podniky u nás i ve světě často operují s vlastním informačním systémem, kde mimo jiné uchovávají veškerá data o firmě, zaměstnancích a samozřejmě zákaznících. Ne všechny podniky ovšem mají zavedený vlastní informační systém a kromě účetního systému či programu využívají CRM systém pro zlepšení vztahů se zákazníky. Tyto CRM systémy zaznamenávají veškerá data o zákaznících a operacích s nimi souvisejících. V případě, že podnik komunikuje přes VoIP technologii, je zde integrita pobočkové ústředny s informačním systémem klíčová. Získané informace z databáze informačního systému umožňují pobočkově ústředně řídit hovory dle potřeb podniku a také naopak zobrazit informace uživateli informačního systému o zákazníkovi za pomoci zjištění identity zákazníka dle telefonního čísla. Právě integrace softwarové PBX Asterisk se CRM systémem je stěžejním tématem této diplomové práce.

Cílem této diplomové práce je vytvořit fiktivní podnik operující s vybraným open-source CRM systémem pro správu zákazníku a komunikující přes VoIP technologii se softwarovou pobočkovou ústřednou Asterisk. Dále návrh průběhu hovoru na support infolinku podniku a jeho realizaci v dialplánu za pomoci SQL dotazů nad databází CRM systému. Také rozšíření notifikačního okna o informace získané z databáze. Součástí je i testování hovorů, funkčnosti konektoru a jeho poskytovaných vlastností jako zaznamenávání logů, nahrávání hovorů a zapisování proběhlých hovorů do databáze.

První kapitola práce je věnována CRM systémům. Je zde popsán obecný popis CRM systémů, jejich architektura a funkce. Součástí této kapitoly je také porovnání tří vybraných open-source CRM systémů a jejich možnosti pro integraci s Asteriskem.

Druhá kapitola se věnuje samotnému Asterisku. Mimo hlavní informace, architekturu a kanály, je popsána i nejdůležitější část Asterisku – dialplán. Protože veškeré testovací hovory v této práci jsou provedeny přes SIP protokol, je i popis tohoto komunikačního protokolu součástí této kapitoly. V této práci je stěžejní část samotná integrace a komunikace Asterisku s informačními systémy, proto je zde dále popis různých Asterisk rozhraní jako AGI, AMI a ARI. Nakonec je vysvětlena funkce dialplánu `func_odbc` a modul `res_odbc` umožňující integraci Asterisku s databázovým systémem přes ODBC konektor.

Třetí kapitola se již věnuje samotné praktické části integraci Asterisku s Vtiger CRM systémem a jeho databází. Součástí je nasazení Vtiger CRM systému na virtuální server, nasazení a konfigurace pobočkové ústředny Asterisk na druhý virtuální server, nasazení a konfigurace Vtiger Asterisk Connectoru, integrace databáze Vtiger CRM s Asteriskem a konfigurace ODBC konektoru. Dále tato kapitola obsahuje návrh a realizaci dialplánu, tedy řízení hovorů. S dialplánem a integrací Vtiger databáze je úzce spojena explorace databáze Vtiger CRM a jejího datového modelu a poté implementace SQL dotazů nad zmíněnou databází. V poslední řadě úprava samotného Vtiger CRM a to konkrétně notifikačního okna s přidáním informací z databáze získané pomocí PHP a zobrazení díky technologiím jQuery a AJAX.

Poslední čtvrtá kapitola se zabývá testováním. Jednak testování CRM systému a to v oblastech interakce s uživatelem pomocí funkce Click to Dial, ukládání proběhlých hovorů v Call Records sekci a zobrazování vyskakovacího okna s informacemi o volajícím v případě příchozího hovoru. Dále

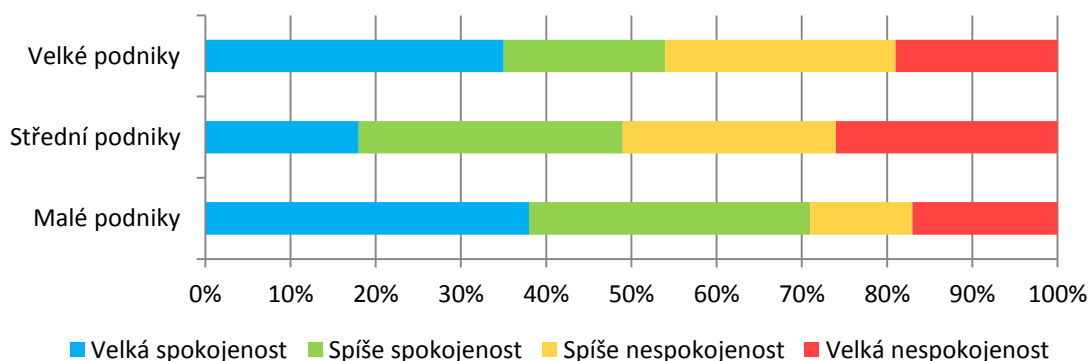
testování hovorů registrovaných a neregistrovaných zákazníků na infolinku podniku a přesměrování hovorů na dané operátory dle zjištěných informací získaných pomocí SQL dotazů. V poslední řadě zapisování logů a nahrávání hovorů na server.

1 CRM systémy

Řízení vztahů se zákazníkem neboli Customer Relationship Management je způsob, jakým lze identifikovat, získat a udržet si zákazníky, kteří představují největší obchodní aktivum společnosti. [1]

CRM systém je tedy technologický nástroj, který shromažďuje, zpracovává a využívá informace o klientech. Umožňuje poznat, pochopit a předvídat jejich potřeby, přání a nákupní zvyklosti. Jak z definice vyplývá, stěžejním úkolem CRM systému je podpora obchodních procesů. K tomu využívá základních funkcionalit, mezi které patří průběžné sledování požadavků zákazníků a jejich chování, evidence současných zákazníků a jejich hodnocení, vytváření nových obchodních příležitostí, udržování dlouhodobých vztahů se zákazníky, analýzy zákazníků na základě všech možných hledisek, a řízení marketingových kampaní na základě výše uvedených analýz a informací o zákaznících. [2]

CRM systémy nevyužívají jenom velké společnosti, ale i malé a středně velké podniky. Úroveň využití a spokojenosti jsou různé a podle výzkumu Software Advice z roku 2014 až překvapivé. Na obrázku 1.1 lze vidět, že největší spokojeností dosahují malé podniky (do 100 zaměstnanců). [3]

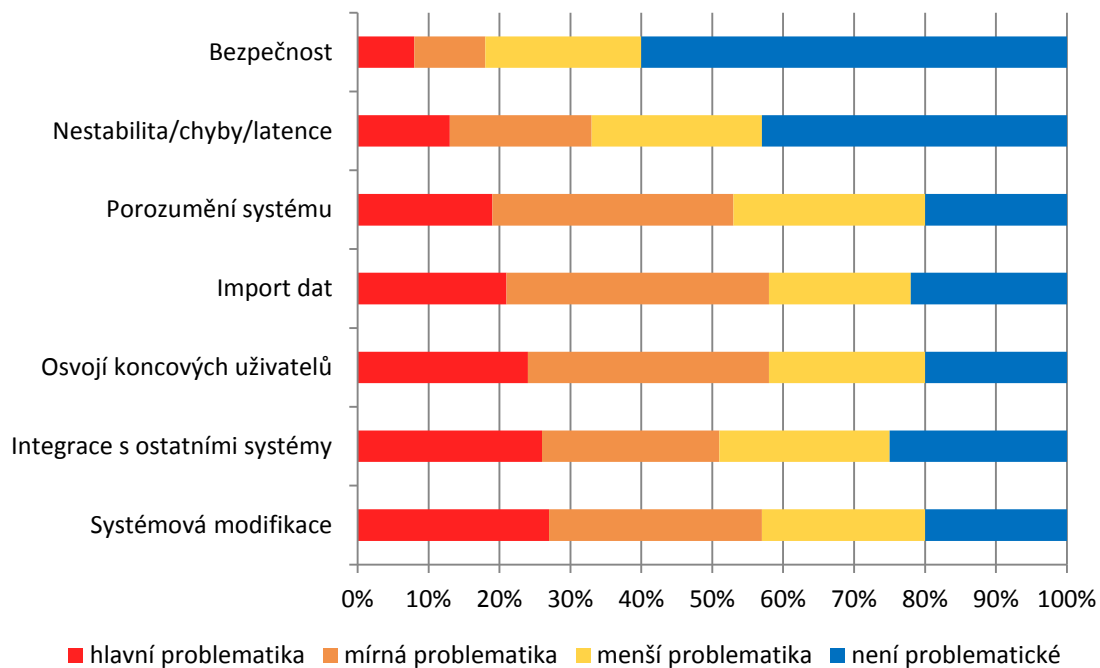


Obrázek 1.1: Graf procentuální spokojenosti s CRM systémem v podnicích[3]

CRM systémy by se rozhodně neměly distancovat od ostatních systémů v podniku. Propojení s informačním systémem, databází, call centrem apod. spojuje samostatné aplikace pro řízení služeb zákazníkům, marketingu a prodeje produktů do jednoho komplexního systému. Také ERP systém integruje aplikace, které byly dříve samostatné, příkladem mohou být aplikace pro zpracování objednávek, řízení logistiky (interní uvnitř podniku i externí k zákazníkům), řízení výroby produktů, fakturace, účetnictví a řízení plateb a další aplikace. V jádru celého podnikového informačního systému se nachází aplikace pro integraci podnikových aplikací (EAI), která zajišťuje sběr dat z jednotlivých součástí celého systému. Tyto data potom slouží k vytváření různých analýz a predikcí vývoje, které zajišťuje systém pro aplikaci řízení znalostí známý pod pojmem Business Intelligence. [2][4]

Integrace podnikového informačního systému se systémem CRM nemusí být vždy jednoduchý úkol. Obzvláště pokud podnik nemá veškerý software od jednoho dodavatele s plnou podporou servisní služby. To platí také u integrace dalších technologií a odvětví např. call centra. Majitelé podniků využívající CRM systém nejvíce čelí problematice přizpůsobení a integrace. Přizpůsobit CRM systém

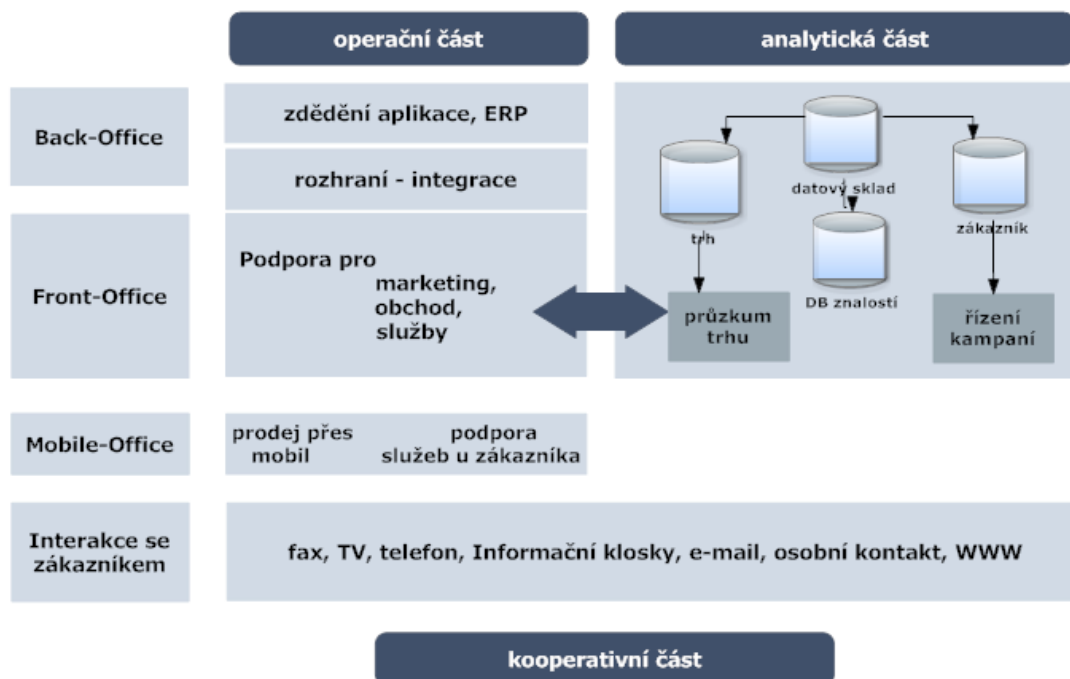
dle vlastních požadavků, obzvláště u open-source produktů, zabere nemálo času a je potřeba mít v podniku na tuto práci určité lidi. Totéž platí i u integrace, kde se však tato problematika řeší při zavedení CRM systému, a poté při dalších inovacích a implementacích nových technologií v podniku.



Obrázek 1.2: Graf procentuální problematiky CRM systémů v podnicích[3]

1.1 Architektura CRM systémů

Aplikační architektura CRM systémů se dělí na tři části – operační, analytická a kooperativní. Operační část je zaměřena na služby a funkce CRM, analytická pak pracuje s daty v systému a kooperativní je uváděná jako interakce se zákazníkem viz obrázek 1.3.



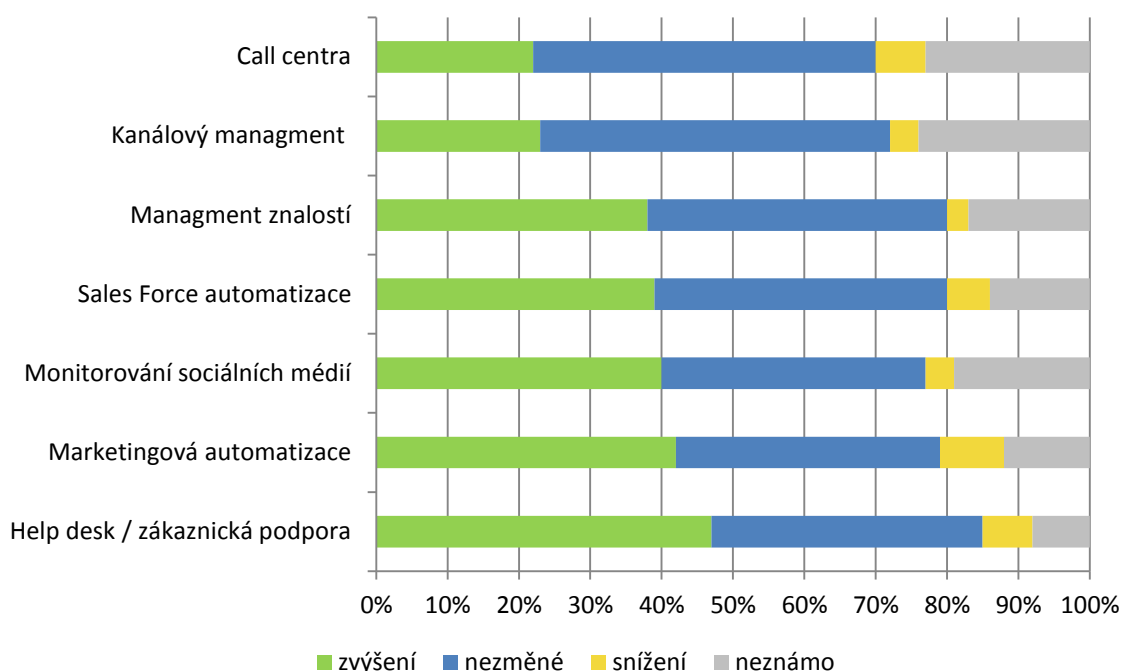
Obrázek 1.3: Architektura CRM systému[5]

1.2 Funkce CRM systémů

Každý CRM systém obsahuje několik funkcí a přídatných modulů. Mezi hlavní funkce CRM systému patří:

- Kontakty - průběžné sledování zákaznických požadavků a chování, evidence a hodnocení současných obchodních kontaktů,
- Obchod a plánování - vytváření nových obchodních příležitosti s využitím zmíněných informací,
- Analýza - analýzy zákazníků podle nejrůznějších hledisek, realizované prostřednictvím tzv. data miningu,
- Marketing - řízení marketingových kampaní s využitím výsledků zákaznických analýz a jejich požadavků.
- Komunikace - integrování e-mailového klienta, call centra, chat kontaktního centra atd. [6]

Graf obrázku 1.4 vyjadřuje procentuální investice do jednotlivých aplikačních funkcí CRM systému. Vyplyvá z něj, že ve výrazné většině podniků zůstaly náklady na podporu jednotlivých CRM funkcí a modulů stejné nebo se zvyšovaly, a to i v případě call center.

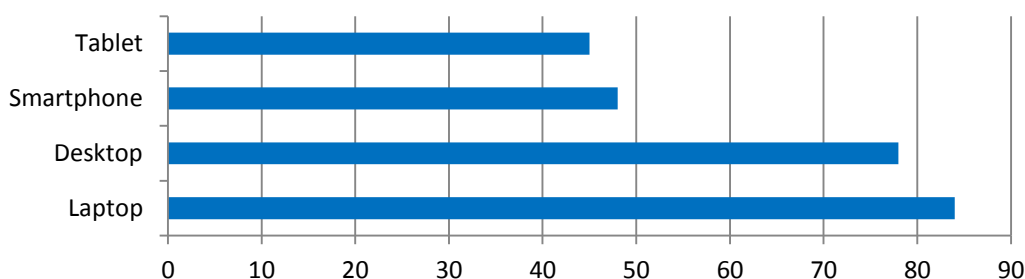


Obrázek 1.4: *Procentuální graf investic do CRM oblastí[3]*

Jelikož je v dnešní době nespočet CRM systémů, stále se vytvářejí nové funkcionality v závislosti na trendech doby. V posledních letech se CRM systémy především zaměřují na tyto funkcionality:

- mobilita
- sociálních sítě
- cloud computing a SaaS

Obzvláště mobilita v dnešní době rapidně stoupá, o čemž vypovídá i obrázek 1.5.[7]



Obrázek 1.5: *Graf procentuálního využití CRM na určitých platformách[3]*

1.3 Open-source řešení

Většina malých a středně velkých podniků si nemůže dovolit větší výdaj na zakoupení licence nákladného CRM systému jako Microsoft Dynamics CRM, SAP, Salesforce, Oracle CRM a další. Další možností je zakoupení CRM systému jako vzdálenou cloudovou službu za měsíční poplatek. Ovšem jako u jiných softwarů i mezi CRM systémy existují open-source verze, které lze zdarma využívat pro své obchodní strategie v podniku. Kromě open-source verzí existují i omezené verze

plnohodnotných placených CRM. Ovšem open-source řešení není nikterak omezeno, např. počtem uživatelů či velikosti úložiště. V dnešní době existuje nespočet open-source CRM systémů a vybrat ten správný je mnohdy nelehký úkol. V mé práci se zaměřuji především na integraci s pobočkovou telefonní ústřednou Asterisk, a proto jsem vybíral z kandidátů, kde není integrita s Asteriskem problematická. Zde uvádím mnou vybrané tři kandidáty vhodných open-source CRM systémů do malých a středních podniků k integraci s Asteriskem:

1.3.1 Vtiger CRM

Vtiger CRM systém byl založen jako open-source větev SugarCRM systému v roce 2004. Později se však osamostatnil, ale i dodnes využívá SugarCRM licenci. Vtiger je postaven na jazyce PHP, "běžící" na Apache web serveru a pracuje s MySQL databází. Vtiger je multiplatformní co se týče operačních systémů.

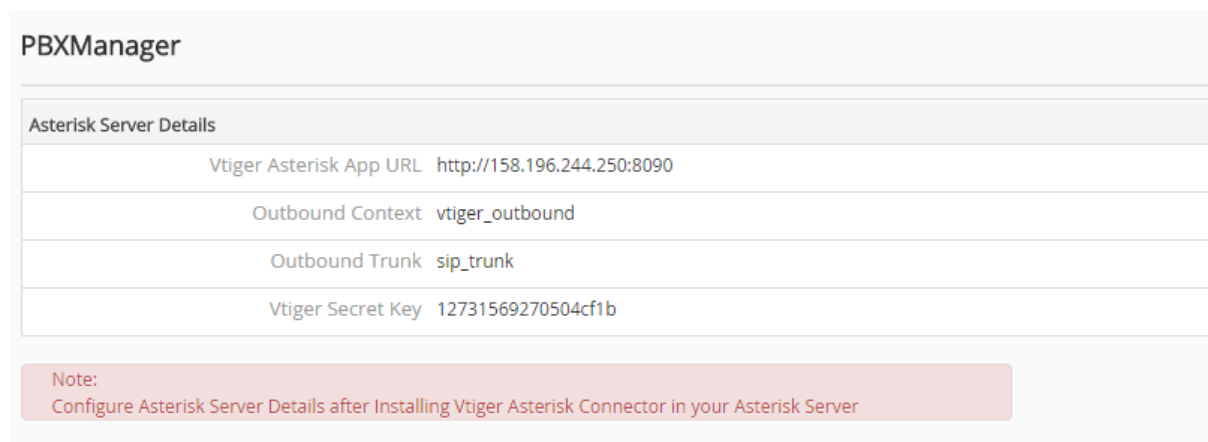
Vtiger CRM nabízí tři placené cloudové verze s 24 hodinovou podporou – Sales, Support a Ultimate edition. V těchto placených verzích, kde se nemusíte starat o stranu serveru, je integrace s PBX řešena pomocí rozšíření Phone Calls. Tento model je zdarma a v konfiguraci poskytovatele je možné si zvolit jako bránu jedno z následujících řešení: Asterisk, AsteriskDirect, Twilio, Vicidial, Plivo či Exotel. Při výběru Asterisku se zobrazí nastavení zobrazené na obrázku 1.6, kde se nastavuje nejruznější informace potřebné k propojení Asterisku se CRM systémem jako např. dialplán kontext pro řízení hovoru, trunk či Vtiger Asterisk App URL. Tato URL adresa je adresa webové aplikace, která slouží spojovací prvek pro komunikaci mezi Vtiger CRM a Asteriskem. Defaultně je od Vtiger CRM poskytována webová aplikace s názvem Vtiger Asterisk Connector. Aktuální komerční verze Vtigeru k únoru 2016 je verze 7.16.1.[8]

Gateway Configuration	
*Gateway	Asterisk
*Gateway Status	Active
*Default Gateway	No
*Vtiger Asterisk App URL	<input type="text"/>
*Outbound Context	<input type="text"/>
*Outbound Trunk	<input type="text"/>
*Outbound Prefix	SIP
*Your Caller ID (For Outbound Calls)	<input type="text"/>
*Ticket / Case Title	<input type="text"/>
*Opportunity Name	<input type="text"/>
*Callback URL	https://schoolproject1.od2.vtiger.com Copy
*Vtiger Secret Key	63432936156b5cf006125f Copy

Obrázek 1.6: Konfigurace Asterisku ve Vtiger Phone Calls modulu

Open-source řešení již nenabízí Phone Calls rozšíření, ale za to je již defaultně přidán PBXManager modul. Ten obdobně jako Phone Calls modul pomáhá s integrací PBX s Vtiger CRM.

PBXManager modul ovšem nabízí pouze konfiguraci telefonní ústředny založené na PBX Asterisk. Omezené nastavení poskytovatele můžeme vidět na obrázku 1.7.



Obrázek 1.7: Konfigurace Asterisku v PBXManager modulu

1.3.1.1 Vtiger Asterisk Connector

Vtiger Asterisk aplikace se chová jako gateway pro připojení k Vtiger CRM z Asterisk serveru. Uspodňuje tak interakce přes HTTP mezi Vtigerem a Asteriskem při zpracovávání příchozích či odchozích hovorů. Mezi jeho hlavní tři funkce patří připojení k Vtigeru a notifikace příchozího hovoru, přijímání příkazů provedených ve Vtiger CRM a jejich předání Asterisku (Click to Call funkce) a správa nahrávání hovorů. Vtiger Asterisk Connector se nasazuje na server s Asteriskem a je napsán v jazyce Java. Tato webová aplikace se používá u Phone Calls modulu v placených verzích, tak i v PBX Manager modulu v open-source verzi. [9]

Vtiger CRM open-source řešení jsem si zvolil pro mé praktické řešení a to s nejaktuálnější verzí k únoru 2016 verzi 6.4.

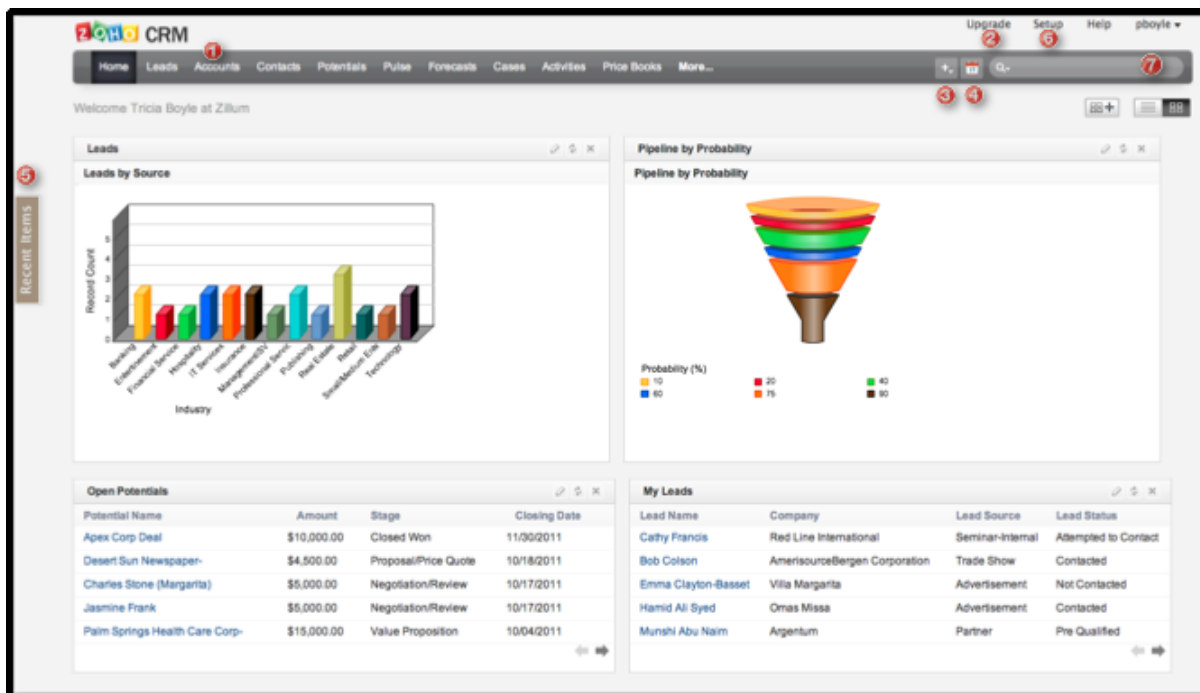
1.3.2 Zoho CRM

Další alternativa v open-source řešeních CRM systému je Zoho CRM od společnosti Zoho, která poskytuje mnoho SaaS business aplikací. Zoho CRM poskytuje API nezávislé na programovacím jazyku, tudíž je možné vyvíjet aplikace v jakémkoliv programovacím jazyce. CRM API umožňuje extrahovat CRM data v XML nebo JSON formátu a pracovat až se sedmnácti moduly. K vyzkoušení je k dispozici Free verze s omezením pro 10 uživatelů a 1000 API požadavků. Pro integraci Asterisku se Zoho CRM, obsahuje tento CRM systém speciální aplikaci pro podporu telefonie s názvem Zoho PhoneBridge.[10]

1.3.2.1 Zoho PhoneBridge

Zoho PhoneBridge spojuje vybranou PBX se Zoho CRM systémem. Zoho PhoneBridge nabízí podobné vlastnosti jako Vtiger CRM a to správu hovorů, vytočení kontaktu pomocí kliknutí, informační vyskakovací okno s mnoha funkcemi při příchozím hovoru. Podporované PBX systémy jsou Avaya, Asterisk a Elastix a také hostované PBX jako Twilio, Ringio, RingCentral, Ozonetel a Promero. Pro integraci Asterisku je nutné stáhnout webovou aplikaci Asterisk Adapter. Asterisk Adapter funguje na podobném principu jako Asterisk Connector u Vtiger CRM. Je zde zapotřebí extrahovat zip soubor na server Asterisku a poté konfigurovat soubory zti.properties, kde je třeba

nastavit PhoneBridge token vygenerovaný v sekci Developer Space v samotném CRM a soubor asterisk14.properties. Zoho PhoneBridge bohužel podporuje pouze Asterisk ve verzích 1.4 - 1.8 s tím, že vývojáři poukazují na nestabilní funkčnost této aplikace s verzí Asterisku 11.[11]



Obrázek 1.8: Ukázka domovské stránky Zoho CRM[10]

1.3.3 SuiteCRM

SuiteCRM systém je další větev oblíbeného CRM systému SugarCRM. Vznikl v roce 2013 a představuje open-source větev SugarCRM Community Edition. Architektura je obdobná jako u Vtiger CRM s výhodou podpory pro Microsoft SQL Server. Obsahuje 22 modulů jako např. Workflow Automation, Mobile responsive, Outlook plugin a další. SuiteCRM podporuje integraci Asterisku a dalších SW PBX jako Elastix, FreePBX, Vicidial, Asterisknow atd., pro příchozí i odchozí hovory. Pro integraci Asterisku nabízí SuiteCRM aplikaci All-In-One CTI, která podporuje PBX systémy jako Asterisk, Avaya, Panasonic a další. Ovšem pro bezplatnou volbu je k dispozici aplikace yaai. [12]

1.3.3.1 YAAI

YAAI neboli Callinize je konektor, který umožňuje integraci Asterisku s SugarCRM a některých odvětví jako SuiteCRM. YAAI je napsaný v php jazyce a volně dostupný na Githubu. Callinize nabízí vlastnosti jako logování hovorů, notifikační okno, Click to Dial funkci, synchronizování oken prohlížeče apod. YAAI je výhradně určen pro Asterisk a to pro verze 1.6 a vyšší, kde verze 1.4 je částečně podporovaná a doporučena je verze 1.8.[13]

The screenshot displays the SuiteCRM interface for a contact named Allyson Savage. The top navigation bar includes 'SuiteCRM' and various modules: Contacts, Sales, Marketing, Support, Activities, Collaboration, and All. A search bar and a user profile for 'Administrator' are on the right. The left sidebar contains 'Actions' (Create Contact, Create Contact From vCard, View Contacts, Import Contacts) and 'Recently Viewed' (Allyson Savage, Administrator, test, Create new p..., Communicate..., Refugio Eric..., Draft Divers...). The main content area shows the contact profile for Allyson Savage, including fields for First Name, Last Name, Photo, Office Phone, Mobile, Title, Email Address, Account Name, and Primary Address City. Below the profile is an 'Activities' section with a table of tasks:

Subject	Status	Contact	Due Date	Assigned User
Discuss New Launch	Planned	Allyson Savage	26-03-2015 06:45	Administrator
Prepare launch document	Not Started	Allyson Savage	25-03-2015	Administrator

Obrázek 1.9: Ukázka domovské stránky SuiteCRM[14]

1.3.4 Výhody a nevýhody

Open-source řešení sebou nese spoustu výhod, ale i nevýhod. Proto je dobré veškeré výhody a nevýhody znát před samotnou implementací.

Výhody:

- Cena – open-source je ke stažení zdarma, tudíž za samotný software není třeba vůbec platit. Pokud ovšem chceme využít jistých služeb, je třeba platit, i když zdaleka menší částku než u komerčních řešení.
- Otevřený zdrojový kód – interní IT oddělení v podniku může upravovat a přizpůsobovat produkt k vlastním potřebám bez zásahu poskytovatele.

Nevýhody:

- Limitovaná podpora – u většiny open-source verzí daných projektů chybí uživatelská podpora, popřípadě je poskytnuta za poplatek.
- Méně funkcionalit – menší nabídka funkcionalit oproti placenných verzí [15]

2 Asterisk

Asterisk je open source framework pro vytváření aplikací pro komunikaci a hlavně softwarová implementace telefonní pobočkové ústředny PBX (Private Branch eXchange) umožňující IP telefonii, analogovou telefonii i ISDN. Asterisk je software, který dokáže změnit obyčejný počítač na komunikační server. Asterisk je oficiálně definován jako open source hybrid TDM a packet voice PBX. Asterisk „běží“ na platformách Linux a Unix a díky tomu, že je open source, se stal jedním s nejpoužívanějším telekomunikačním softwarem ve VoIP oblasti. Využívají ho malé a velké podniky, call centra, dopravci a vládní agentury po celém světě. Vzhledem k tomu, že Asterisk dominuje jako nejpoužívanější softwarová pobočková ústředna, vývojáři jiných softwarových produktů, kde je i možná práce s PBX, často vytvářejí moduly či jiné řešení pro případnou integraci a konektivitu Asterisku s jejich řešením. [16]

Architektura Asterisku je velmi odlišná od jiných, tradičnějších telefonních ústředěn a to v tom, že dialplán zachází se všemi příchozími kanály ("channels") v podstatě stejným způsobem. Asterisk je jakoby středový prvek spojující telefonní technologie a aplikace. V Asterisku je prakticky vše, co přichází nebo vychází ze systému, zařazeno v určitém kanálu, který pak zpracovává dialplán. Například Asterisk nerozlišuje rozdíly mezi stanicemi a trunky jako tradiční PBX.

Asterisk je postaven na modulech. Spolu se samotnou aplikací se spouští moduly Asterisku, které zajišťují nejrůznější funkcionalitu (např. kanálové ovladače), zdroje k umožnění propojení s externí technologií (např. ODBC), informace a také kodeky. Moduly jsou načteny a konfigurovány v souboru `/etc/asterisk/modules.conf` [17]

2.1 Dialplán

Dialplán je jedna z nejdůležitějších částí Asterisku vůbec. Určuje, jak má Asterisk zpracovávat příchozí a odchozí hovory, a řídí jejich směrování. Dialplán je konfigurován v souboru `/etc/asterisk/extensions.conf`, kde jsou formou skriptovacího jazyku popsány pokyny. Je plně přízpusobitelný a nezávislý na jakékoli technologii.

Dialplán je tvořen ze sekcí `general`, `globals` a dále vytvořených kontextů a popřípadě i makra. Sekce `general` obsahuje základní nastavení např. ohledně ukládání dialplánu viz níže.

```
[general]
static=yes
writeprotect=yes
```

Sekce `globals` obsahuje globální proměnné používané jako konstanty s nastavenými hodnotami. Např. nastavení doby vyzvánění a jaký zvukový soubor bude přehrán ve voicemailu, viz níže.[17]

```
[globals]
RINGTIME => 3
MANNOUNCE => mysounds/my-vm-announce
```

V dialplánu pracujeme také s dalšími proměnnými např. aplikační, kanálové atd.

Dialplán se skládá ze souboru několika kontextů neboli context. Kontexty dále obsahují jednotlivé extensions , které jsou nejdůležitějšími částmi dialplánu. Extensions vyjadřují co a jakými funkcemi má Asterisk provést. Tyto kontexty dále pak vykonávají funkce podle priorit, které nám určují, v jakém pořadí se budou vykonávat. Mimo funkce mohou být extensions přesměrovány na jiný příkaz či jiný kontext. Každý příkaz je uveden na samostatném řádku v následujícím formátu.

```
exten => extension,priority,Command(parameters)
```

Např. kontext hello s extensions 123 zvedne hovor, přehraje zprávu s názvem „hello“ a zavěsí, viz níže.

```
[hello]
```

```
exten => 123,1,Answer
```

```
exten => 123,2,Playback(hello)
```

```
exten => 123,3,Hangup
```

Extensions mohou být také implementovány s různými funkcemi. Například Security - povolení mezinárodních volání pouze z určitých telefonů. Routing - směrování hovorů na základě extension. Autoattendant - uvítání volajících a vyzvání k volbě extension. Multilevel menus - menu pro prodej, podporu atd. Authentication - dotaz na přístupové heslo pro určité extension. Callback - redukce mezinárodních poplatků. Privacy - seznam nežádoucích volajících čísel na dané extension. PBX Multihosting - lze vytvářet „virtuální hosty“ na PBX. Daytime/Nighttime - lze měnit chování na základě hodin. Macros - vytváření skriptů pro běžně používané funkce. Dialplán tak ovládá veškeré kanály přicházející do systému. [17]

2.2 SIP

SIP (Session Initiation Protocol) je signalizační protokol, který sestavuje, dohlíží a ukončuje spojení mezi účastníky hovoru. SIP pracuje na aplikační vrstvě a může být přenášen pomocí TCP i UDP protokolu. Standardně komunikuje na portu 5060. SIP je textově orientovaný protokol, který komunikuje pomocí žádostí a odpovědí. SIP protokol má podobnou strukturu jako protokoly HTTP nebo SMTP. I přes občasné problémy s NATem je nejvíce využívaným protokolem v IP telefonii. Kromě SIP protokolu se také používá interní IAX2 protokol uveden Asteriskem.[16]

SIP je end-to-end orientovaný protokol. To znamená, že koncová zařízení mají uloženou veškerou logiku a znají i jednotlivé stavy komunikace. SIP entita je vázána k doméně či hostiteli SIP proxy. Jako identita SIP entit se používá URI. Zjednodušený tvar SIP URI vypadá následovně: sip:user@host. Signalizace SIP spolupracuje s dalším protokolem k popisu relací, médií a časování. Ten určuje použitá kódování pro média, čísla portů atd. Nejčastěji se používá protokol SDP. K přenosu multimédií používá SIP protokol RTP. [16]

Vzhledem k obrovské popularitě SIP protokolu, je SIP kanálový modul nejvyspělejší a obsahuje nejvíce funkčních možností ze všech kanálových modulů v Asterisku. Konfigurace SIPu se provádí v souboru */etc/asterisk/sip.conf*. Konfigurační soubor obsahuje sekci general, kde se nastavují základní parametry, a dále sekce jednotlivých uživatelů či šablon pro více uživatelů se stejnými parametry. Následující ukázka níže popisuje konfiguraci šablony "firma" a dvou uživatelů využívající

tuto šablonu pro svou konfiguraci s odlišným heslem. V konfiguraci šablony "firma" jsou zahrnuty atributy jako povolené kodeky, kontext dialplánu či typ, který určuje, jak Asterisk interpretuje příchozí SIP požadavky. Atribut type má tři možnosti - peer, user a friend, přičemž friend je kombinací dvou předešlých. Peer se registruje k ústředně pomocí IP adresy a portu. User se neregistruje, pouze autorizuje na začátku každého hovoru pomocí uživatelského jména. [17]

```
[firma] (!)
type=friend
context=from-internal
host=dynamic
disallow=all
allow=alaw
```

```
[vratnice] (firma)
secret=12345
```

```
[kancelar] (firma)
secret=heslo
```

2.3 Rozhraní AMI

Asterisk Manager Interface, zkráceně AMI, je monitorovací systém a rozhraní pro správu Asterisku. Je to rozhraní, které umožňuje real-time monitoring a posílání žádostí o provedení nejrůznějších akcí Asterisku. Lze tak vzdáleně ovládat Asterisk konzoli pomocí příkazů prostřednictvím TCP/IP socketu. AMI komunikuje jednoduchým protokolem v podobě "Key: value", kde Key je typ paketu. Existují tři typy paketů a oznamují nám typ požadavku:

Action - akce, požadavek ze strany klienta na ústřednu

Response - odpověď ústředny na požadavek klienta

Event - událost, údaje týkající se události generované z Asterisk jádra nebo rozšiřujícím modulem

Existuje několik způsobů, jak se připojit k rozhraní AMI. Mezi nejpoužívanější patří připojení pomocí TCP socketu (např. přes Telnet) či HTTP protokolu. Zatímco u TCP nabízí AMI rozhraní pouze jeden typ struktury zpráv, u HTTP je možnost několika kódování, např. ve formátu XML či v klasickém HTML. Typ kódování se volí v poli v URL požadavku.

Konfigurace rozhraní AMI lze nastavit v souboru *etc/asterisk/manager.conf*, kde se kromě sekce general konfiguruje jednotliví uživatelé a jejich práva pro přístup k AMI rozhraní. Sekce general konfiguruje hlavní nastavení AMI rozhraní jako adresu, port, TLS nastavení apod. Prvek "enabled", který vůbec umožní chod AMI, je defaultně nastavený na "no". Kromě enabled je v sekci general ještě prvek webenabled, který umožňuje chod AMI přes HTTP protokol. Pro zprovoznění AMI přes HTTP

protokol musíme i nakonfigurovat soubor `/etc/asterisk/http.conf`, který umožní nastavení a spuštění jednoduchého Asterisk HTTP serveru. Přístup z webové aplikace či prohlížeče je umožněn přes AJAM technologii přístupnou od verze Asterisku 1.4.[18]

Akce ze strany uživatele umožňují vyvolat příkazy v Asterisk CLI. Těchto příkazů je veliké množství a je možné je zobrazit příkazem v Asterisk příkazovém řádku pomocí příkazu níže:

```
manager show commands
```

Asterisk Manager Interface je hojně využíváno při vývoji různých aplikací a systému programovaných v různých jazycích, a proto již existují nejrůznější frameworky, které usnadňují práci s AMI rozhraním. Tabulka níže představuje úspěšně používané knihovny.

Tabulka 2.1: *AMI frameworky[19]*

Název	Jazyk	Rozhraní
Asterisk-Java	Java	AMI/FastAGI
StarPy	Python	AMI/FastAGI
Adhearsion	Ruby	AMI/FastAGI
Node-asterisk	JavaScript	AMI
Pyst2	Python	AMI/AGI
PAMI	PHP	AMI
Panoramisk	Python	AMI/FastAGI

2.4 Rozhraní AGI

Asterisk Gateway Interface, zkráceně AGI, je rozhraní Asterisku, jenž umožňuje řízení příchozích a odchozích hovorů a dalších přidávaných funkcionalit do Asterisku v různých programovacích či skriptovacích jazycích. Většina programátorů totiž preferuje svůj oblíbený jazyk před skriptováním v dialplánu. AGI poskytuje rozhraní mezi dialplánem a externí aplikací. Tím usnadňuje prostředí pro snazší integraci s externími aplikacemi či systémy. Rozhraní AGI v určitých případech spolupracuje s rozhraním AMI.

2.4.1 AGI varianty

AGI variant pro komunikaci s Asteriskem je hned několik. Je dobré znát všechny pro vhodný výběr na základě potřeb dané aplikace.

Process-Based AGI

Process-Based je nejjednodušší AGI varianta. AGI se zavolá AGI() aplikaci v dialplánu obsahující příkaz, respektive název agi skriptu, který se má vykonat a další případné argumenty.

```
exten => 101,1,AGI(hello.sh)
```

Výše vidíme ukázkou použití AGI v dialplánu, kde se zavolá skript "hello.sh". Defaultní cesta, kde AGI hledá soubory, je `/var/lib/asterisk/agi-bin`. Jakmile Asterisk provede AGI aplikaci, komunikace mezi Asteriskem a danou aplikací bude probíhat v standartních proudech `stdin` a `stdout`.

EAGI

EAGI, neboli Enhanced AGI, je téměř totožné jako základní Process-Based AGI. Jediný rozdíl je v komunikaci datových proudů a to ten, že EAGI zvládá aplikace s audio streamem. V dialplánu se místo AGI() volá EAGI() aplikace.[17]

FastAGI

FastAGI je AGI přes TCP spojení. U základní Process-Based AGI se spouští instance AGI aplikace v systému pro každý hovor, kde komunikuje v standartních proudech. Zatím co u FastAGI se vytvoří spojení přes TCP na straně FastAGI serveru. FastAGI využívá stejný AGI protokol, ovšem komunikace probíhá přes TCP a není potřeba vytvářet nový proces při dalším novém hovoru. V dialplánu se obdobně volá AGI() aplikace, kde je namísto názvu aplikace IP adresa, popřípadě s portem (defaultní port je 4573) FastAGI serveru, kde se nachází daná aplikace, viz ukázka níže:

```
exten => 101,1,AGI(agi://127.0.0.1:4574)
```

FastAGI také podporuje záznamy SVR records, jestliže je potřeba použít URL jako hostname. Asterisk tak projde záznamy v DNS pro nalezení hostname zapsané v dialplánu ve formě hagi zápisu, viz ukázka níže.

```
exten => 101,1,AGI(hagi://domena.cz)
```

Async AGI

Účelem Async AGI je povolit aplikaci, která používá AMI rozhraní, asynchronně zařadit AGI příkazy, které mají být provedeny na určitém kanálu. To znamená, že existující AMI aplikace může použít AGI příkazy pro řízení hovorů. U daného uživatele musí být povolen parametr `agi` v konfiguračním souboru `/etc/asterisk/manager.conf` a následně konfigurace dialplánu vypadající ve formě vypsane níže:

```
exten => 101,1,AGI(agi:async)
```

Po natavení AGI relace Asterisku a AGI aplikací, které probíhají trochu odlišně v závislosti na variantě AGI, Asterisk zahájí zpracování hovorů v pořadí příchozích AGI příkazů. AGI je synchronní, tudíž Asterisk zpracovává vždy jeden aktuální příkaz. List AGI příkazů je možné si zobrazit v Asterisk CLI díky příkazu níže: [17]

```
agi show commands
```

2.5 Rozhraní ARI

Dalším Asterisk rozhraním je rozhraní ARI, neboli Asterisk REST Interface. Toto rozhraní je nejnovější a přišlo až s Asteriskem 12. Důvodem byl příchod popularity konceptů jako SOAP, XML/JSON-RPC či REST a také zjednodušení implementace Asterisk komunikační aplikace, která šla napsat pouze jako modul v jazyce C nebo použitím kombinací AMI a AGI rozhraní. ARI je asynchronní API, které umožňuje vývojářům vytvořit komunikační aplikaci tím, že vystaví nezpracované primitivní objekty Asterisku rozhraní REST API. Stav objektů jsou pak kontrolovány uživatelem a přenášeny v podobě JSON objektů přes WebSocket. ARI není o spuštění VoiceMail služby v dialplánu na určitém kanálu či přesměrování kanálu v dialplánu na VoiceMail službu, ale

o možnosti implementovat svou vlastní VoiceMail službu. ARI se skládá ze tří základních částí, které jsou propojené a používané společně:

- RESTful rozhraní, které klient používá k řízení zdrojů v Asterisku
- WebSocket, který zprostředkovává události v JSON formátu o zdrojích Asterisku ke klientovi
- Statis() dialplán aplikaci, která předává kontrolu daného kanálu Asterisku klientovi[20]

Konfigurace Asterisku se týká jednak souboru `/etc/asterisk/http.conf`, kde je třeba povolit Asterisk HTTP server v sekci `general`, dále v souboru `/etc/asterisk/ari.conf`, kde je potřeba povolení této služby plus přidání uživatele a v poslední řadě je třeba vyvolání Statis() aplikace v dialplánu, viz ukázka níže: [20]

```
exten => 101,1,Statis(nazevAplikace)
```

2.6 Integrace Asterisku s relačními databázemi

Integrace Asterisku s podnikovými sítěmi úzce souvisí s integrací s relačními databázemi, neboť většina podnikových informačních systémů je založena právě na stavebním kameni zvaném relační databáze. Veškeré informace uživatelů systémů, zaměstnanců či zákazníků jsou právě uloženy v databázi informačního systému a Asterisk s těmito dynamicky se měnícími informacemi může pracovat a využívat je ve svém dialplánu a dalších informativních úlohách. Asterisk lze integrovat hned s několika databázemi jako MySQL, PostgreSQL či dokonce Microsoft SQL. Pro spojení Asterisku s externí databází slouží např. ODBC. Open Database Connectivity umožní přístup k datům z libovolné aplikace bez ohledu na to, který systém pro správu databáze (DBMS) je použit. [17]

2.6.1 ODBC a dialplán funkce `func_odbc`

ODBC connector je databázová abstraktní aplikační vrstva, která umožňuje komunikaci Asterisku s širokou řadou databázových systémů. Jakmile je ODBC nainstalovaný v operačním systému, je potřeba nakonfigurovat dvojici souborů pro daný databázový systém. V souboru `/etc/odbcinst.ini` se konfiguruje ovladač pro daný databázový systém. Poté v souboru `/etc/odbc.ini` se konfiguruje identifikátor s danou databází a serverem pro Asterisk. Dále je pak nutná konfigurace na straně Asterisku. Zde je třeba nakonfigurovat (a povolit samotnou aplikaci) v souboru `/etc/res_odbc.conf`, kde se nastavuje i mimo jiné parametr `dsn`, který odkazuje na zmíněný identifikátor ze souboru `odbc.ini`. Na závěr zbývá už jen soubor `/etc/func_odbc.conf`, kde kromě parametru `dns`, který odkazuje na soubor `res_odbc.conf`, se už píše samotné SQL dotazy, které se potom dále využívají při tvorbě dialplánu. V dialplánu se volá funkce s prefixem ODBC a s případnými argumenty. Dotazy SQL mohou být dvojího typu a to buď čtecího, v podobě `readsql`, či zapisovacího `writesql`. Vztahy mezi těmito konfiguračními soubory můžeme vidět na obrázku 2.1 [17]

Níže je znázorněná ukázka použití `func_odbc` v dialplánu. V souboru `func_odbc.conf` máme napsaný dotaz na vybrání daného čísla ze sloupce `cislo` z tabulky `kontakty`, kde ID kontaktu je zadáno přímým argumentem. Nejprve ukázka z dialplánu:

```
exten => dejCislo,1,Set(CISLO=${ODBC_CISLO(1)})
```

V ukázce z dialplánu můžeme vidět použitou funkci `ODBC_CISLO()` s argumentem 1. Tato funkce se konfiguruje v souboru `func_odbc.conf` následovně:

```
[CISLO]
```

```
dsn=asterisk
```

```
readsql= SELECT cislo FROM kontakty WHERE id='${ARG1}'
```

Nakonfigurovaných funkcí s různými dotazy může být v souboru `func_odbc` nespočet. Stejně tak argumentů a hodnot může být i popřípadě více dle potřeby a obtížnosti dotazu. Je možné výběrem vybrat i více řádků, které postupně zpracovává funkce `ODBC_FETCH`. SQL dotazy se dají vnořit i přímo do samotného dialplánu. V souboru `func_odbc.conf` stačí tedy pouze nakonfigurovat jednu funkci, např. takto: [17]

```
[SQL]
```

```
dsn=asterisk
```

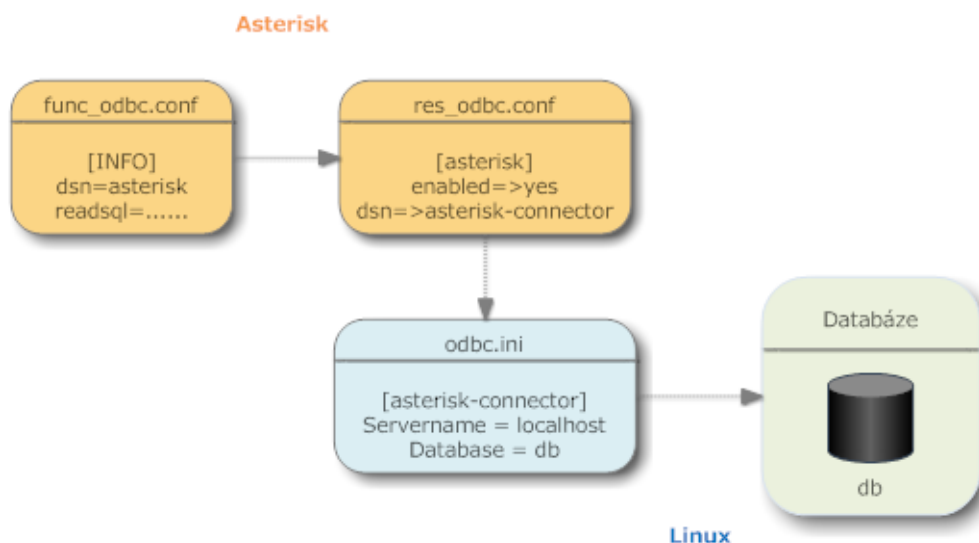
```
readsql=${SQL_ESC(${ARG1})}
```

```
writesql=${SQL_ESC(${VAL1})}
```

S pomocí výše konfigurované univerzální funkce SQL můžeme psát jakékoliv dotazy v přímo v dialplánu. Ukázka výběru čísla z kontaktů by vypadala následovně:

```
exten => dejCislo,1,Set(CISLO=${ODBC_SQL(SELECT cislo FROM kontakty WHERE id=1)})
```

Funkce `func_odbc` je tedy velmi silný a užitečný nástroj pro integraci relačních databází s Asteriskem. Proto jsem ho také využil ve své práci.



Obrázek 2.1: Vztah mezi konfiguračními soubory ODBC pro Asterisk[17]

2.7 Integrace Asterisku se CRM

Asterisk lze integrovat s celou řadou různých softwarových řešení pro zlepšení a usnadnění běžným uživatelům. Jedním z častých technologických řešení je integrace Asterisku se CRM systémy. Integrace Asterisku se CRM poskytuje nové funkce a možnosti jako nahrávání hovorů se zákazníky, řízení volajících zákazníků, hromáždění dat a analýzy proběhlých hovorů, vyhledávání čísel a řadu dalších výhod, které samostatný Asterisk nebo nezávislé CRM nemůže poskytnout. Vývojáři

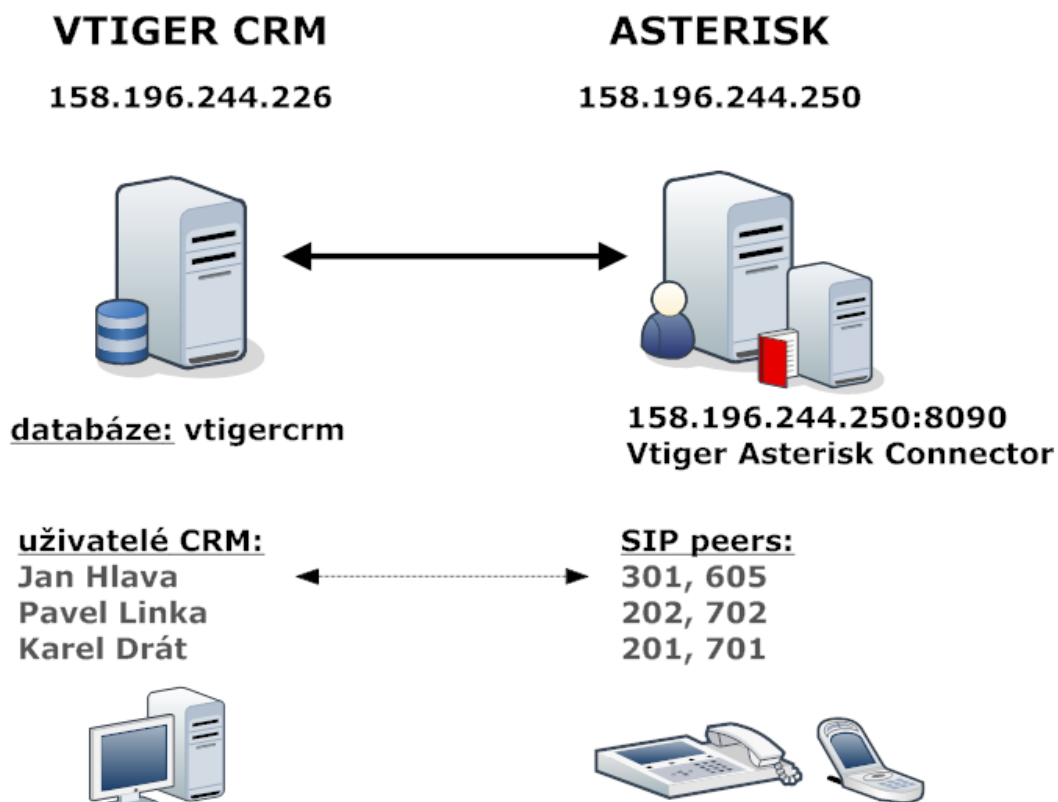
nejrůznějších CRM systémů berou integraci s PBX jako standartní prvek, a proto je u některých open-source produktů k dostání i dokumentace k integraci s určitými softwarovými PBX. Ve většině případů pomáhá k integraci externí aplikace, která jednak komunikuje se CRM systémem a Asteriskem. V případě komunikace s Asteriskem se zde nejčastěji využívá kombinace rozhraní AMI a AGI, popřípadě rozhraní ARI. Pro podniky s vybudovaným call centrem postaveném na Asterisku je tato integrace klíčová.

3 Integrace Asterisku s Vtiger CRM

Právě zmíněný Vtiger CRM systém a jeho integrace s PBX Asterisk je hlavní náplní této práce. Přesněji instalace a konfigurace Asterisku a Vtiger CRM na dva odlišné linuxové servery ve stejné síti, instalace a konfigurace Vtiger Asterisk Connector a ODBC, návrh a implementace dialplánu a modifikace notifikačního okna příchozích hovorů ve Vtiger CRM. Má experimentální topologie byla implementována v zabezpečené školní síti, tudíž jsem nemusel brát zřetel na bezpečnostní opatření. Testování hovorů probíhalo výhradně mezi SIP klienty.

3.1 Experimentální topologie

Mnou zvolená experimentální topologie se skládala ze dvou virtuálních vzdálených serverů s operačním systémem linux, přesněji distribucí Debian GNU/Linux 8.2 (jessie). Oba servery byly ve školní síti a disponovaly statickou IP adresou 158.196.244.226 a 158.196.244.250. Připojení k těmto serverům bylo možné pouze ze školní sítě, kam se uživatel mohl připojit popřípadě přes VPN. Bezpečnost tedy byla z velké míry zabezpečena. Na Asterisk serveru jsem implementoval i Vtiger Asterisk Connector, který operoval na portu 8090. Pro mnou zvolený experimentální návrh jsem vytvořil tři uživatele Vtiger CRM a ke každému přidělil dva SIP účty, kde první SIP účet představoval SIP extension softphone v pracovním počítači a druhý imitoval číslo mobilní telefonu. Vtiger CRM systém byl hostován na webovém serveru Apache a společně s Vtigerem byla vytvořena databáze vtigercrm v databázovém serveru MySQL. Topologii společně s danými uživateli a extensions můžete vidět na obrázku 3.1.



Obrázek 3.1: Experimentální topologie

3.2 Implementace Vtiger CRM systému

Vtiger CRM má k dispozici i open-source verzi volně ke stažení s aktuální verzí 6.4, kterou jsem využil pro svou práci. Vtiger CRM systém je napsán v jazyce PHP a "běží" na webovém serveru Apache s MySQL databází. Ještě před samotným stažením a instalací open-source verze jsem si tedy připravil virtuální server a nainstaloval důležité předpokládané požadavky jako PHP, Apache a MySQL server. Tyto požadavky jsou nutné ve verzích zmíněných níže:

- Apache 2.0.40 a výš
- PHP 5.2.x a výš, doporučeno spíše PHP 5.5.x
- MySQL verze 5.1.x

Postupně jsem zmíněné komponenty nainstaloval přes repositář Debianu pomocí příkazů níže:

```
# apt-get install apache2
# apt-get install php5 libapache2-mod-php5
# apt-get install mysql-server php5-mysql
```

Dále jsem si z oficiální webové stránky www.vtiger.com/download zkopíroval odkaz pro stáhnutí open-source verze Vtiger CRM 6.4.0 přes web sourceforge.net a příkazem `wget` stáhnul. Bylo nutné přemístit rozbalenou složku z tar souboru do adresáře `/var/www/` a udělit celému adresáři oprávnění pomocí `chmod` příkazu. Apache defaultně zobrazuje svou domovskou stránku, a proto bylo zapotřebí nastavení virtuálního hosta pro zobrazení Vtiger CRM systému jako defaultní stránku na webovém serveru. Vytvořil jsem tedy soubor `vtigercrm.conf` v adresáři `/etc/apache2/sites-available/` a nastavil následující parametry:

```
<VirtualHost *:80>
    ServerAdmin admin@localhost

    ServerName vtigercrm

    DocumentRoot /var/www/vtigercrm

    ErrorLog ${APACHE_LOG_DIR}/error.log

    CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

Deaktivace defaultní stránky a aktivace nové se provádí příkazy `a2dissite` a `a2ensite`, v mém případě tedy:

```
# a2dissite 000-deafult.conf
# a2ensite vtigercrm
```

Po restartu Apache web serveru se mi po zadání IP adresy ve webovém prohlížeči úspěšně zobrazil instalační průvodce Vtiger CRM systému. Po odkliknutí `Install` tlačítka a odsouhlasení s licenčními podmínky mě trochu zmátla následující stránka s požadavky, kde instalační průvodce nemohl detekovat mou verzi PHP. Tuto chybnou výstrahu lze ignorovat a pokračovat v instalaci či lze opravit chybu v souboru `modules/install/models/utills.php`, kde na řádce 129 je zadaný špatný operátor. Na další stránce následovala systémová konfigurace, kde kromě nastavení a vytvoření uživatele je

blok ohledně databáze. Ten obsahuje prvky jako Host Name serveru, User Name uživatele pro přístup k databázi a jméno databáze. Databázi si lze vytvořit už před instalací pomocí SQL příkazu či v phpmyadmin pokud je nainstalovaný v operačním systému anebo stačí pouze zaškrtnout možnost vytvoření databáze v instalačním průvodci a doplnit informace ohledně Root uživatele. Dále už následuje jen shrnutí informací a výběr modulů nainstalovaných do CRM systému a instalace Vtiger CRM je kompletní.

The screenshot shows the 'System Configuration' window for Vtiger CRM. It is divided into three main sections:

- Database Information:** Database Type is MySQL. Host Name is localhost. User Name is root. Password is masked with asterisks. Database Name is vtiger6_beta_doc. The 'Create new database' checkbox is checked. Root User Name is root. Root Password is masked with asterisks.
- System Information:** Currency is set to USA, Dollars (\$).
- Admin User Information:** User Name is admin. Password and Retype Password are masked with asterisks. First Name is Rehman. Last Name is Shareef. Email is shareef@vtiger.com. Date format is mm-dd-yyyy. Time Zone is (UTC+05:30) Chennai, Kolkata, Mumbai, New Delhi.

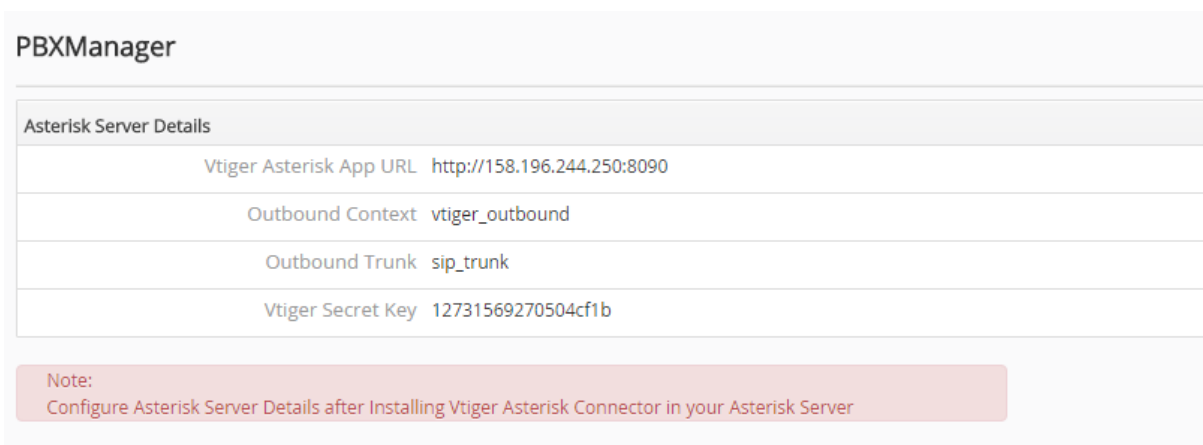
At the bottom right, there are 'Back' and 'Next' buttons.

Obrázek 3.2: Systémová konfigurace při instalaci Vtiger CRM[20]

Po přihlášení do CRM systému jsem si vytvořil tři uživatele v sekci Users, které jsem využil pro svůj praktický návrh smyšleného podniku. Jediné důležité informace při vytváření těchto uživatelů byly CRM Phone Extension, Mobile, Role a samozřejmě User Name a jméno. CRM Phone Extension a Mobile byly telefonní kontakty na softphone v jejich pracovních počítačích či na mobilní telefon při nepřítomnosti v práci. Role byly vybrány dvě - Sales Person pro dva operátory a Sales Manager pro jednoho managera.

Pro integraci Vtiger CRM s Asteriskem je důležité nastavení PBXManager modulu v CRM systému. To se nachází v CRM settings > Integration. Nastavení PBXManagera obsahuje čtyři informace a to:

- Vtiger Asterisk App URL - URL Vtiger Asterisk Connectoru ve formátu (protokol):(ip_asterisk_serveru):(VtigerConnector_port)
- Outbound Context - kontext v dialplánu
- Outbound Trunk - trunk spojení
- Vtiger Secret Key - bezpečnostní klíč generovaný Vtiger CRM



Obrázek 3.3: Konfigurace Asterisk serveru ve Vtiger CRM

Jelikož má testovací topologie působila v jedné síti o jedné PBX, nebylo zapotřebí řešit trunk spojení, tudíž ani konfigurovat Outbound Trunk. Toto pole je ovšem povinné, a proto je zapotřebí vepsat textový řetězec.

3.3 Konfigurace Asterisk serveru

Softwarovou telefonní ústřednu Asterisk jsem nainstaloval na druhý virtuální server. Asterisk má několik verzí a tou nejaktuálnější je verze 13, ovšem pro svou práci jsem si vybral verzi 11, která je stále jedna z nejpoužívanějších verzí Asterisku. Asterisk lze také nainstalovat přes repositář pomocí příkazu:

```
# apt-get install asterisk
```

Asterisk v mém případě vyžadoval konfiguraci pěti souborů. Veškeré tyto soubory jsou konfiguračního typu a nachází se v adresáři `/etc/asterisk/`. Tato pětice zahrnovala soubory `sip.conf`, `manager.conf`, `extensions.conf`, `res_odbc.conf` a `func_odbc.conf` sloužící ke správě SIP účtů, AMI rozhraní, dialplánu hovorů a ODBC propojení s databází.

První soubor `sip.conf` spravuje veškeré SIP připojení přes SIP kanál k Asterisku. V tomto souboru jsem konfiguroval SIP účty jednotlivých uživatelů ze CRM systému. Jak jsem již zmiňoval výše, veškeré mé testovací hovory proběhly mezi SIP telefony, což znamená, že i mobilní telefon CRM uživatele byl realizován pomocí SIP účtu. Má sekce `general` souboru `sip.conf` obsahuje pár parametrů jako adresu, defaultní kontext dialplánu v případě neznámého kontextu dialplánu, povolení neznámých hovorů a definování transportního protokolu.

```
[general]
binaddr=158.196.244.250
context=incoming_calls
allowguest=yes
transport=udp
```

V druhé sekci mého souboru `sip.conf` je implementována šablona `vtiger` pro všechny sip klienty. Šablony slouží k omezení redundantních konfigurací pro každý účet.

```
[vtiger](!)
host=dynamic

type=friend

disallow=all

allow=alaw

qualify=yes
```

Každý účet má nastavenou šablonu vtiger a dva unikátní parametry - secret reprezentující heslo pro registraci a context odkazující na daný kontext v dialplánu pro řízení hovoru. Ukázka konfigurace SIP klienta Karla Dráta pro jeho CRM extension phone níže.

```
[201] (vtiger)

secret=karelcrm

context=vtiger_outbound
```

Dalším editovaným konfiguračním souborem byl manager.conf. Tento soubor slouží k nastavení AMI rozhraní. Tahle konfigurace je důležitá zejména pro Asterisk Vtiger Connector, který právě využívá Asterisk Manager Interface s kombinací s AGI příkazy pro řízení hovorů. Více o Asterisk Vtiger Connectoru v kapitole 4.4. Stejně jako u sip.conf, se i zde konfiguruje nejdřív obecné nastavení v general sekci a poté uživatelé s přístupem k AMI rozhraní. V mé sekci general je důležité vůbec povolit tento modul, nastavit IP adresu, port rozhraní a zobrazení připojených uživatelů.

```
[general]

enabled=yes

port=5038

binaddr=0.0.0.0

displayconnects=yes
```

Po sekci general jsem přidal uživatele vtiger s heslem vtigerPass a povolenými adresami virtuálního serveru CRM systému a také local hostu, jelikož Asterisk Vtiger Connector je vždy implementován na serveru Asterisku. Důležité body jsou zde parametry read a write, které uvádějí jaká veškerá čtecí a zapisovací práva má daný uživatel. V mém případě by stačila práva system, call, log, verbose, command, agent, user, config a originate. K získání veškerých práv je možné zadáním hodnoty all. Ukázka konfigurace vtiger uživatele níže:

```
[vtiger]

secret=vtigerPass

deny=0.0.0.0

permit=158.196.244.226/255.255.255.0

permit=158.196.244.250/255.255.255.0

permit=127.0.0.1/255.255.255.0

read=all,system,call,log,verbose,command,agent,user,config,
originate
```

```
write=all,system,call,log,verbose,command,agent,user,config,originate
```

Jeden z nejdůležitějších konfiguračních souborů v asterisku `extensions.conf`, neboli dialplán, rozvíjím v mé práci v kapitole 3.6. Poslední dva konfigurační soubory `res_odbc` a `func_odbc`, související s integrací relační databáze CRM systému s Asteriskem, popisují v kapitole 3.5.

3.4 Implementace Vtiger Asterisk Connector

Jak už je zmíněno v kapitole 2.3.1.1 Vtiger Asterisk Connector je aplikace, která se chová jako gateway pro připojení k Vtiger CRM z Asterisk serveru. Vtiger Asterisk Connector je volně ke stažení na stránkách www.vtiger.com/add-ons/. Stáhl jsem si příkazem `wget` Asterisk Connector na virtuální server s Asteriskem a extrahoval jej v domovském adresáři. U Asterisk Connectoru není důležité, kde bude v systému umístěn. Za to je důležité mít v systému Javu, bez které Asterisk Connector nebude spuštěn. Tato webová aplikace je napsána v jazyce Java a je nutné mít v systému Javu verzi 1.7 a vyšší. Je nutno podotknout, že Asterisk Connector pracuje pouze s verzí Asteriskem 1.8 a výše. Nejprve jsem tedy nainstaloval potřebnou Javu a to OpenJDK Runtime Environment s Java verzí 1.7.0. do svého systému příkazem:

```
# apt-get install default-jre
```

Adresář `VtigerAsteriskConnector` obsahuje složky `agi`, `bin`, `conf`, `lib`, `logs` a `webapps`. Složka `agi` obsahuje `jar` soubor `vtigeragi.jar`, `bin` složka obsahuje čtyři skripty pro spuštění aplikace, složka `conf` zase konfigurační soubor `VtigerAsteriskConnector.properties`, `lib` složka obsahuje potřebné knihovny, složka `log`, pro ukládání logů a složka `webapps` kde se nachází samotná aplikace.

Co se týče konfigurace je zde jediný soubor a to `VtigerAsteriskConnector.properties`. Zde se určuje lokace aplikace, lokace nahrávek hovorů a databáze aplikace a detaily Asterisk serveru a Vtiger CRM.

Nejprve tedy IP adresa aplikačního serveru a portu:

```
ServerIP=158.196.244.250
ServerPort=8090
```

Dále povolení nahrávek hovorů a jejich lokace a lokace databáze aplikace:

```
Recording=true
StorageDir=/VtigerAsteriskConnector/records/
AsteriskAppDBPath=/VtigerAsteriskConncetor/db/
```

Detaily Asterisk serveru se jménem a heslem uživatele konfigurovaného v `Asterisk manager.conf` souboru:

```
AsteriskServerPublicIP=158.196.244.250
AsteriskServerIP=0.0.0.0
AsteriskServerPort=5038
AsteriskUsername=vtiger
AsteriskPassword=vtigerPass
```

URL adresa Vtiger CRM systému s generovaným bezpečnostním klíčem:

```
VtigerURL=http://158.196.244.226
```

```
VtigerSecretKey=145145546356ed849952a7b
```

A poslední informace o povolení Asterisk Events a databazových logů:

```
AsteriskLog=yes
```

```
DatabaseLog=yes
```

Po uložení souboru je třeba samotnou aplikaci spustit. V adresáři bin jsou dva skripty pro spuštění a pozastavení aplikace. Skripty start.sh a stop.sh. Zbylé dva skripty není třeba spouštět. Aplikační server jsem tedy spustil příkazem:

```
# ./start.sh
```

Pro kontrolu ověření, zda aplikační server opravdu "běží", jsem si do prohlížeče zadal IP adresu společně s portem. Zobrazila jsem mi defaultní stránka (adresář), jenž potvrzuje správný chod aplikačního serveru. Tuto stránku lze vidět na obrázku 3.4 níže.

Directory: /

agi.sh	339 bytes	Feb 10, 2015 8:48:16 PM
start.sh	228 bytes	Feb 10, 2015 8:48:16 PM
stop.sh	115 bytes	Feb 10, 2015 8:48:16 PM
webapp.sh	377 bytes	Feb 10, 2015 8:48:16 PM

Obrázek 3.4: Vtiger Asterisk Connector startovní stránka běžící aplikace

Vtiger Asterisk Connector tedy není třeba dál modifikovat. Poslední důležitá věc je zavolat incoming.agi v dialplánu v podobném příkazu:

```
exten => _X.,1,Agi(agi://158.196.244.250/incoming.agi)
```

3.5 Integrace Asterisku s databází Vtiger CRM

Integrace Asterisku s podnikovými sítěmi a hlavně s jejich informačními systémy je velice důležitá pro spojení telefonní sítě v podniku a systémovými daty celého podniku. V tom je právě klíčová integrace Asterisku s databází daného podniku a jejího informačního systému. V mém případě tedy integrace s databází Vtiger CRM, kde jsou zaznamenáni jednak uživatelé CRM, přičemž to jsou většinou zaměstnanci podniku, či také důležité kontakty zákazníků. Mimo telefonních kontaktů lze s databáze vyčíst nespočet dalších užitečných informací, například zdali je uživatel přihlášen do systému, zdali volající zákazník je již registrován v systému či ke kterému uživateli CRM je daný zákazník právě přiřazen apod. V podstatě každá změna vytvořena v CRM systému se projeví a zapíše do databáze, ze které můžeme tyto informace číst.

Jak jsem již zmiňoval výše, k propojení Asterisku se vzdálenou databází a přístup k jejímu databázovému systému použiji softwarové API Open Database Connectivity zkráceně ODBC. ODBC umožňuje nezávislý přístup na databázovém či operačním systému. Nejprve jsem si nainstaloval samotné ODBC do mého systému na Asterisk serveru z Debian repositáře příkazem:

```
# apt-get install unixODBC unixODBC-dev
```

Bylo třeba nainstalovat ODBC i s vývojářským balíčkem `unixODBC-dev`, protože jej Asterisk využívá k sestavení ODBC modulů. Zapotřebí je nainstalovat konektor k vybranému databázovému systému, v mém případě MySQL. Nainstaloval jsem si tedy MySQL ODBC konektor z Debian repositáře příkazem:

```
# apt-get install libmyodbc
```

Po instalaci je potřeba konfigurace a ta se v případě ODBC samotného týká dvou souborů v adresáři `/etc/`. Jsou to soubory `odbcinst.ini` a `odbc.ini`. Konfigurace MySQL ODBC ovladače se provádí v souboru `odbcinst.ini`, kde je mimo jiné zapotřebí nastavit cestu k souborům `libmyodbc.so` a `libodbcmyS.so`. Oba soubory se vyskytují převážně v adresáři `/usr/lib/odbc/`, ovšem záleží na verzi operačního systému. Konfigurace `odbcinst.ini` na mém Asterisk serveru vypadala následovně:

```
[MySQL]
Description = ODBC for MySQL
Driver       = /usr/lib/x86_64-linux-gnu/odbc/libmyodbc.so
Setup       = /usr/lib/x86_64-linux-gnu/odbc/libodbcmyS.so
FileUsage   = 1
```

Konfigurace identifikátoru pro Asterisk, kde a na jakou databázi se má Asterisk odkázat, se provádí v souboru `odbc.ini`. Zde se nastavuje vybraný ovladač ze souboru `odbcinst.ini`, název databáze a server, kde se daná databáze nachází a také cesta soket souboru `mysql.sock` či `mysqld.sock`, který se nachází na MySQL serveru s danou databází. Má konfigurace `odbc.ini` vypadala následovně:

```
[asterisk-identifier]
Description = MySQL connection to 'vtigercrm' database
Driver      = MySQL
Database    = vtigercrm
Server      = 158.196.244.226
Socket      = /run/mysqld/mysqld.sock
```

Následně jsem si vytvořil v MySQL uživatele pro Asterisk, který bude moci pracovat s databází `vtigercrm`. Na CRM serveru jsem se tedy přepnul do MySQL a vytvořil jsem si uživatele `asterisk` s hesle `uniNetIsSafe`, který bude vlastnit veškerá práva na již vytvořenou databázi `vtigercrm` a bude se moci připojit z jakéhokoliv host serveru pomocí symbolu `%`. Postu byl následující:

```
# mysql -u root -p
mysql> CREATE USER 'asterisk'@'%' IDENTIFIED BY 'uniNetIsSafe';
mysql> GRANT ALL PRIVILEGES ON vtigercrm.* TO 'asterisk'@'%';
```

Na straně MySQL serveru byla zapotřebí ještě jedna úprava v konfiguraci souboru */etc/mysql/my.cnf*. A to zakomentování řádků s *bind-address*, které by jinak umožňovalo naslouchání MySQL serveru pouze na localhostu z bezpečnostních důvodů. Řádek v souboru *my.cnf* tedy poté vypadal následovně:

```
#bind-address = 127.0.0.1
```

Verifikaci správné konfigurace a připojení ODBC konektoru do MySQL databáze jsem si ověřil pomocí aplikace *isql*, kde jsem si v příkazové řádce napsal primitivní dotaz s pipe přepínačem do *isql* se zadaným identifikátorem a MySQL uživatelem s heslem:

```
# echo "select 1" | isql -v asterisk-connector asterisk uniNetIsSafe
+-----+
| Connected! |
|           |
| sql-statement |
| help [tablename] |
| quit |
|           |
+-----+
SQL> select 1
+-----+
| 1 |
+-----+
| 1 |
+-----+
SQLRowCount returns 1
1 rows fetched
```

Před konfigurací ODBC na straně Asterisku je třeba verifikovat, zda jsou všechny potřebné ODBC související moduly zapnuté a to konkrétně *cdr_odbc*, *cdr_adaptive_odbc*, *func_odbc*, *func_realtime*, *pbx_realtime*, *res_config_odbc*, a *res_odbc*. V Asterisku 11 již jsou veškeré tyto moduly nainstalované. Všechny existující moduly Asterisku se nacházejí v adresáři */usr/lib/asterisk/modules*. K povolení připojení Asterisku k databázi skrze ODBC slouží konfigurační soubor */etc/asterisk/res_odbc.conf*. Zde se konfiguruje identifikátor ze souboru *odbc.ini* a dále MySQL uživatel s heslem k cílené databázi. Soubor *res_odbc.conf* v mé konfiguraci:

```
[vtigercrm]
enabled => yes
dsn => asterisk-connector
username => asterisk
```

```
password => uniNetIsSafe
```

Pro verifikaci připojení jsem si v Asterisk příkazové řádce zadal příkaz níže:

```
CLI> odbc show
```

```
ODBC DSN Settings
```

```
-----
```

```
Name:    vtigercrm
```

```
DSN:    asterisk-connector
```

```
Last connection attempt: 1970-01-01 01:00:00
```

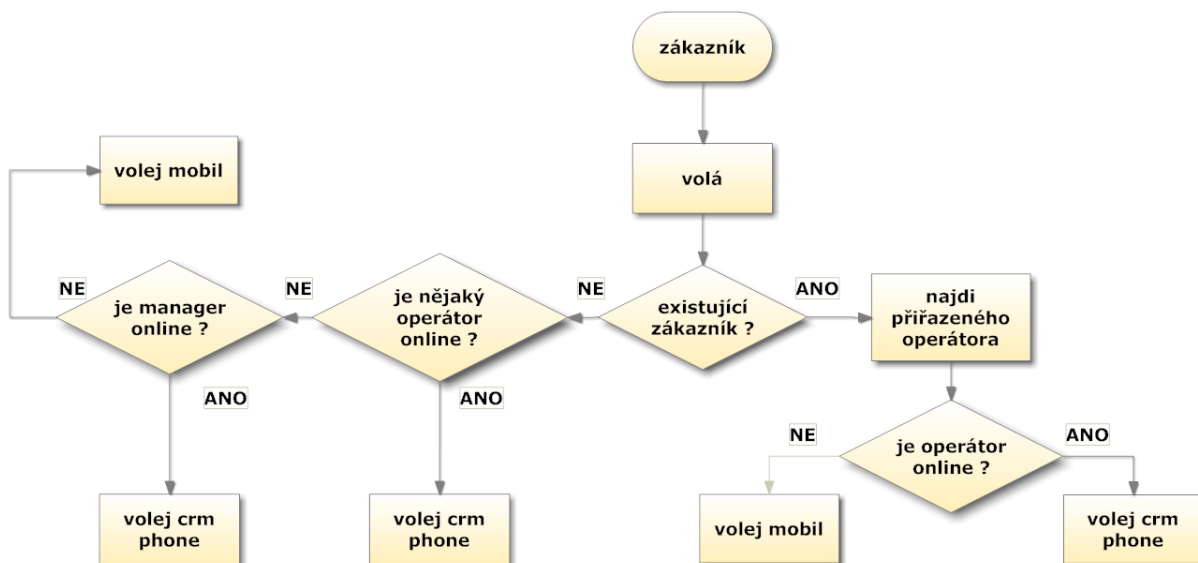
```
Pooled: No
```

```
Connected: Yes
```

Poslední konfigurovaný soubor z ODBC v Asterisku je soubor `func_odbc.conf`. Jak již bylo vysvětleno v kapitole 3.6.1, v souboru `func_odbc.conf` se implementují jednotlivé SQL dotazy, které se volají pomocí dialplán funkce ODBC v dialplánu. Detailní rozbor mé konfigurace a jednotlivých dotazů rozepisují v následující kapitole návrhu dialplánu.

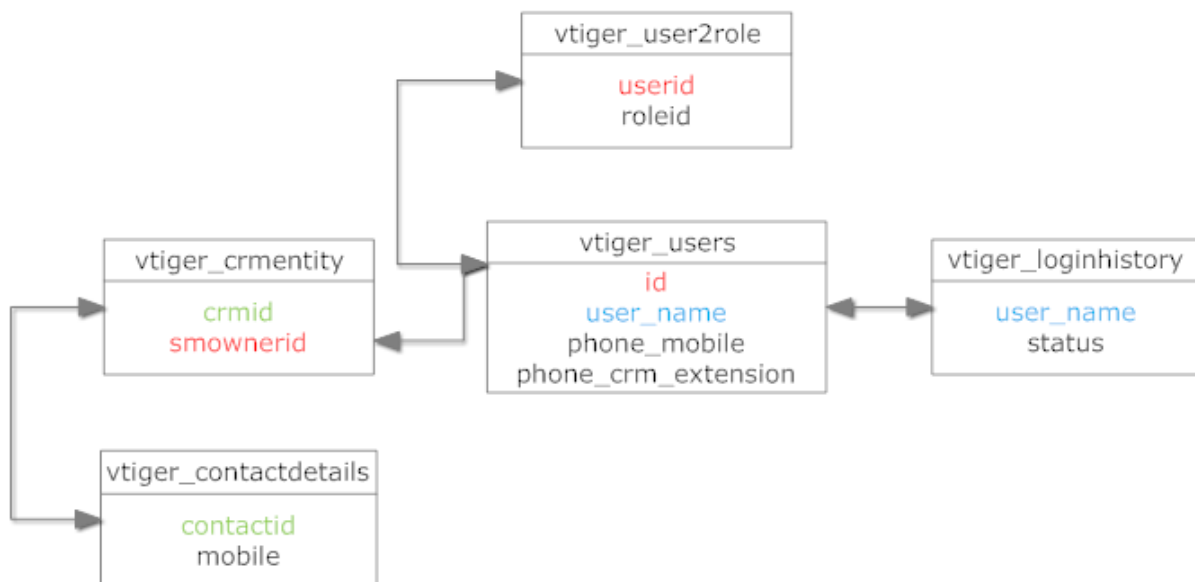
3.6 Návrh a implementace dialplánu

Praktická realizace mé práce spočívala v integraci pobočkové ústředny se CRM fiktivního podniku o dvou operátorech a jednoho manažera. Nejprve jsem si navrhnul diagram průběhu hovoru zákazníka na support infolinku podniku. Manažer a operátoři pracují se CRM systémem a jsou v kontaktu se zákazníky. Zákazníci jsou zde velice důležití, a proto se v případě nepřítomnosti online operátorů hovor přesměrovává na manažera. Pokud je však zákazník již registrovaný a je přiřazen k jednomu z operátorů, hovor se v nepřítomnosti operátora u PC přesměruje na jeho mobilní telefon. V případě, že je operátor nebo manažer online, je hovor přesměrován na jejich CRM phone neboli softphone nainstalovaný v pracovním počítači. Když přijde operátor do práce, má zapnutý CRM phone registrovaný k Asterisk serveru a přihlásí se do Vtiger CRM systému. Při odchodu z práce se vždy zase odhlásí. Diagram hovoru zákazníka na support infolinku je zobrazen na obrázku 3.5.



Obrázek 3.5: Diagram hovoru zákazníka na support infolinku

Další fáze byla zjištění datového modelu vtigercrm databáze. Jelikož jsem pracoval s databází Vtiger CRM, potřeboval jsem znát její strukturu. Na stránkách wiki.vtiger.com je k dispozici pdf soubor s datovým modelem k verzi 5.2.1. Tento model byl hodně nápomocný, avšak ne zcela dostačující. Pro zjištění dalších potřebných dat jsem prošel jednotlivé tabulky v MySQL na mém virtuálním serveru. Rozepsal jsem si jednotlivé entity a označil si jejich klíče pro spojení mezi nimi. Potřeboval jsem získat data ohledně identifikačních čísel uživatelů a zákazníků, čísla CRM a mobilních telefonů uživatele a zákazníka, rolích uživatelů a jejich statusu, zda jsou zrovna uživatelé přihlášení v systému. Na obrázku 3.6 jsou zobrazeny potřebné entity z vtigercrm databáze a jejich vztahy.



Obrázek 3.6: Vztahy mezi entitami databáze vtigercrm

Pro získání potřebných dat k implementaci dialpánu ve svých dotazech využívám pět tabulek zobrazených na obrázku 3.6. Z tabulky vtiger_user2role získávám informace o rolích, tedy zda-li je uživatel operátor nebo manažer. Role uživatelů ve Vtiger CRM jsou v databázi rozděleny do id hodnot

Integrace Asterisku s Vtiger CRM

H1 až H5, kde H1 je role nejvyšší. Id hodnota manažera je H4 a operátora H5. Níže je zobrazen výpis z tabulky vtiger_user2role, kde je možné vidět dva operátory, jednoho manažera a také admin uživatele, který byl vytvořen při instalaci CRM systému.

```
mysql> SELECT * FROM vtiger_user2role;
```

```
+-----+-----+
| userid | roleid |
+-----+-----+
|      1 | H2     |
|      7 | H4     |
|      5 | H5     |
|      6 | H5     |
+-----+-----+
```

Další data získávám z tabulky uživatelů vtiger_users, která je spojovacím prvkem pro tabulky user2role, loginhistory a crmetnity. Z této tabulky získávám data ohledně telefonních čísel uživatelů. Výpis viz níže:

```
mysql> SELECT id, user_name, phone_mobile, phone_crm_extension FROM vtiger_users;
```

```
+----+-----+-----+-----+
| id | user_name | phone_mobile | phone_crm_extension |
+----+-----+-----+-----+
|  1 | admin     | 777         | 101                 |
|  5 | Karel Drat | 701         | 201                 |
|  6 | Pavel Linka | 702         | 202                 |
|  7 | Jan Hlava  | 605         | 301                 |
+----+-----+-----+-----+
```

V tabulce vtiger_loginhistory jsou zaznamenávány jednotlivé přihlášení a odhlášení uživatelů. To jsou nápomocné informace k zjištění statusu uživatele:

```
mysql> SELECT login_id, user_name, status FROM vtiger_loginhistory;
```

```
+-----+-----+-----+
| login_id | user_name | status      |
+-----+-----+-----+
|      1 | admin     | Signed in  |
|      2 | admin     | Signed off |
|      3 | Jan Hlava | Signed in  |
```

Integrace Asterisku s Vtiger CRM

```
|          4 | Karel Drat | Signed off |
|          5 | Karel Drat | Signed in  |
|          6 | admin      | Signed in  |
|          7 | Jan Hlava  | Signed off |
|          8 | Pavel Linka| Signed in  |
|          9 | Karel Drat | Signed off |
|         10 | admin      | Signed in  |
|         11 | Pavel Linka| Signed in  |
+-----+-----+-----+
```

Tabulka `vtiger_contactdetails` obsahuje data id zákazníka a jeho číslo mobilního telefonu. Tabulka `vtiger_crmentity` obsahuje spoustu dat o každé další vzniklé entitě ve Vtiger CRM. Tuto tabulku jsem využil k získání informací o přiřazení uživatele k danému zákazníkovi. Zákazníky jsem si pro demonstraci vytvořil tři, viz výpis z tabulky níže:

```
mysql> SELECT contactid, mobile, lastname FROM
vtiger_contactdetails;
```

```
+-----+-----+-----+
| contactid | mobile | lastname |
+-----+-----+-----+
|          2 | 501    | Novak    |
|          4 | 721    | Hrdinova |
|          5 | 737    | Haluz    |
+-----+-----+-----+
```

Veškeré potřebné informace byly dostupné a začal jsem s implementací skriptu dialplánu. Dialplán se konfiguruje v souboru `/etc/asterisk/extensions.conf`. Pro lepší přehlednost jsem smazal všechna defaultní data vyskytující se v konfiguračním souboru dialplánu. Dialplán jsem rozdělil do dvou kontext sekcí. První sekce se nazývala `incoming_calls` a sloužila pro příchozí hovory zákazníků na support infolinku. Druhá sekce měla název `vtiger_outbound` a sloužila pro řízení hovorů operátorů CRM. Kontext `incoming_calls` ve skriptu dialplánu kopíroval diagram na obrázku 3.5. S implementací dialplánu jsem postupně vytvářel i SQL dotazy pro `func_odbc`.

Při příchodu hovoru zákazníka se jako první hovor vyzvedne a přesměruje na první úsek `check_customer`. V tomto úseku se získává id zákazníka díky telefonního čísla volajícího, což umožní funkci `func_odbc` s názvem `GET_CUSTOMER`, a uloží zmíněné id do proměnné `CUST_ID`. Následující krok ověří, zda volající funkce `GET_CUSTOMER` našla id v záznamu s odpovídajícím volajícím číslem. Pokud ne, skript přesměruje další krok na úsek `check_users`, pokud ano, skript dále pokračuje v dalším kroku a to získáním id uživatelem CRM, ke kterému je volající zákazník přiřazen. Opět pomocí funkce `func_odbc` a to `GET_OWNER`. V následujících krocích se provedou další dvě `func_odbc` funkce - `GET_NAME` a `CHECK_STATUS`. Na závěr této sekce je objevuje podmínka,

zda je daný uživatel přihlášen v systému. Pokud ano, je další krok přesměrován do sekce `get_phone` pro získání CRM extension čísla, pokud ne, bude dalším krokem přesměrování do sekce `get_mobile` pro získání čísla mobilního telefonu daného uživatele. Viz níže.

```
exten => check_customer,1,NoOp()

same => n,Set(CUST_ID=${GET_CUSTOMER(${CALLERID(num)})})

same => n,GotoIf($[${CUST_ID} < 1]?check_users,1)

same => n,Set(USER=${GET_OWNER(${CUST_ID})})

same => n,Set(USER_NAME=${GET_NAME(${USER})})

same => n,Set(STATUS=${CHECK_STATUS(${USER_NAME})})

same => n,GotoIf($[${STATUS} = "Logged In"?get_phone,1)

same => n,Goto(get_mobile)
```

Výše zmíněný úsek obsahuje čtyři `func_odbc` funkce. Všechny tyto funkce je třeba naimplementovat v konfiguračním souboru `func_odbc.conf`. Veškeré tyto funkce jsem naimplementoval do jednotlivých sekcí s vlastním prefixem, většinou `GET` nebo `CHECK`. Vlastní zvolené prefixy poté nahradí defaultní prefix `ODBC` v diaplánu. U každé funkce jsou konfigurovány základní parametry. Důležitý parametr je `dsn`, který odkazuje na vybranou extension v souboru `res_odbc.conf` a ta dále směřuje dotazy do konfigurované databáze. V mém případě směřuji veškeré dotazy do jediné vtigercrm databáze, tím pádem u každé funkce konfigurované v souboru `func_odbc` bude mít vždy parametr `dsn` nastavenou hodnotu "vtigercrm". U první použité funkce `GET_CUSTOMER` se předává argument `${CALLERID(num)}`, což je Asterisk funkce, jenž získá číslo volajícího. Implementovaná funkce `GET_CUSTOMER` obsahuje SQL dotaz získávající id zákazníka z tabulky `vtiger_contactdetails`, pokud číslo volajícího se bude shodovat s hledaným záznamem. Funkce `GET_CUSTOMER` vypadá následovně:

```
[CUSTOMER]

prefix=GET

dsn=vtigercrm

readsql=SELECT contactid FROM vtiger_contactdetails WHERE mobile =
readsql+= '${ARG1}'
```

Následující funkce `GET_OWNER` slouží k získání id uživatele přiřazeného k volajícímu zákazníkovi. Tabulka `vtiger_contactdetails` bohužel neobsahuje atribut id přiřazeného uživatele, a proto je potřeba získat id volajícího a vytvořit další dotaz nad tabulkou `vtiger_crmentity` a tím získat id přiřazeného uživatele. Ve funkci `GET_OWNER` se předává argument id zákazníka a vypadá následovně:

```
[OWNER]

prefix=GET

dsn=vtigercrm

readsql=SELECT smownerid FROM vtiger_crmentity WHERE crmid =
readsql+= '${ARG1}'
```

Funkci GET_NAME jsem implementoval proto, abych získal user_name daného uživatele pro další dotaz k získání statusu. V tabulce vtiger_login history totiž nenajdeme atribut id uživatele, proto klíčovým spojovacím prvkem je zde atribut user_name. Nejprve jsem tedy získal atribut user_name funkcí GET_NAME, kde je předáván argument id uživatele a poté jsem zjistil status daného uživatele funkcí CHECK_STATUS s argumentem uživatelského jména. Tabulka vtiger_loginhistory obsahuje záznamy stavů jednotlivých uživatelů. Uživatel vytvoří nový záznam v databázi při každém odhlášení či přihlášení uživatele. Poněvadž funkce CHECK_STATUS nemá povolený parametr mode na multirow pro získání pole více záznamů z tabulky, vybere nám první nalezený řádek. Abychom se ujistili, že vybraný status daného uživatele je právě ten aktuální, seřadíme výsledné řádky dle dat login_id sestupně, protože při každém či odhlášení se zaznamenává nové id, tudíž nejvyšší id je právě to aktuální. Funkce GET_NAME a CHECK_STATUS vypadají následovně:

```
[NAME]
```

```
prefix=GET
```

```
dsn=vtigercrm
```

```
readsql=SELECT user_name FROM vtiger_users WHERE id = '${ARG1}'
```

```
[STATUS]
```

```
prefix=CHECK
```

```
dsn=vtigercrm
```

```
readsql=SELECT status FROM vtiger_loginhistory WHERE user_name =  
readsql+= '${ARG1}' ORDER BY login_id DESC
```

V dalším úseku dialplánu check_users prochází skript jednotlivé operátory a zjišťuje jejich status. Je-li první nalezený uživatel online, skript se přesměruje na úsek get_phone, kde se dále získá číslo softphonu operátora. Pokud není první nalezený operátor online, smyčka se opakuje s dalšími operátory. V případě, že není žádný z operátorů online, je skript přesměrován na úsek call_manager. Umožnění této iterace a práce s více řádky z jednoho dotazu je díky funkci ODBC_FETCH, která prochází jednotlivé řádky z pole dat. Pole dat získáme nastavením parametru mode na multirow v případě, že očekáváme, že vybraný dotaz vrátí více hodnot. Další Asterisk ODBC funkcí je ODBCROWS zjišťující aktuální počet řádků. Úsek check_users z dialplánu viz níže:

```
exten => check_users,1,NoOp()
```

```
same => n,Set(ID=${GET_OPERATOR()})
```

```
same => n(loop),NoOp()
```

```
same => n,Set(USER=${ODBC_FETCH(${ID})})
```

```
same => n,GotoIf(${${ODBCROWS} < 1}?call_manager,1)
```

```
same => n,Set(USER_NAME=${GET_NAME(${USER})})
```

```
same => n,Set(STATUS=${CHECK_STATUS(${USER_NAME})})
```

```
same => n,GotoIf(${${STATUS} = "Logged In"}?get_phone,1)
```

```
same => n,Goto(loop)
```

V tomto úseku jsem použil tři vlastní func_odbc funkce a to GET_OPERATOR a již známé GET_NAME a CHECK_STATUS. GET_OPERATOR funkce nepředávám žádný argument, jelikož, chci hledat veškeré záznamy s jasně danou roleid H5 neboli veškeré operátory. Funkce GET_OPERATOR vypadá následovně:

```
[OPERATOR]
prefix=GET
dsn=vtigercrm
mode=multirow
readsql=SELECT userid FROM vtiger_user2role WHERE roleid = 'H5'
```

Následující dva úseky get_phone a get_mobile jsou si velmi podobné. Úseku get_phone začíná krokem ODBCFinish(), který ukončí funkci ODBC_FETCH a jako parametr si předá posledně zpracovávaný argument. V mém případě tedy id uživatele. Podle id uživatelé poté najde funkce GET_EXTENSION dané číslo CRM telefonu. Poté je úsek přesměrován na získané extension a provede incoming.agi skript, který zahájí hovor. Ve funkci Goto jsou zadány všechny tři parametry pro případ zahrnutí incoming_calls kontextu do jiného kontextu pomocí include a tedy vyvarování se nesprávnému přesměrování. Jak úsek get_phone tak get_mobile používají funkci GET_EXTENSION pro získání čísla CRM či mobilního telefonu. Proto se v této funkci předávají dva argumenty, z nichž je první statický, a to cílený objekt vyhledávacího dotazu, tedy phone_crm_extension nebo phone_mobile.

```
exten => get_phone,1,NoOp()
same => n,ODBCFinish(${USER})
same => n,Set(EXT=${GET_EXTENSION(phone_crm_extension,${USER})})
same => n,Goto(incoming_calls,${EXT},1)

exten => get_mobile,1,NoOp()
same => n,Set(EXT=${GET_EXTENSION(phone_mobile,${USER})})
same => n,Goto(incoming_calls,${EXT},1)
```

Funkce GET_EXTENSION v func_odbc.conf tedy vypadá následovně:

```
[EXTENSION]
prefix=GET
dsn=vtigercrm
readsql=SELECT ${ARG1} FROM vtiger_users WHERE id = '${ARG2}'
```

Předposlední úsek v incoming_calls extension dialplánu je úsek call_manager. Skript se přesměruje na tento úsek v případě, že žádný z operátorů není online při příchodu hovoru neznámého zákazníka. Obdobně jako v úseku check_users, zde začíná skript podobnou funkcí GET_MANAGER, v tomto případě s daným argumentem H4 neboli roleid manažerů. V mém návrhu fiktivní obchodní

společnosti figuruje vždy pouze jeden manažer, což znamená, že není zapotřebí procházet více řádků funkcí ODBC_FETCH.

```
exten => call_manager,1,NoOp()
same => n,Set(ID=${GET_MANAGER()})
same => n,Set(USER_NAME=${GET_NAME(${ID})})
same => n,Set(STATUS=${CHECK_STATUS(${USER_NAME})})
same => n,GotoIf($[${STATUS} = "Logged In"]?get_phone,1)
same => n,Goto(get_mobile)
```

V posledním úseku již zahájíme hovor pomocí agi skriptu incoming.agi Asterisk konektoru pomocí FastAgi funkce. Tento skript nám poté vyvolá funkce Dial a Monitor pro vytočení daného čísla a nahrávání hovoru. Zde jsem zvolil dynamický parametr `_X.` pro jakékoliv číslo, jelikož čísla extensions a mobilních telefonů operátorů se mohou měnit.

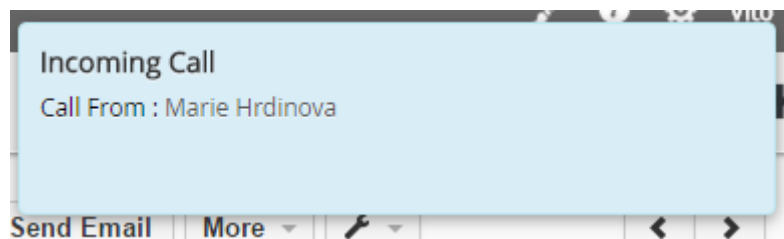
```
exten => _X.,1,Agi(agi://158.196.244.250/incoming.agi)
same => n,Hangup()
```

Další kontext v dialplánu patřil veškerým uživatelům CRM systému. Kontext vtiger_outbound odkazoval na již zmíněný agi skript incoming.agi při příchozím nebo odchozím hovoru jakéhokoliv čísla.

```
exten => _X.,1,Agi(agi://158.196.244.258/incoming.agi)
same => n,Hangup()
```

3.7 Modifikace notifikace příchozího hovoru

Jedna z dovedností PBXManager v systému Vtiger CRM je zobrazení notifikačního okna při příchozím hovoru. Notifikační okno zobrazuje jméno zákazníka, zdali se nachází v systému, či v opačném případě pouze číslo volajícího. V případě neznámého volajícího se v notifikačním okně zobrazí i textové pole, výběrové pole a tlačítko na uložení kontaktu. Defaultní notifikační okno, viz obrázek 3.7 níže.



Obrázek 3.7: Defaultní notifikační okno příchozího hovoru ve Vtiger CRM

V mé práci jsem si toto notifikační okno upravil, dle požadavků fiktivní firmy, která chtěla zobrazit v notifikaci více informací o zákazníkovi. Nově modifikované notifikační okno by mělo zobrazovat údaje o čísle volajícího, emailu a organizaci. Mimo tyto údaje jsem doplnil notifikační okno o logo školy. Nejprve jsem provedl exploraci souborů PBXManager modulu. Tento modul obsahuje jedenáct PHP souborů, jeden XML soubor a jeden JavaScript soubor. Tyto soubory se nachází v adresáři `modules/PBXManager/`. Notifikační okno se vytváří a "volá" pomocí JavaScriptu

v souboru *PBXManagerJS.js* v adresáři *resources*. V adresáři *resources* jsem si vytvořil PHP soubor s názvem *dbqueries.php* pro získání dat z databáze *vtigercrm*. V souboru *dbqueries.php* jsem si vytvořil připojení do databáze pomocí funkce *mysqli*, viz kód níže:

```
$servername = "localhost";
$username = "asterisk";
$password = "uniNetIsSafe";
$databasename = "vtigercrm";
$db = new mysqli($hostname, $username, $password, $databasename);
```

Poté jsem si vytvořil pole, do kterého ukládám výsledná data a proměnnou *recordid*, použitou k uložení dat přijatých pomocí globální proměnné *\$_GET*, konkrétně číslo volajícího, reprezentované parametrem *num*, viz kód níže.

```
$data = array();
$recordid = $_GET["num"];
```

Číslo volajícího se získává přímo v souboru *PBXManagerJS.js*, ovšem bylo potřeba získat email a název organizace, kterou případně zákazník zastupuje. Tyto informace jsem získal pomocí SQL dotazů nad databází *vtigercrm*. Získané informace se vložili do pole a v případě žádného výsledku se do pole zapsal textový řetězec "undefined" pro zobrazení neznámého výsledku, viz kód níže.

```
$result = $db->query("SELECT email FROM vtiger_contactdetails WHERE
mobile = ".$recordid.");
if ($result->num_rows > 0) {
    while($row = $result->fetch_assoc()) {
        array_push($data, $row["email"]);
    }
} else array_push($data, "undefined");
```

V případě získání informací o jméně organizace vypadal blok kódu totožně jako v případě získání emailu. Jediný rozdíl byl v SQL dotazu, kde jsem musel využít funkci *INNER JOIN* pro spojení dvou tabulek - tabulky zákazníka *contactdetails* a *account* tabulky organizace, viz kód níže.

```
$result = $db->query("SELECT accountname FROM vtiger_account INNER
JOIN vtiger_contactdetails ON
vtiger_account.accountid=vtiger_contactdetails.accountid WHERE
vtiger_contactdetails.mobile = ".$recordid.");
```

Nakonec jsem výsledné pole převedl do JSON formátu pro snadnější práci s daty v JavaScriptu a uzavřel spojení s databází, viz kód níže.

```
echo json_encode($data);
mysqli->close();
```


Po vytvoření PHP souboru pro získání dat z databáze jsem si upravil JavaScriptový soubor pro zobrazení notifikačního okna *PBXManagerJS.js*. Soubor obsahuje několik úseků, avšak důležité úseky pro modifikaci notifikace byly funkce *showPBXIncomingCallPopup* pro zobrazení notifikačního okna a funkce *requestPBXgetCalls* pro přijímání žádosti od pobočkové ústředny k vyvolání notifikačního okna. Oba tyto úseky bylo potřeba upravit. Ve funkci *requestPBXgetCalls* se získávají informace o volajícím pomocí funkce *searchIncomingCalls* z PHP souboru *IncomingCallPoll* a následně se informace uloží do proměnné *record*, viz kód níže.

```
requestPBXgetCalls : function() {  
    var url =  
'index.php?module=PBXManager&action=IncomingCallPoll&mode=searchIncomingCalls';  
  
    AppConnector.request(url).then(function(data) {  
        if(data.success && data.result) {  
            for(i=0; i< data.result.length; i++) {  
                var record = data.result[i];
```

Poté následovala podmínka, kde v případě pravdivosti byla volána funkce *showPBXIncomingCallPopup*, viz kód níže.

```
if(jQuery('#pbxcall_'+record.pbxmanagerid+'').size()== 0 ) {
```

Před samotným zavoláním funkce *showPBXIncomingCallPopup* bylo potřeba předat této funkci další parametr obsahující JSON soubor se získanými daty. K získání informací jsem použil jQuery funkci *ajax* umožňující provedení asynchronního HTTP požadavku. AJAX umožňuje získat a zobrazit data bez obnovení webové stránky. *Ajax* funkce obsahuje parametr *url*, jenž obsahuje URL adresu, na kterou je požadavek zaslán. V parametru *url* jsem tedy nastavil cestu k souboru *qbqueris.php* společně s předaným parametrem *num*. Parametru *num* předávám číslo volajícího reprezentované jako proměnnou *customernumber* z proměnné *record*. V případě úspěšného spojení jsem zavola funkci *showPBXIncomingCallPopup* a předal parametr *response*, viz kód níže.

```
jQuery.ajax({  
    type: "GET",  
    url:  
"/modules/PBXManager/resources/dbqueries.php?num="+record.customernumber,  
    success:function(response) {  
Vtiger_PBXManager_Js.showPBXIncomingCallPopup(record, response);  
    }  
});
```

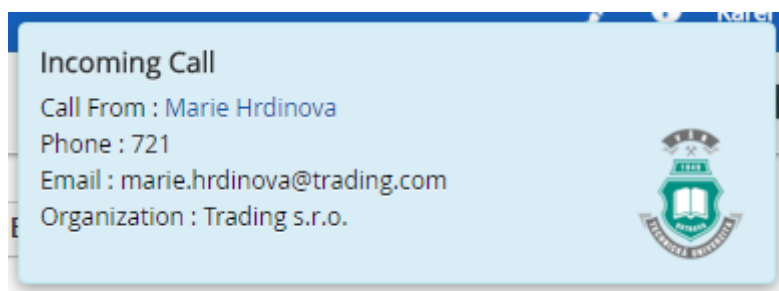
V dalším úseku se definuje zmíněná funkce *showPBXIncomingCallPopup* pro samotné zobrazení notifikačního okna, které navíc předávám parametr *response* reprezentující JSON soubor se získanými daty z databáze. Nejprve jsem inicializoval proměnnou *data* rozdělením JSON souboru, viz kód níže.

```
showPBXIncomingCallPopup : function(record, response) {  
var data = JSON.parse(response)
```

Funkce *showPBXIncomingCallPopup* obsahovala proměnnou *params* obsahující atributy k vytvoření notificačního okna. Proměnná *params* se poté volá ve funkci *Vtiger_Helper_Js.showPnotify*. Důležitý atribut proměnné *params* je atribut *text*, jenž obsahuje HTML kód pro zobrazení notificačního okna. Atribut *text* jsem upravil o doplňující *span* a *img* elementy pro zobrazení potřebných dat. Pro zobrazení emailu a jména organizace daného uživatele jsem ve *span* elementu zavolał proměnnou *data* s vybraným prvkem pole z JSON souboru, viz kód níže.

```
<span>Email : '+data[0]+'</span>
```

Výsledné nově modifikované notificační okno se zobrazilo se všemi získanými informacemi a logem, viz obrázek 3.8. Kompletní kód souboru *PBXManagerJS.js* v příloze.



Obrázek 3.8: Modifikované notificační okno příchozího hovoru ve Vtiger CRM

4 Testování a analýza hovorů

Testovací hovory probíhaly na laptopu s několika softphony. Během testování jsem neustále sledoval výpis dění hovorů přes příkazovou řádku Asterisku. Otestoval jsem veškeré možné varianty hovorů, a tím i správnou implementaci dialplánu a ODBC funkcí s SQL dotazy. Testoval jsem také samotný chod Vtiger Asterisk Connectoru a PBXManageru a jejich vlastností, jako zaznamenávání logů, nahrávání hovorů a interakci v CRM systému spojenou s Asteriskem jako je funkce Click to Dial a zobrazení notifikačního okna s rozšířenými informacemi.

Vtiger CRM nabízí funkci Click to Dial umožňující vytočit zvolené číslo kliknutím na záznam telefonního čísla v CRM systému. Je důležité, aby daný uživatel měl registrovaného SIP klienta se stejným číslem extension jako v nastavení CRM. Před samotným testováním jsem se přihlásil přes telnet na Asterisk server jako AMI uživatel, abych mohl sledovat AMI komunikaci. Pro přihlášení k AMI rozhraní je potřeba napsat příkazy v provedení "příkaz: hodnota". Nakonec je potřeba dvakrát odřádkovat pro zaslání příkazů, viz příkazy níže.

```
# telnet 158.196.244.250 5038
Action: Login
ActionID: 1
Username: vtiger
Secret: vtigerPass
```

Přihlásil jsem se jako uživatel Jan Hlava do CRM systému a registroval jsem si SIP klienta 301 přes aplikaci Zoiper. Přes aplikaci Xlite jsem si naopak registroval klienta 201 operátora Karla Dráta. V CRM systému jsem tedy v seznamu uživatelů našel záznam Karla Dráta a kliknul na jeho CRM phone extension. Očekával jsem od Click to Dial funkce okamžité vytočení zvoleného čísla z mého softphonu, místo toho mi vyzváněl můj klient 301 s příchozím hovorem od klienta 201. Ovšem po vyzvednutí tohoto hovoru se hovor přeměroval a můj klient 301 rázem automaticky vytočil klienta 201. Vyzvedl jsem hovor klientem 201 a nahrál testovací záznam. Hovor byl úspěšný. Výpis Asterisk příkazová řádka, viz níže:

```
== Manager 'vtiger' logged on from 127.0.0.1
  == Using SIP RTP CoS mark 5
    > 0x7f224c015b90 -- Probation passed - setting RTP source
address to 158.196.194.87:8000
    > Channel SIP/301-00000003 was answered
  -- Executing [301@vtiger_outbound:1] AGI("SIP/301-00000003",
"agi://158.196.244.250/incoming.agi") in new stack
    > 0x7f224c015b90 -- Probation passed - setting RTP source
address to 158.196.194.87:8000
  == Manager 'vtiger' logged on from 127.0.0.1
```

Testování a analýza hovorů

```
-- AGI Script Executing Application: (Monitor) Options:
(wav,/VtigerAsteriskConnector/records//11b85c4327b649e1a621a932f1d49
826,m)

-- AGI Script Executing Application: (Dial) Options: (SIP/201,
60)

== Using SIP RTP CoS mark 5

-- Called SIP/201

-- SIP/201-00000004 is ringing

> 0x7f225402da20 -- Probation passed - setting RTP source
address to 158.196.194.87:57292

-- SIP/201-00000004 answered SIP/301-00000003

> 0x7f225402da20 -- Probation passed - setting RTP source
address to 158.196.194.87:57292

== Manager 'vtiger' logged off from 127.0.0.1

-- <SIP/301-00000003>AGI Script
agi://158.196.244.250/incoming.agi completed, returning 0
```

AGI skript tedy vyvolal dvě aplikace a to Monitor pro nahrávání hovoru a Dial pro vytvoření daného uživatele. Před začátkem AMI komunikace se Vtiger Asterisk Connector přihlásí z localhostu jako *vtiger* uživatel na AMI rozhraní. Jelikož jsem se připojil jako stejný uživatel, je možné vidět veškeré AMI komunikaci. V mém testovacím hovoru klienta 301 na klienta 201 se provedlo celkem 89 Event událostí, z nichž osm bylo AGIExec. Prvních osm událostí probíhalo v následujícím pořadí: Newchannel, VarSet, NewAccountCode, NewCallerid, Newstate, Newstate, Newexten, AGIExec. Pro ukázkou jsem je zde vidět rozbor pár AMI událostí, např. Newchannel událost pro vytvoření nového kanálu:

```
Event: Newchannel
Privilege: call,all
Channel: SIP/301-00000003
ChannelState: 0
ChannelStateDesc: Down
Context: vtiger_outbound
Uniqueid: 1459073568.3
```

Každá událost má své privilegium. Tato privilegia jsou rozdělena do tříd oblastí týkajících se a jsou přiřazena ke každému uživateli. Aby uživatel mohl vytvořit nový kanál přes AMI rozhraní je potřeba, aby daný uživatel měl povolené privilegium *call* nebo *all*. V následující ukázce Newstate události je zobrazena změna stavu a to vyzvánění uživatel 201, viz níže.

```
Event: Newstate
Privilege: call,all
```

Testování a analýza hovorů

Channel: SIP/301-00000003
ChannelState: 5
ChannelStateDesc: Ringing
CallerIDNum: 201
ConnectedLineNum: 201
Uniqueid: 1459073568.3

Následující událost týkající se dialplánu je událost Newexten. Událost s vytvořenou extension 301 provede spuštění AGI aplikace v dialplánu, viz níže:

Event: Newexten
Privilege: dialplan,all
Channel: SIP/301-00000003
Context: vtiger_outbound
Extension: 301
Priority: 1
Application: AGI
AppData: agi://158.196.244.250/incoming.agi
Uniqueid: 1459073568.3

Ukázka spolupráce AMI události a AGI příkazů je událost AGIExec, která spustí AGI příkaz Monitor pro nahrávání hovoru, viz níže:

Event: AGIExec
Privilege: agi,all
SubEvent: Start
Channel: SIP/301-00000003
CommandId: 96235725
Command: EXEC "Monitor"
"wav,/VtigerAsteriskConnector/records//11b85c4327b649e1a621a932f1d49826,m"

Poslední ukázkou AMI události je Bridge událost spojující dva kanály, viz níže:

Event: Bridge
Privilege: call,all
Bridgestate: Link
Bridgetype: core
Channel1: SIP/301-00000003
Channel2: SIP/201-00000004

Testování a analýza hovorů

Uniqueid1: 1459073568.3

Uniqueid2: 1459073575.4

CallerID1: 201

CallerID2: 301

Spuštění hovoru po interakci s Vtiger CRM systémem mělo zpoždění přibližně 2000ms. Click to Dial funkci jsem otestoval na pár hovorech se stejným výsledkem.

Další testování proběhlo několika hovory na support infolinku fiktivní společnosti. Během testování jsem přihlašoval a odhlašoval operátory a manažera v CRM systému pro otestování správného přesměrování hovoru díky dotazům nad databází. Pro toto testování jsem vytvořil další dva SIP klienty a to neznámého zákazníka a již registrovaného zákazníka.

Testování již registrovaného zákazníka proběhlo ve dvou variantách. V první variantě byl zákazníkům přiřazený operátor přihlášen v CRM systému a hovor by měl být přesměrován na jeho CRM extension. V druhé byl naopak odhlášen a hovor by měl být přesměrován na jeho mobil. V mém testování registrovaného zákazníka byl vytvořen nový SIP klient 721 registrovaného zákazníka Marii Hrdinové. Marie byla zákaznicí operátora Karla Dráta, jehož CRM extension bylo 201 a číslo telefonní mobilu, samozřejmě také SIP klient, bylo 701. Nejprve jsem tedy přihlásil uživatele Karla Dráta do CRM systému a poté vytočil support infolinku 800 ze SIP účtu Marii Hrdinové. Hovor byl úspěšně proveden a přesměrován dle ODBC funkcí za pomoci SQL dotazů na extension 201. Výpis Asterisku CLI i s popisem, viz níže.

```
-- Executing [800@customer:2] Goto("SIP/721-00000052",  
"check_customer,1") in new stack  
    -- Goto (customer,check_customer,1)  
    -- Executing [check_customer@customer:1] NoOp("SIP/721-  
00000052", "") in new stack  
    -- Executing [check_customer@customer:2] Set("SIP/721-00000052",  
"CUST_ID=4") in new stack
```

Číslo zákazníka Marie bylo nalezeno v databázi zákazníků pod ID číslem čtyři a v následujícím kroku se dialplán skript dotazoval na přiřazeného uživatele.

```
-- Executing [check_customer@customer:3] GotoIf("SIP/721-  
00000052", "0?check_users,1") in new stack  
    -- Executing [check_customer@customer:4] Set("SIP/721-00000052",  
"USER_ID=5") in new stack  
    -- Executing [check_customer@customer:5] Set("SIP/721-00000052",  
"USER_NAME=Karel Drat") in new stack
```

Po nalezení ID uživatele a následně jeho uživatelského jména, skript dále zjišťoval status uživatele.

```
-- Executing [check_customer@customer:6] Set("SIP/721-00000052",  
"STATUS=Signed in") in new stack
```

Podle statusu byl tedy uživatel přihlášen, tudíž se skript přeměroval na úsek o zjištění CRM extension čísla.

```
-- Executing [check_customer@customer:7] GotoIf("SIP/721-00000052", "1?get_phone,1:get_mobile,1") in new stack
-- Goto (customer,get_phone,1)
-- Executing [get_phone@customer:1] NoOp("SIP/721-00000052", "") in new stack
-- Executing [get_phone@customer:2] ODBCFinish("SIP/721-00000052", "5") in new stack
-- Executing [get_phone@customer:3] Set("SIP/721-00000052", "EXT=201") in new stack
```

Díky nalezenému číslu uživatele se skript v této fázi přeměroval na úsek s odpovídajícím extension, tedy na agi skript pro veškerá čísla, který umožní vytočení daného čísla a nahrávání hovoru.

```
-- Executing [get_phone@customer:4] Goto("SIP/721-00000052", "incoming_calls,201,1") in new stack
-- Goto (incoming_calls,201,1)
-- Executing [201@incoming_calls:1] AGI("SIP/721-00000052", "agi://158.196.244.250/incoming.agi") in new stack
== Manager 'vtiger' logged on from 127.0.0.1
-- AGI Script Executing Application: (Monitor) Options: (wav,/VtigerAsteriskConnector/records//045f731038f545a4a6f1ae8443211ba8,m)
-- AGI Script Executing Application: (Dial) Options: (SIP/201,60)
-- Called SIP/201
-- SIP/201-00000010 is ringing
-- SIP/201-00000010 answered SIP/721-00000052
> 0x7f03b8060bb0 -- Probation passed - setting RTP source address to 158.196.194.151:8000
== Manager 'vtiger' logged off from 127.0.0.1
-- <SIP/721-00000052>AGI Script agi://158.196.244.250/incoming.agi completed, returning 0
-- Executing [201@incoming_calls:2] Hangup("SIP/721-00000052", "") in new stack
== Spawn extension (incoming_calls, 201, 2) exited non-zero on 'SIP/721-00000052'
```

V případě odhlášení uživatele Karla Dráta ze CRM systému byl hovor přeměrován na jeho mobilní telefon, viz výpis Asterisku CLI níže.

```
-- Executing [check_customer@customer:5] Set("SIP/721-00000056",
"USER_NAME=Karel Drat") in new stack
    -- Executing [check_customer@customer:6] Set("SIP/721-00000056",
"STATUS=Signed off") in new stack
        -- Executing [check_customer@customer:7] GotoIf("SIP/721-
00000056", "0?get_phone,1:get_mobile,1") in new stack
            -- Goto (customer,get_mobile,1)
                -- Executing [get_mobile@customer:1] NoOp("SIP/721-00000056",
"") in new stack
                    -- Executing [get_mobile@customer:2] Set("SIP/721-00000056",
"EXT=701") in new stack
```

Další testování proběhlo s neregistrovaným zákazníkem. Neznámému potenciálnímu zákazníkovi jsem registroval SIP klienta 500, který nefiguroval nikde v záznamech CRM systému. Testování proběhlo ve třech variantách. V první variantě byl jeden z operátorů online, konkrétně Pavel Linka se CRM extension 202, a druhý z operátorů byl odhlášen. Dialplán tedy procházel jednotlivé operátory a jakmile byl nalezen online operátor, přeměroval se hovor na něj. Tudíž jsem vytočil support infolinku 800 s neznámým potenciálním zákazníkem se SIP klientem 500 a hovor byl úspěšně přeměrován na extension 202 online operátora Pavla Linku, viz výpis Asterisk CLI s popisem níže.

```
-- Executing [800@customer:2] Goto("SIP/500-0000005c",
"check_customer,1") in new stack
    -- Goto (customer,check_customer,1)
        -- Executing [check_customer@customer:1] NoOp("SIP/500-
0000005c", " ") in new stack
            > Found no rows [SELECT contactid FROM vtiger_contactdetails
WHERE mobile = '500']
```

Volající s číslem 500 nebyl nalezen v databázi Vtiger CRM jako registrovaný zákazník, proto se skript přeměroval na úsek check_users pro zjištění online operátora.

```
-- Executing [check_users@customer:6] Set("SIP/500-0000005c",
"USER_NAME=Karel Drat") in new stack
    -- Executing [check_users@customer:7] Set("SIP/500-0000005c",
"STATUS=Signed off") in new stack
        -- Executing [check_users@customer:8] GotoIf("SIP/500-0000005c",
"0?get_phone,1:loop") in new stack
            -- Goto (customer,check_users,3)
```

Skript tedy prošel prvního uživatele a zjistil, že uživatel Karel Drát je offline. Skript se vrátil zpět na začátek smyčky a prověřil status dalšího uživatele.

Testování a analýza hovorů

```
-- Executing [check_users@customer:6] Set ("SIP/500-0000005c",
"USER_NAME=Pavel Linka") in new stack
-- Executing [check_users@customer:7] Set ("SIP/500-0000005c",
"STATUS=Signed in") in new stack
-- Executing [check_users@customer:8] GotoIf("SIP/500-0000005c",
"1?get_phone,1:loop") in new stack
-- Goto (customer,get_phone,1)
```

Skript již našel uživatele, který je online, a tím se skript přesměroval na úsek získání čísla daného uživatele. Dále pak pokračoval až k agi skriptu a zahájení hovoru.

Následující testovací varianta proběhla s odhlášenými operátory a přihlášeným manažerem. Neznámý potenciální zákazník tedy vytočil support infolinku 800 a hovor byl úspěšně přesměrován na CRM extension 301 manažera Jana Hlavu, viz výpis Asterisk CLI níže.

```
-- Executing [check_users@customer:5] GotoIf("SIP/500-0000005f",
"1?call_manager,1") in new stack
-- Goto (customer,call_manager,1)
-- Executing [call_manager@customer:1] NoOp("SIP/500-0000005f",
"") in new stack
-- Executing [call_manager@customer:2] Set ("SIP/500-0000005f",
"USER_ID=7") in new stack
-- Executing [call_manager@customer:3] Set ("SIP/500-0000005f",
"USER_NAME=Jan Hlava") in new stack
-- Executing [call_manager@customer:4] Set ("SIP/500-0000005f",
"STATUS=Signed in") in new stack
-- Executing [call_manager@customer:5] GotoIf("SIP/500-
0000005f", "1?get_phone,1:get_mobile,1") in new stack
```

Poslední testovací varianta proběhla jak s odhlášenými operátory, tak s odhlášeným manažerem. Neznámý potenciální zákazník s extension 500 tedy vytočil support infolinku 800 a hovor byl přesměrován na mobilní telefon manažera Jana Hlavy s extension 605. Kompletní výpis z Asterisk CLI posledního testovacího hovoru v příloze.

Veškeré SQL dotazy ODBC funkcí a tím i integrita Asterisku s relační databází Vtiger CRM systému fungovala výborně. Zpoždění během přesměrování hovorů a provedení veškerých SQL dotazů po vytočení support infolinky bylo nepatrné přibližně 500ms.

Při každém příchozím hovoru na CRM extension přihlášeného uživatele se zobrazilo notifikační okno Incoming Call s údaji o volajícím. Úspěšně se notifikační okno zobrazilo při každém hovoru a se všemi doplňujícími informacemi získanými díky modifikaci tohoto okna. Jméno volajícího je interaktivní a po kliknutí se ve Vtiger CRM zobrazí karta s údaji o volajícím, viz obrázek 4.1.

Testování a analýza hovorů

The screenshot shows the Vtiger CRM interface. At the top, there are navigation tabs: Contacts, Opportunities, Products, Documents, Tickets, and All. Below the navigation is a search bar with the text 'Type keyword and press enter' and a search icon. The main content area is divided into several sections. On the left, there is a contact profile for Marie Hrdinova, Trading s.r.o., with fields for First Name, Last Name, Office Phone, Organization Name, Title, Primary Email, Assigned To, and Mailing City. In the center, there is a section for Activities and Updates. The Activities section shows two items: 'Meeting - Sch?ze' (Planned) and 'To Do - Info mail' (In Progress). The Updates section shows two items: 'Calendar added Info mail' (8 minutes ago) and 'Calendar added Sch?ze' (9 minutes ago). On the right, there is a 'Contact Summary' sidebar with links for Contact Details, Comments, Updates, Opportunities, Activities, Emails, Tickets, Quotes, and Purchase Order. A notification window is overlaid on the top right, displaying 'Incoming Call' details for Marie Hrdinova.

Obrázek 4.1: *Vtiger CRM s notifikací o příchozím hovoru*

V příchozích hovorem neregistrovaného zákazníka se také úspěšně zobrazilo notifikační okno s doplňujícími informacemi, viz obrázek 4.2.

The screenshot shows a notification window for an incoming call. The notification text is: 'Incoming Call', 'Call From : 500', 'Phone : 500', 'Email : undefined', and 'Organization : undefined'. Below the notification, there is a form with an 'Enter Email-id' input field, a 'Select' dropdown menu, and a 'Save' button.

Obrázek 4.2: *Notifikační okno při příchodu neznámého hovoru ve Vtiger CRM*


Součástí modulu PBXManager ve Vtiger CRM je výpis proběhlých hovorů, takzvaný Recors List. V tomto výpisu lze najít všechny hovory na Phone Extension číslo daného uživatele. Výpis obsahuje atributy jako status hovoru, zdali byl hovor úspěšně dokončen či zda nebyl vyzvednut v nepřítomnosti operátora apod. Kromě informací, jako jsou číslo a jméno zákazníka nebo čas délky hovoru, je zde atribut Recording. Atribut Recording umožní stáhnout nahrávku hovoru z Vtiger Asterisk Connectoru. Veškeré nahrávky jsou totiž uloženy v adresáři Vtiger Asterisk Connectoru na Asterisk serveru a právě Recording odkaz umožňuje zjednodušený přístup k těmto nahrávkám.

Testování a analýza hovorů

<input type="checkbox"/>	Call Status	Customer Number	Customer	User	Recording	Duration (sec)	Start Time
<input type="checkbox"/>	↑ ringing		Jan Novak	Jan Hlava		0	04-09-2016 06:44 AM
<input type="checkbox"/>	↑ ringing		Jan Novak	Jan Hlava		0	04-09-2016 06:43 AM
<input type="checkbox"/>	↑ ringing		Jan Novak	Jan Hlava		0	04-09-2016 06:43 AM
<input type="checkbox"/>	↑ ringing		Jan Novak	Jan Hlava		0	04-09-2016 06:39 AM
<input type="checkbox"/>	↑ ringing		Jan Novak	Jan Hlava		0	04-09-2016 06:37 AM
<input type="checkbox"/>	↑ ringing		Jan Novak	Jan Hlava		0	04-09-2016 06:33 AM
<input type="checkbox"/>	↓ busy	721	Marie Hrdinova	--		60	04-09-2016 06:31 AM
<input type="checkbox"/>	↓ busy	721	Marie Hrdinova	--		60	04-09-2016 06:29 AM
<input type="checkbox"/>	↓ completed	721	Marie Hrdinova	Jan Hlava	🔊	42	04-09-2016 08:25 AM
<input type="checkbox"/>	↓ busy	721	Marie Hrdinova	--		60	04-02-2016 08:04 AM

Obrázek 4.3: Ukázka Record List ve Vtiger CRM

U jednotlivých záznamů hovorů lze pak zobrazit detailní informace. Detail hovoru poskytuje nejrozsáhlejší informace o daném hovoru, například délka trvání hovoru a délka účtování hovoru od počátku vyzvednutí hovoru v sekci Bill Duration. Ukázka detailu hovoru Marii Hrdinové na uživatele Jana Hlavu, viz níže na obrázku 4.4.


 Call From Marie Hrdinova
Mobile: 721 More ▾

▼ Call Details

Direction	inbound	Call Status	completed
Customer	Marie Hrdinova	User	Jan Hlava
Customer Number	721	Customer Type	Contacts
Start Time	04-09-2016 08:25 AM	End Time	04-09-2016 08:26 AM
Recording	http://158.196.244.250:8090/recording?id...	Duration (sec)	42
Bill Duration (sec)	18	Source UUID	93c1f1ef14f9423892dfd4da49b9a028
Gateway	PBXManager	Assigned To	Jan Hlava
Created Time	04-09-2016 06:25 AM	Modified Time	04-09-2016 06:25 AM

Obrázek 4.4: Detail hovoru v záznamu Records List

Dalším zajímavým prvkem Vtiger CRM systému je možnost reportování analýz a grafů pomocí dat z PBXManageru. Lze tedy analyzovat a vytvářet nejrozsáhlejší grafy pomocí dat z proběhlých hovorů z PBXManager modulu. Pro ukádku jsem vytvořil koláčový graf proběhlých hovorů, který se skládá z veškerých hovorů rozdělených do tří statusů - completed, busy a ringing. Tyto tři statusy jsou myšleny stavem hovoru pro kompletní hovor, nevyzvednutý hovor v nepřítomnosti přihlášeného uživatele a vyzváněcí hovor nepřijatý přihlášeným uživatelem. Ukázky vytvoření grafu a samotný graf lze vidět na obrázku 4.5 a obrázku 4.6.

Proběhlé hovory Customize 

Select Groupby Field* Select Data Fields*

Call Status Record Count

Modify Conditions ▼

All Conditions (All conditions must be met)

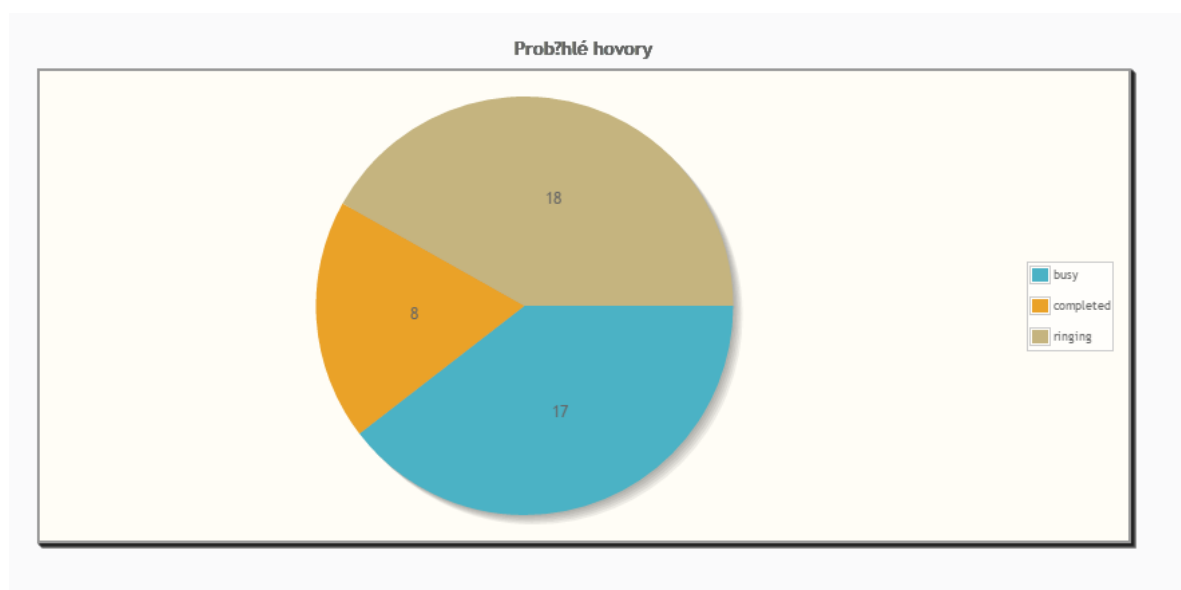
Add Condition

Any Conditions (At least one of the conditions must be met)

(PBXManager) Call Status ▼	equals ▼	busy
(PBXManager) Call Status ▼	equals ▼	completed
(PBXManager) Call Status ▼	equals ▼	ringing

Add Condition

Obrázek 4.5: Vytváření reportu Proběhlé hovory



Obrázek 4.6: Graf reportu Proběhlé hovory

Kromě testování hovorů jsem si ověřil i zapisování logů a nahrávání hovorů na straně Vtiger Asterisk Connectoru. Hovory se úspěšně nahrávaly při každém testovacím hovoru přes aplikaci Monitor spuštěnou AGI příkazem. Hovory se ukládaly do mnou zvoleného adresáře */VtigerAsteriskConnector/records/*.

Logy Asterisk Vtiger Connector aplikace se také úspěšně ukládali při testovaných hovorech a to do adresáře */VtigerAsteriskConnector/logs/*. V tomto adresáři je ukládáno hned několik log souborů. Logy informující o AGI skriptu - *AgiError.log* a *AgiInfo.log*, logy o samotné aplikaci konektoru - *WebappError.log* a *WebappInfo.log* a nakonec logy hovorů jednotlivých dnů - rozděleny

na agi a webapp ve formátu nohup.agi.yyyymmdd.out a nohup.webapp.yyyymmdd.out. Veškeré logy se trvale zaznamenávají od prvotního spuštění aplikace.

Závěr

Účelem této diplomové práce bylo integrovat pobočkovou ústřednu Asterisk s Vtiger CRM systémem a jeho databází. Navrhnout a implementovat dialplán na support linku fiktivního podniku za pomoci získaných dat z databáze. Modifikovat notifikační okno příchozího hovoru o přidaná data z databáze. Součástí bylo také testovat a analyzovat hovory, Asterisk Vtiger Connector aplikaci a PBXManager modul ve Vtiger CRM.

V teoretické části byly nejprve rozebrány CRM systémy a porovnány tři vybrané open-source verze CRM systémů a jejich možnosti pro integraci s PBX Asterisk. Dále byl popsán Asterisk a jeho rozhraní pro komunikaci s externími aplikacemi a to převážně rozhraní AMI a AGI, které se vyskytují i v praktické části této práce.

V praktické části práce jsem implementoval a konfiguroval pobočkovou ústřednu Asterisk, Vtiger CRM, PBXManager modul a Asterisk Vtiger Connector pro správnou integraci Asterisku s Vtiger CRM. Pobočková ústředna figurovala hned s několika SIP klienty, kteří zobrazovali účastníky testovacích hovorů, a to jak zákazníky, tak uživatele CRM systému, tedy operátory support infolinky.

V další části jsem navrhnul a implementoval dialplán průběhu hovoru na support infolinku fiktivního podniku. Zde byl navržen diagram průběhu hovoru, proběhla explorace datového modelu databáze CRM systému a implementace a konfigurace ODBC konektoru pro integraci Asterisku s relační databází Vtiger CRM systému. Součástí implementace dialplánu byla také implementace SQL dotazů nad databází CRM systému pro získání správných dat pro přesměrování hovorů.

V poslední části praktické části bylo modifikováno notifikační okno příchozího hovoru ve Vtiger CRM. V notifikačním okně byla přidána data o volajícím pomocí získaná pomocí PHP a zobrazena díky technologiím jQuery a AJAX.

Na závěr práce proběhlo testování hovorů. Analýza Vtiger Asterisk Connectoru a s ním spojenými rozhraní AMI a AGI. Dále testování Vtiger CRM a jeho modulu PBXManager.

Podarilo se mi integrovat Asterisk s Vtiger CRM systémem a jeho relační databází. Dále zobrazovat data o volajícím ve Vtiger CRM. Veškeré hovory proběhly úspěšně se správným přesměrováním na daného uživatele dle informací z databáze. Nyní jsou mé zkušenosti z oblasti CRM systémů daleko větší, spolu s dalšími zkušenostmi z PBX Asterisk a integrací s relačními databázemi. Znalosti nabyté prostudováním této diplomové práce umožní aplikovat integraci Asterisku s informačním systémem v reálném podniku.

Použitá literatura

- [1] PETERSEN, Rob. 21 experts define CRM in their own words and pictures. Barn Raisers [online]. 2012 [cit. 2016-04-24]. Dostupné z: <http://barnraisersllc.com/2012/06/21-experts-define-crm-words-pictures/>
- [2] GÁLA, Libor, Jan POUR a Zuzana ŠEDIVÁ. Podniková informatika. 2., přeprac. a aktualiz. vyd. Praha: Grada, 2009. Expert (Grada). ISBN 978-80-247-2615-1
- [3] IVEY, Jay. CRM Software UserView. Software Advice [online]. 2014 [cit. 2016-04-24]. Dostupné z: <http://www.softwareadvice.com/crm/userview/report-2014/>
- [4] ŠLAPÁK, Ondřej. Různá pojetí architektury informačních systémů. [online]. [cit. 2016-04-24]. Dostupné z: <http://www.slapak.cz/ondrej/archIS.htm>
- [5] DOHNAL, Jan a Miroslav KUČERA. Úvod do CRM v informační společnosti. Vyd. 1. Praha: Vysoká škola ekonomická, Fakulta informatiky a statistiky, 2000. ISBN 80-245-0139-2
- [6] PODNIKOVÉ INFORMAČNÍ SYSTÉMY – CRM [online]. 2011 [cit. 2016-04-24]. Dostupné z: http://homel.vsb.cz/~dan11/is_skripta/IS%202011%20-%20CRM.pdf. Vedoucí práce Ing. Roman Danel, Ph.D.
- [7] VOŘÍŠEK, Jiří a Josef BASL. Principy a modely řízení podnikové informatiky. Vyd. 1. V Praze: Oeconomica, 2008. ISBN 978-80-245-1440-6
- [8] Vtiger CRM [online]. 2016 [cit. 2016-04-24]. Dostupné z: <https://www.vtiger.com>
- [9] Asterisk Integration. In: Wiki Vtiger CRM [online]. 2016 [cit. 2016-04-24]. Dostupné z: https://wiki.vtiger.com/vtiger6/index.php/Asterisk_Integration#Vtiger_Asterisk_Connector
- [10] Understanding Zoho CRM. In: Zoho CRM [online]. 2016 [cit. 2016-04-24]. Dostupné z: <https://www.zoho.com/crm/help/understanding-zohocrm.html>
- [11] Zoho PhoneBridge. In: Zoho CRM [online]. 2016 [cit. 2016-04-24]. Dostupné z: <https://www.zoho.com/crm/help/understanding-zohocrm.html>
- [12] SuiteCRM [online]. 2016 [cit. 2016-04-24]. Dostupné z: <https://suitecrm.com/>
- [13] User Manual YAI. In: GitHub, Inc. [online]. 2013 [cit. 2016-04-24]. Dostupné z: <https://github.com/AlertusTechnologiesLLC/yai/wiki/User-Manual>
- [14] SuiteCRM. In: Source Force [online]. 2016 [cit. 2016-04-24]. Dostupné z: <https://sourceforge.net/projects/suitecrm>
- [15] The Pros And Cons Of Open-Source CRM. In: CRM Switch [online]. 2014 [cit. 2016-04-24]. Dostupné z: <https://www.crmswitch.com/crm-value/open-source-crm-pros-and-cons/>
- [16] VOZŇÁK, Miroslav. Voice over IP. 1. vyd. Ostrava: VŠB - Technická univerzita Ostrava, 2008, 176 s. ISBN 978-80-248-1828-3
- [17] BRYANT, Russell. Asterisk: the definitive guide. Fourth edition. Sebastopol: O'Reilly, 2013. ISBN 9781449332426

- [18] Asterisk manager API. In: Voip-info [online]. 2014 [cit. 2016-04-24]. Dostupné z: <http://www.voip-info.org/wiki/view/Asterisk+manager+API>
- [19] AMI Libraries and Frameworks. In: Wiki Asterisk [online]. 2016 [cit. 2016-04-24]. Dostupné z: <https://wiki.asterisk.org/wiki/display/AST/AMI+Libraries+and+Frameworks>
- [20] Asterisk REST Interface (ARI). In: Wiki Asterisk [online]. 2015 [cit. 2016-04-25]. Dostupné z: <https://wiki.asterisk.org/wiki/pages/viewpage.action?pageId=29395573>
- [21] Vtiger CRM 6 Installation. In: Wiki Vtiger [online]. 2013 [cit. 2016-04-25]. Dostupné z: https://wiki.vtiger.com/index.php/Vtiger_CRM_6_Installation

Seznam příloh

Příloha A:	Konfigurační soubor extensions.conf.....	I
Příloha B:	Konfigurační soubor sip.conf.....	III
Příloha C:	Konfigurační soubor func_odbc.conf.....	V
Příloha D:	Soubor pro dotazování do databáze dbqueries.php	VI
Příloha E:	Soubor pro zobrazení notifikačního okna PBXManagerJS.js	VII
Příloha F:	Výpis hovoru neznámého zákazníka na manažera.....	XVI

Součástí diplomové práce je CD.

Adresářová struktura přiloženého CD:

- DP.pdf
- /soubory/extensions.conf
- /soubory/sip.conf
- /soubory/func_odbc.conf
- /soubory/dbqueries.php
- /soubory/PBXManagerJS.js
- /soubory/vypisHovoru.txt

Konfigurační soubor extensions.conf

Příloha A: *Konfigurační soubor extensions.conf*

```
[vtiger_outbound]
```

```
exten => _X.,1,Agi(agi://158.196.244.250/incoming.agi)
```

```
same => n, Hangup()
```

```
[incoming_calls]
```

```
exten => 800,1, Answer()
```

```
same => n, Goto(check_customer,1)
```

```
exten => check_customer,1, NoOp()
```

```
same => n, Set(CUST_ID=${GET_CUSTOMER(${CALLERID(num)})})
```

```
same => n, GotoIf(${ODBCROWS} < 1)?check_users,1
```

```
same => n, Set(USER_ID=${GET_OWNER(${CUST_ID})})
```

```
same => n, Set(USER_NAME=${GET_NAME(${USER_ID})})
```

```
same => n, Set(STATUS=${CHECK_STATUS(${USER_NAME})})
```

```
same => n, GotoIf("${STATUS}" = "Signed  
In"?get_phone,1:get_mobile,1)
```

```
exten => check_users,1, NoOp()
```

```
same => n, Set(ID=${GET_OPERATOR()})
```

```
same => n(loop), NoOp()
```

```
same => n, Set(USER_ID=${ODBC_FETCH(${ID})})
```

```
same => n, GotoIf(${ODBCROWS} < 1)?call_manager,1
```

```
same => n, Set(USER_NAME=${GET_NAME(${USER_ID})})
```

```
same => n, Set(STATUS=${CHECK_STATUS(${USER_NAME})})
```

```
same => n, GotoIf("${STATUS}" = "Signed In"?get_phone,1:loop)
```

```
exten => get_phone,1, NoOp()
```

```
same => n, ODBCFinish(${USER_ID})
```

```
same => n, Set(EXT=${GET_EXTENSION(phone_crm_extension,${USER_ID})})
```

```
same => n, Goto(incoming_calls,${EXT},1)
```

```
exten => call_manager,1, NoOp()
```

Konfigurační soubor extensions.conf

```
same => n,Set(USER_ID=${GET_MANAGER()})
same => n,Set(USER_NAME=${GET_NAME(${ID})})
same => n,Set(STATUS=${CHECK_STATUS(${USER_NAME})})
same => n,GotoIf("${STATUS}" = "Signed In"?get_phone,1)
same => n,Goto(get_mobile,1)

exten => get_mobile,1,NoOp()
same => n,Set(EXT=${GET_EXTENSION(phone_mobile,${USER_ID})})
same => n,Goto(incoming_calls,${EXT},1)

exten => _X.,1,Agi(agi://158.196.244.250/incoming.agi)
same => n,Hangup()
```

Konfigurační soubor sip.conf

Příloha B: *Konfigurační soubor sip.conf*

```
[general]
binaddr=158.196.244.250
context=incoming_calls
allowguest=yes
transport=udp

[vtiger] (!)
host=dynamic
type=friend
qualify=yes
bindport=5038
disallow=all
allow=alaw

[201] (vtiger)
secret=karelcrm
context=vtiger_outbound

[202] (vtiger)
secret=pavelcrm
context=vtiger_outbound

[301] (vtiger)
secret=jancrm
context=vtiger_outbound

[701] (vtiger)
secret=karelmobil
context=vtiger_outbound

[702] (vtiger)
secret=pavelmobil
```

Konfigurační soubor sip.conf

context=vtiger_outbound

[605] (vtiger)

secret=janmobil

context=vtiger_outbound

[721] (vtiger)

secret=marie

context=incoming_calls

[500] (vtiger)

secret=unknown

context=incoming_calls

Konfigurační soubor func_odbc.conf

Příloha C: *Konfigurační soubor func_odbc.conf*

[OWNER]

prefix=GET

dsn=vtigercrm

readsql=SELECT smownerid FROM vtiger_crmentity WHERE crmid='\${ARG1}'

[OPERATOR]

prefix=GET

dsn=vtigercrm

mode=multirow

readsql=SELECT userid FROM vtiger_user2role WHERE roleid='H5'

[MANAGER]

prefix=GET

dsn=vtigercrm

readsql=SELECT userid FROM vtiger_user2role WHERE roleid='H4'

[NAME]

prefix=GET

dsn=vtigercrm

readsql=SELECT user_name FROM vtiger_users WHERE id='\${ARG1}'

[STATUS]

prefix=CHECK

dsn=vtigercrm

readsql=SELECT status FROM vtiger_loginhistory WHERE
user_name='\${ARG1}'

readsql+= ORDER BY login_id DESC

[EXTENSION]

prefix=GET

dsn=vtigercrm

readsql=SELECT \${ARG1} FROM vtiger_users WHERE id='\${ARG2}'

Příloha D: *Soubor pro dotazování do databáze dbqueries.php*

```
<?php
$servername = "localhost";
$username = "asterisk";
$password = "uniNetIsSafe";
$databasename = "vtigercrm";
$db = new mysqli($hostname, $username, $password, $databasename);
$data = array();
$recordid = $_GET["num"];
$result = $db->query("SELECT email FROM vtiger_contactdetails WHERE
mobile = ". $recordid .";");
if ($result->num_rows > 0) {
    while($row = $result->fetch_assoc()) {
        array_push($data, $row["email"]);
    }
} else array_push($data, "undefined");
$result = $db->query("SELECT accountname FROM vtiger_account INNER
JOIN vtiger_contactdetails ON
vtiger_account.accountid=vtiger_contactdetails.accountid WHERE
vtiger_contactdetails.mobile = ". $recordid .";");
if ($result->num_rows > 0) {
    while($row = $result->fetch_assoc()) {
        array_push($data, $row["accountname"]);
    }
} else array_push($data, "undefined");
echo json_encode($data);
mysqli->close();
?>
```

Příloha E: *Soubro pro zobrazení notificačního okna PBXManagerJS.js*

```
var Vtiger_PBXManager_Js = {
    /**
     * Function registers PBX for popups
     */
    registerPBXCall : function() {
        Vtiger_PBXManager_Js.requestPBXgetCalls();
    },

    /**
     * Function registers PBX for Outbound Call
     */
    registerPBXOutboundCall : function(number,record) {
        Vtiger_PBXManager_Js.makeOutboundCall(number,record);
    },

    /**
     * Function request for PBX popups
     */
    requestPBXgetCalls : function() {
        var url =
        'index.php?module=PBXManager&action=IncomingCallPoll&mode=searchIncomingCalls';

        AppConnector.request(url).then(function(data) {
            if(data.success && data.result) {
                for(i=0; i< data.result.length; i++) {
                    var record = data.result[i];

                    if(jQuery('#pbxcall_'+record.pbxmanagerid+'').size()== 0 )    {
                        jQuery.ajax({
                            type: "GET", // HTTP method POST or GEST
                            url:
                            "/modules/PBXManager/resources/dbqueries.php?num="+record.customernumber, //Where to make Ajax calls
                            success:function(response) {
```



```
Vtiger_PBXManager_Js.showPBXIncomingCallPopup(record, response);
    }
    });
}
else

Vtiger_PBXManager_Js.updatePBXIncomingCallPopup(record);
    }
}
});
Vtiger_PBXManager_Js.removeCompletedCallPopup();
},

/**
 * Function display the PBX popup
 */

showPBXIncomingCallPopup : function(record, response) {

    var data = JSON.parse(response)

    var params = {
        title: app.vtranslate('JS_PBX_INCOMING_CALL'),
        text: '<div class="row-fluid pbxcall"
id="pbxcall_'+record.pbxmanagerid+'" callid='+record.pbxmanagerid+'
style="color:black"><span class="span12" id="caller"
value="'+record.customernumber+'">'+app.vtranslate('JS_PBX_CALL_FROM
')+ ' : '+record.customernumber+'</span> <span>Email :
'+data[0]+'</span><span>Organization : '+data[1]+'</span><span
class="hide span12" id="contactsave_'+record.pbxmanagerid+'">\n\
        <span><input class="span3"
id="email_'+record.pbxmanagerid+'" type="text" placeholder="Enter
Email-id"></input>&nbsp;&nbsp;&nbsp;&nbsp;<select class="input-small"
id="module_'+record.pbxmanagerid+'"
```

Soubro pro zobrazení notificačního okna PBXManagerJS.js

```
placeholder="Select"><option>Select</option></select><h5
class="alert-danger hide span3"
id="alert_msg">'+app.vttranslate('JS_PBX_FILL_ALL_FIELDS')+'</h5>\n\
        <button class="btn btn-success pull-right"
id="pbxcontactsave_'+record.pbxmanagerid+'"
recordid="'+record.pbxmanagerid+'" type="submit">Save</button>\n\
        </span></span><br/><span class="span12"
style="display:none" id="answeredby"><i class="icon-
headphones"></i>&nbsp;<span
id="answeredbyname"></span></span></div>',
        width: '28%',
        min_height: '75px',
        addclass:'vtCall',
        icon: 'vtCall-icon',
        hide:false,
        closer:true,
        type:'info',
        after_open:function(p) {
            jQuery(p).data('info', record);
        }
    };
    Vtiger_Helper_Js.showPnotify(params);

    //To remove the popup for all users except answeredby
    (existing record)
    if(record.user) {
        if(record.user != record.current_user_id) {
Vtiger_PBXManager_Js.removeCallPopup(record.pbxmanagerid);
        }
    }

    // To check if it is new or existing contact

Vtiger_PBXManager_Js.checkIfRelatedModuleRecordExist(record);
```

```
        if(record.answeredby!=null) {

jQuery('#answeredbyname','#pbxcall_'+record.pbxmanagerid+'.text(re
cord.answeredby);

jQuery('#answeredby','#pbxcall_'+record.pbxmanagerid+'.show();
        }

jQuery('#pbxcontactsave_'+record.pbxmanagerid+'.bind('click',
function(e) {

        var pbxmanagerid =
jQuery(e.currentTarget).attr('recordid');

        if(jQuery('#module_'+pbxmanagerid+'.val() ==
'Select'){

                jQuery('#alert_msg').show();
                return false;

        }

        if(jQuery('#email_'+pbxmanagerid+'.val() == ""){
                jQuery('#alert_msg').show();
                return false;

        }

        Vtiger_PBXManager_Js.createRecord(e, record);
        //To restrict the save button action to one click

jQuery('#pbxcontactsave_'+record.pbxmanagerid+'.unbind('click');
        });

},

createRecord: function(e, record) {
        var pbxmanagerid = jQuery(e.currentTarget).attr('recordid');
        var email = jQuery('#email_'+pbxmanagerid+'.val());
        var moduleName = jQuery('#module_'+pbxmanagerid+'.val());
```

```
        var number =
jQuery('#caller', '#pbxcall_'+pbxmanagerid+'').attr("value");

        var url =
'index.php?module=PBXManager&action=IncomingCallPoll&mode=createRecord&number='+encodeURIComponent(number)+'&email='+encodeURIComponent(email)+'&callid='+record.sourceuuid+'&modulename='+moduleName;

        AppConnector.request(url).then(function(data) {
            if(data.success && data.result) {
                jQuery('#contactsave_'+pbxmanagerid+'').hide();
            }
        });
    },

    checkIfRelatedModuleRecordExist: function(record) {
        switch(record.callername){
            case null:
                var url =
'index.php?module=PBXManager&action=IncomingCallPoll&mode=checkModuleViewPermission&view=EditView';

                AppConnector.request(url).then(function(data) {
                    var respondedata = JSON.parse(data);
                    var showSaveOption = false;
                    var moduleList = respondedata.result.modules;
                    var contents =
jQuery('#module_'+record.pbxmanagerid+'');
                    var newEle;
                    for(var module in moduleList){
                        if(moduleList.hasOwnProperty(module)) {
                            if(moduleList[module]){
                                newEle = '<option
id="select_'+module+'
value="'+module+'"'>'+app.vtranslate(module)+'</option>';
                                contents.append(newEle);
                                showSaveOption = true;
                            }
                        }
                    }
                });
            }
        }
    }
};
```

```
        }
    }
    if(responedata.success && showSaveOption)

jQuery('#contactsave_'+record.pbxmanagerid+'').show();
    });
    break;
    default:

jQuery('#caller','#pbxcall_'+record.pbxmanagerid+'').html(app.vtranslate('JS_PBX_CALL_FROM')+ ' :&nbsp;<a
href="index.php?module='+record.customertype+'&view=Detail&record='+
record.customer+'">'+record.callername+'</a>');
        break;
    }
},

/**
 * Function to update the popup with answeredby, hide
contactsave option e.t.c.,
 */
updatePBXIncomingCallPopup: function(record) {
    if(record.answeredby!=null) {

jQuery('#answeredbyname','#pbxcall_'+record.pbxmanagerid+'').text(re
cord.answeredby);

jQuery('#answeredby','#pbxcall_'+record.pbxmanagerid+'').show();
    }
    if(record.customer!=null && record.customer!=''){

jQuery('#caller','#pbxcall_'+record.pbxmanagerid+'').html(app.vtrans
late('JS_PBX_CALL_FROM')+ ' :&nbsp;<a
href="index.php?module='+record.customertype+'&view=Detail&record='+
record.customer+'">'+record.callername+'</a>');
        jQuery('#contactsave_'+record.pbxmanagerid+'').hide();
    }
}
```

```
        //To remove the popup for all users except answeredby (new
record)
        if(record.user) {
            if(record.user != record.current_user_id) {

Vtiger_PBXManager_Js.removeCallPopup(record.pbxmanagerid);
            }
        }
    },

    /**
     * Function to remove the call popup which is completed
     */
    removeCompletedCallPopup:function() {
        var callid = null;
        var pbxcall = jQuery('.pbxcall');
        for(var i=0; i<pbxcall.length;i++){
            callid = pbxcall[i].getAttribute('callid');
            var url =
'index.php?module=PBXManager&action=IncomingCallPoll&mode=getCallSta
tus&callid='+encodeURIComponent(callid)+'';
            AppConnector.request(url).then(function(data) {
                if(data.result) {
                    if(data.result!='in-progress' &&
data.result!='ringing') {

Vtiger_PBXManager_Js.removeCallPopup(callid);
                    }
                }
            });
        }
    },

    /**
     * Function to remove call popup
```

```
    */
    removeCallPopup: function(callid) {

jQuery('#pbxcall_'+callid+'.parent().parent().parent().remove();
    },

    /**
     * To get contents holder based on the view
     */
    getContextHolder:function(view){
        if(view == 'List')
            return jQuery('.listViewContentDiv');
        else
            return jQuery('.detailViewContainer');
    },

    /**
     * Function to forward call to number
     */
    makeOutboundCall : function(number, record){
        var params = {
            'number' : number,
            'record' : record,
            'module' : 'PBXManager',
            'action' : 'OutgoingCall'
        }
        AppConnector.request(params).then(function(data) {
            if(data.result){
                params = {
                    'text' :
app.vtranslate('JS_PBX_OUTGOING_SUCCESS'),
                    'type' : 'info'
                }
            }
        })else{
```

```
        params = {
            'text' :
app.vtranslate('JS_PBX_OUTGOING_FAILURE'),
            'type' : 'error'
        }
    }
    Vtiger_Helper_Js.showPnotify(params);
});
},

/**
 * Function to register required events
 */
registerEvents : function(){
    var thisInstance = this;
    //for polling
    var url =
'index.php?module=PBXManager&action=IncomingCallPoll&mode=checkPermi
ssionForPolling';
    AppConnector.request(url).then(function(data){
        if(data.result) {
            Vtiger_PBXManager_Js.registerPBXCall();

setInterval("Vtiger_PBXManager_Js.registerPBXCall()", 3000);
        }
    });
}

//On Page Load
jQuery(document).ready(function() {
    Vtiger_PBXManager_Js.registerEvents();
});
```


Výpis hovoru neznámého zákazníka na manažera

Příloha F: *Výpis hovoru neznámého zákazníka na manažera*

```
-- Executing [800@customer:2] Goto("SIP/500-0000006e",
"check_customer,1") in new stack
-- Goto (customer,check_customer,1)
-- Executing [check_customer@customer:1] NoOp("SIP/500-
0000006e", "") in new stack
> Found no rows [SELECT contactid FROM vtiger_contactdetails
WHERE mobile = '500']
-- Executing [check_customer@customer:2] Set("SIP/500-0000006e",
"CUST_ID=") in new stack
-- Executing [check_customer@customer:3] GotoIf("SIP/500-
0000006e", "1?check_users,1") in new stack
-- Goto (customer,check_users,1)
-- Executing [check_users@customer:1] NoOp("SIP/500-0000006e",
"") in new stack
-- Executing [check_users@customer:2] Set("SIP/500-0000006e",
"ID=16") in new stack
-- Executing [check_users@customer:3] NoOp("SIP/500-0000006e",
"") in new stack
-- Executing [check_users@customer:4] Set("SIP/500-0000006e",
"USER_ID=5") in new stack
-- Executing [check_users@customer:5] GotoIf("SIP/500-0000006e",
"0?call_manager,1") in new stack
-- Executing [check_users@customer:6] Set("SIP/500-0000006e",
"USER_NAME=Karel Drat") in new stack
-- Executing [check_users@customer:7] Set("SIP/500-0000006e",
"STATUS=Signed off") in new stack
-- Executing [check_users@customer:8] GotoIf("SIP/500-0000006e",
"0?get_phone,1:loop") in new stack
-- Goto (customer,check_users,3)
-- Executing [check_users@customer:3] NoOp("SIP/500-0000006e",
"") in new stack
-- Executing [check_users@customer:4] Set("SIP/500-0000006e",
"USER_ID=6") in new stack
-- Executing [check_users@customer:5] GotoIf("SIP/500-0000006e",
"0?call_manager,1") in new stack
-- Executing [check_users@customer:6] Set("SIP/500-0000006e",
"USER_NAME=Pavel Linka") in new stack
```

Výpis hovoru neznámého zákazníka na manažera

```
-- Executing [check_users@customer:7] Set("SIP/500-0000006e",
"STATUS=Signed off") in new stack
-- Executing [check_users@customer:8] GotoIf("SIP/500-0000006e",
"0?get_phone,1:loop") in new stack
-- Goto (customer,check_users,3)
-- Executing [check_users@customer:3] NoOp("SIP/500-0000006e",
"") in new stack
-- Executing [check_users@customer:4] Set("SIP/500-0000006e",
"USER_ID=") in new stack
-- Executing [check_users@customer:5] GotoIf("SIP/500-0000006e",
"0?call_manager,1") in new stack
> Found no rows [SELECT user_name FROM vtiger_users WHERE
id='']
-- Executing [check_users@customer:6] Set("SIP/500-0000006e",
"USER_NAME=") in new stack
> Found no rows [SELECT status FROM vtiger_loginhistory WHERE
user_name='' ORDER BY login_id DESC]
-- Executing [check_users@customer:7] Set("SIP/500-0000006e",
"STATUS=") in new stack
-- Executing [check_users@customer:8] GotoIf("SIP/500-0000006e",
"0?get_phone,1:loop") in new stack
-- Goto (customer,check_users,3)
-- Executing [check_users@customer:3] NoOp("SIP/500-0000006e",
"") in new stack
-- Executing [check_users@customer:4] Set("SIP/500-0000006e",
"USER_ID=") in new stack
-- Executing [check_users@customer:5] GotoIf("SIP/500-0000006e",
"1?call_manager,1") in new stack
-- Goto (customer,call_manager,1)
-- Executing [call_manager@customer:1] NoOp("SIP/500-0000006e",
"") in new stack
-- Executing [call_manager@customer:2] Set("SIP/500-0000006e",
"USER_ID=7") in new stack
-- Executing [call_manager@customer:3] Set("SIP/500-0000006e",
"USER_NAME=Jan Hlava") in new stack
-- Executing [call_manager@customer:4] Set("SIP/500-0000006e",
"STATUS=Signed off") in new stack
```

Výpis hovoru neznámého zákazníka na manažera

```
-- Executing [call_manager@customer:5] GotoIf("SIP/500-0000006e", "0?get_phone,1:get_mobile,1") in new stack
-- Goto (customer,get_mobile,1)
-- Executing [get_mobile@customer:1] NoOp("SIP/500-0000006e", "") in new stack
-- Executing [get_mobile@customer:2] Set("SIP/500-0000006e", "EXT=605") in new stack
-- Executing [get_mobile@customer:3] Goto("SIP/500-0000006e", "incoming_calls,605,1") in new stack
-- Goto (incoming_calls,605,1)
-- Executing [605@incoming_calls:1] AGI("SIP/500-0000006e", "agi://158.196.244.250/incoming.agi") in new stack
== Manager 'vtiger' logged on from 127.0.0.1
-- AGI Script Executing Application: (Monitor) Options: (wav,/VtigerAsteriskConnector/records//045f731038f545a4a6f1ae8443211ba8,m)
-- AGI Script Executing Application: (Dial) Options: (SIP/605,60)
-- Called SIP/605
-- SIP/605-0000006f is ringing
-- SIP/605-0000006f is ringing
-- SIP/605-0000006f answered SIP/605-0000006e
> 0x7f03b8060bb0 -- Probation passed - setting RTP source address to 158.196.194.151:8000
== Manager 'vtiger' logged off from 127.0.0.1
-- <SIP/605-0000006f>AGI Script agi://158.196.244.250/incoming.agi completed, returning 0
-- Executing [605@incoming_calls:2] Hangup("SIP/500-0000006e", "") in new stack
== Spawn extension (incoming_calls, 605, 2) exited non-zero on 'SIP/605-0000006e'
```
