



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH
TECHNOLOGIÍ

ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF TELECOMMUNICATIONS

ŠETŘÍCÍ AUTOMAT PRO PBX ASTERISK A VAZBA NA VOIP OPERÁTORY V ČR

LEAST COST ROUTING (LCR) IMPLEMENTATION IN PBX ASTERISK
AND RELATIONS TO VOIP PROVIDERS IN CZECH REPUBLIC

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

AUTOR PRÁCE

AUTHOR

VLADISLAV WALEK

VEDOUCÍ PRÁCE

SUPERVISOR

Ing. PAVEL ŠILHAVÝ, Ph.D.

BRNO 2009



VYSOKÉ UČENÍ
TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

Ústav telekomunikací

Bakalářská práce

bakalářský studijní obor
Teleinformatika

Student: Vladislav Walek

ID: 98425

Ročník: 3

Akademický rok: 2008/2009

NÁZEV TÉMATU:

Šetřící automat pro PBX Asterisk a vazba na VoIP operátory v ČR

POKYNY PRO VYPRACOVÁNÍ:

Provedte průzkum VoIP operátorů v ČR. Seznamte se s vlastnostmi Softswitche Asterisk. Implementujte v PBX Asterisk šetřící automat. Šetřící automat by měl být schopen určit, zdali se jedná o VoIP telefonní číslo a kterému operátoru patří. V případě, že se nebude jednat o VoIP telefonní číslo, určí nejlevnějšího poskytovatele a bude přes něj směřovat hovor. Volajícího upozorní na cenu hovoru a sdělí mu vybraného operátora. Šetřící automat implementujte tak, aby jeho parametry bylo možno snadno a přehledně konfigurovat.

DOPORUČENÁ LITERATURA:

[1] Meggelen, J.V, Smith, J., Madsen, L. Asterisk™ The Future of Telephony. Sevastopol: O'Reilly Media, Inc., 2005. ISBN 0-596-00962-3. [2] Bazala, D. Telekomunikace a VoIP telefonie I. Praha: BEN - technická literatura, Praha 2006, ISBN 80-7300-201-9. [3] WALLACE, Zdeněk. VoIP bez předchozích znalostí. Brno: Computer Press, 2007. 232 s. ISBN 978-80-251-1458-2.

Termín zadání: 9.2.2009

Termín odevzdání: 2.6.2009

Vedoucí práce: Ing. Pavel Šilhavý, Ph.D.

prof. Ing. Kamil Vrba, CSc.

Předseda oborové rady

UPOZORNĚNÍ:

Autor bakalářské práce nesmí při vytváření bakalářské práce porušit autorská práva třetích osob, zejména nesmí zasahovat nedovoleným způsobem do cizích autorských práv osobnostních a musí si být plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.

Abstrakt

Šetřící automat je aplikace, která určuje nejlevnější cenu pro vytočené číslo a pak směřuje hovor přes nejlevnějšího poskytovatele. Pracuje s databází, kde čte hodnoty cen a jiné informace. Je napsán v jazyce Perl a slouží pro implementaci v VoIP ústředně Asterisk. Šetří peníze účastníka a snižuje náklady za volání. Je určen pro malé a středně velké obchodní společnosti.

Klíčová slova.

ENUM, adresa SIP URI, standart E.164, SIP protokol, UDP protokol, RTP protokol, PTSN telefonie, VoIP telefonie, Asterisk PBX, DNS systém, Perl, PostgreSQL, AGI.

Abstract

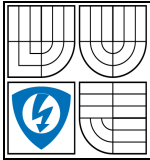
Least Count Routing is application, which establishes the cheapest price for dialed number and routes call forward the cheapest provider. It works with database, where LCR reads the value of price and other informations. It have been written in Perl a serves for implementation in VoIP PBX Asterisk. It saves money of subscriber and lowers costs for calling. LCR is intended for small and medium business corporations.

Key words.

ENUM, address SIP URI, standard E.164, SIP protocol, UDP protocol, RTP protocol, PTSN telephony, VoIP telephony, Asterisk PBX, DNS system, Perl, PostgreSQL, AGI.

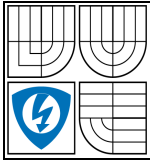
Bibliografická citace:

WALEK, V. Šetřící automat pro PBX Asterisk a vazba na VoIP operátory v ČR. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2009. 53 s. Vedoucí bakalářské práce Ing. Pavel Šilhavý, Ph.D.

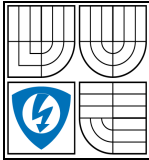


OBSAH

1. ÚVOD.....	4
2. TELEFONNÍ OPERÁTOŘI V ČESKÉ REPUBLICE.....	5
2.1 802.CZ.....	5
2.2 VOIPEX.....	5
2.3 NETPHONE.....	5
2.4 HA-LOO.....	5
2.5 SOFTPHONE.....	6
2.6 VIPHONE.....	6
3. ADRESACE V TELEFONNÍCH SÍTÍCH.....	8
3.1 HISTORIE ADRESACE.....	8
3.2 STANDART E.164.....	8
3.3 ČÍSLOVÁNÍ V KLASICKÉ TELEFONII.....	9
3.4 ADRESOVÁNÍ V IP TELEFONII.....	9
3.5 ADRESOVÁNÍ V PROTOKOLU SIP.....	10
3.6 PŘEDČÍSLÍ OPERÁTORŮ.....	11
4. PROTOKOL SIP.....	13
4.1 CO JE TO PROTOKOL SIP.....	13
4.2 METODY SPOJENÍ POMOCÍ SIP.....	13
5. ENUM.....	16
5.1 CO JE TO ENUM.....	16
5.2 Z HISTORIE ENUM.....	17
5.3 PŘEVOD ČÍSLA NA IDENTIFIKÁTOR URI.....	17
5.4 VÝHODY A NEVÝHODY SYSTÉMU ENUM.....	18
6. ASTERISK.....	19
6.1 POBOČKOVÁ ÚSTŘEDNA ASTERISK PBX.....	19
7. PERL, AGI A POSTGRESQL.....	21
7.1 PERL.....	21
7.2 AGI (THE ASTERISK GATEWAY INTERFACE).....	21
7.3 POSTGRESQL.....	22
8. ŠETŘÍCÍ AUTOMAT.....	23
8.1 AUTOMAT PRO VYHLEDÁVÁNÍ PLATNÝCH ČÍSEL.....	23
8.2 PŘEDČÍSLÍ ZDARMA.....	27
8.3 ČERNÝ SEZNAM.....	30
8.4 KONFIGURACE VOLACÍHO PLÁNU.....	31
8.5 VSTUPNÍ HODNOTY.....	33
9. KONFIGURAČNÍ MENU ŠETŘÍCÍHO AUTOMATU.....	37
10. ZÁVĚR.....	46
11. SEZNAM POUŽITÉ LITERATURY.....	47
12. PŘÍLOHA A. - ZDROJOVÝ KÓD ŠETŘÍCÍHO AUTOMATU.....	48
12.1 APLIKACE PRO VYHLEDÁNÍ NEJLEVNĚJŠÍ CENY.....	48
12.2 ZDROJOVÝ KÓD APLIKACE PRO VYHLEDÁVÁNÍ ČÍSEL ZDARMA.....	50
12.3 APLIKACE PRO BLOKOVÁNÍ PŘÍCHOZÍCH HOVORŮ.....	52



12.4 INSTALACE ŠETŘÍČÍHO AUTOMATU.....52

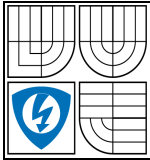


SEZNAM OBRÁZKŮ

OBR. 1: PRŮBĚH SPOJENÍ HOVORU.....	14
OBR. 2: REGISTRACE ÚČASTNÍKA K SIP SERVERU.....	15
OBR. 3: PRINCIP DNS SYSTÉMU.....	16
OBR. 4: DIAGRAM ZOBRAZUJÍCÍ PRŮBĚH ŠETRČÍHO AUTOMATU.....	29
OBR. 5: ZÁLOHA TABULEK V PROGRAMU PGADMIN.....	35
OBR. 6: ÚVODNÍ STRÁNKA KONFIGURAČNÍHO MENU.....	36
OBR. 7: MENU PRO NASTAVOVÁNÍ PLATNÝCH ČÍSEL.....	37
OBR. 8: VÝPIS CEN PŘEDČÍSLÍ OPERÁTORA.....	38
OBR. 9: STRÁNKA PRO KONFIGURACI OPERÁTORŮ.....	39
OBR. 10: VÝPIS OPERÁTORŮ V ŠETRČÍM AUTOMATU.....	40
OBR. 11: MENU PRO KONFIGURACI BEZPLATNÝCH ČÍSEL.....	41
OBR. 12: KONFIGURACE ČERNÉHO SEZNAMU.....	42
OBR. 13: VYHLEDÁVÁNÍ ČÍSEL V ŠETRČÍM AUTOMATU.....	43
OBR. 14: PROVEDENÍ ZÁLOHY SYSTÉMU.....	44

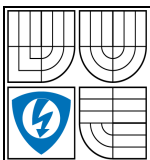
SEZNAM TABULEK

TAB. 1: SROVNÁNÍ CEN OPERÁTORŮ V ČR (* - NENÍ UVEDENO).....	7
TAB. 2: MEZINÁRODNÍ TVAR ČÍSLA DLE E.164.....	8
TAB. 3: PŘEDČÍSLÍ PRO PTSN LINKU (PODLE KRAJŮ).....	12
TAB. 4: TABULKA PŘEDČÍSLÍ MOBILNÍCH OPERÁTORŮ.....	12



1. ÚVOD

Hlavním problémem při tvorbě IP telefonie je propojení s doporučením ITU, který definuje adresaci mezi všemi prvky celé telefonní sítě, jak normální PSTN linky tak mobilní telefonie nebo VoIP. Služby přes tyto sítě jsou provozovány přes operátory a jsou zpoplatněny. Každý operátor poskytuje určitou službu za určitou cenu, a proto nejde najít nejlevnějšího operátora, přes kterého by se všechny hovory směrovaly. Nejlepším řešením je vytvoření programu, který na základě vstupních hodnot, bude určovat nejlevnějšího operátora pro aktuální volané číslo a pak přes něj bude hovor směřovat.



2. TELEFONNÍ OPERÁTOŘI V ČESKÉ REPUBLICE

Na dnešním trhu figuruje velké množství firem zprostředkující telefonní služby na principu IP telefonie. Mezi ty největší patří:

- 802.cz
- VoIPEX
- NetPhone
- Ha-Loo
- SoftPhone
- VIPhone
- a další.

2.1 802.CZ

Asi největším operátorem je firma 802.cz. Nabízí telefonní služby 802.VOX svým zákazníkům. Při zřízení služby nabízí volbu vlastního čísla za určitý poplatek, při připojení k internetu přes jmenovaného operátora nabízí volbu čísla zdarma. Také mezi nadstandardní služby nabízí možnost přenesení čísla od jiného operátora k novému také za poplatek nebo připojení vlastní ústředny k hlavní PBX. Tato služba je ale velice zpoplatněna, protože zákazník může si svoji ústřednu nakonfigurovat a volat tak zdarma.

2.2 VOIPEX

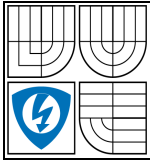
Tento operátor nabízí skoro stejné služby jako předchozí, navíc přidává volání v rámci ENUM zdarma. Nabízí i možnost výběru dotovaného telefonu nebo nedotovaného.

2.3 NETPHONE

NetPhone nabízí oproti jiným možnost zřízení testovacího účtu bez další aktivace. Nabízí emailovou podporu, hlasovou schránku a sms.

2.4 HA-LOO

Ha-Loo nabízí online výpis hovorů, možnost přenesení čísla za velký poplatek v řádu tisíc korun.

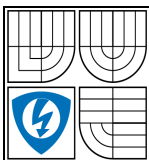


2.5 SOFTPHONE

Tento operátor nabízí bezplatné vyzkoušení služby volání, přidělení více čísel k telefonnímu účtu a možnost připojení k účtu ze zahraničí. Mezi základní funkce patří vteřinové počítání času hovorů zaslání sms a emailová schránka.

2.6 VIPHONE

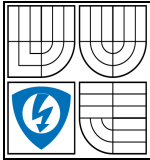
VIPhone nabízí svým zákazníkům zřízení účtu zdarma, volbu vlastního čísla zdarma a aktivaci účtu prostřednictvím SMS. Operátor účtuje hovorné pomocí kreditu a ne paušálu což je výhodné pro fyzické osoby a domácnosti, kdy nemusí zákazník platit i když si nezavolal.



Pro srovnání všech služeb a cen hovorů je zde srovnávací tabulka (Tab. 1):

	802.cz	VoIPEX	NetPhone	Ha-Loo	SoftPhone	VIphone
Zřizovací poplatek	bez poplatku	bez poplatku	200,00 Kč	1,19 Kč	500,00 Kč	bez poplatku
Měsíční paušál	kredit	kredit	kredit	kredit	kredit	kredit
Volání v rámci sítě	zdarma	zdarma	zdarma	zdarma	zdarma	zdarma
Volání do jiných VoIP operátorů	zdarma	zdarma	*	ano	ano	ano
Volání do pevné sítě – ve špičce v Kč	0,49	0,85	1	1	1,1	1,8
Volání do pevné sítě – mimo špičku v Kč	0,26	0,5	0,6	0,5	0,9	0,5
Volání do mobilní sítě – ve špičce v Kč	3,17	3,85	4,5	4	5	3,8
Volání do mobilní sítě – mimo špičku v Kč	3,17	3,85	4,5	4	5	3,8
Volání s předvolbou 800	zdarma	zdarma	zdarma	zdarma	zdarma	zdarma
Tísňová volání	zdarma	zdarma	zdarma	zdarma	zdarma	zdarma
Privátní sítě	ano	ano	ano	ano	ano	ano
Bezplatné vyzkoušení	ne	30,00 Kč	ano	ano	ano	ne
Více čísel k telefonnímu účtu	ne	ne	ne	ne	ano	ne
Vteřinové počítání času hovorů	ne	ne	ano	ano	ano	ano
Emailová schránka	ne	ne	ano	ano	ano	ano
Přenos čísla	2 000 Kč	1 900 Kč	*	1 500 Kč	ano	*
Volba vlastního čísla	0,84 Kč	ne	1,00 Kč	*	*	ano
Připojení vlastní PBX k hlavní ústředně	ano	ano	ano	ano	ano	ano
Možnost dotovaného telefonu	ano	ano	ano	ano	ne	ano
Testovací účet	ne	ano	ano	ano	ano	ne
Hlasová schránka	ano	ano	ano	ano	ano	ano
Podpora ENUM	*	ano	*	*	*	*
Online výpis hovorů	ano	ne	*	ano	*	ne
Softwarový telefon	ano	ano	ano	ano	ano	ano
Možnost zřízení i internetu	ano	ne	ne	ne	ano	ne
Telefonní seznam uživatelů sítě	ano	*	*	*	*	*

Tab. 1: Srovnání cen operátorů v ČR (* - není uvedeno)



3. ADRESACE V TELEFONNÍCH SÍTÍCH

3.1 HISTORIE ADRESACE

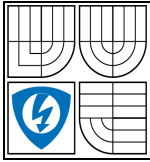
Číslování v telefonních sítích je dáno hlavně historickými důvody a konstrukcí telefonních přístrojů s vestavěnými numerickými klávesnicemi, ať už tlačítkovými nebo rotačními. První ústředny se začaly stavět na začátku 20. století jako základ místních sítí, které ale nebyly vzájemně propojeny. Tyto ústředny byly zpočátku manuální a používaly tzv. dekadického číslovacího principu. Toto číslování fungovalo na principu kde účastníci byli adresováni prostými dekadickými čísly, které byly sdělovány spojovatelce a ta spojila hovor, pak při automatických ústřednách jednoduše odvolit na číselnici. Délka čísla závisela na velikosti ústředny a na počtu účastníků, která byla v minulém století velmi malá, řádově čtyř-číselná. Později se začaly místní ústředny spojovat a vytvářet tak různé složitější a rozlehlé sítě, pak i meziměstské až mezinárodní. Postupem času se vytvořila hierarchicky strukturovaná síť s místní, uzlovou, meziuzlovou a tranzitní rovinou. Na těchto rovinách se používalo různých číslovacích systémů, a proto organizace ITU rozhodla, že stanoví doporučení o realizování v moderních telefonních sítích. Toto doporučení bylo vydáno pod názvem E.164 a dodnes je toto doporučení základem na němž je postaveno adresování jak mezinárodních tak i v místních sítích.

3.2 STANDART E.164

Tento standart je doporučením organizace ITU (International Telecommunication Union), které definuje mezinárodní veřejný telefonický číslovací plán používaný jak v klasické telefonii PSTN nebo v mobilní telefonii, tak i v IP telefonii. Definuje formát telefonního čísla a prefix pro danou zemi. Standart E.164 lze rozdělit na 3 struktury CC – Country Code – směrové číslo země, NDC – National Destination Code – národní směrové číslo, SN – Subscriber Number – účastnické číslo (Tab. 2).

CC	NDC	SN
----	-----	----

Tab. 2: Mezinárodní tvar čísla dle E.164



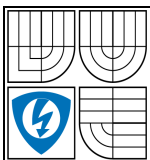
V mezinárodní síti je délka čísla variabilní, ale maximální délka nesmí přesahovat 15 číslic. Části NDC a SN se říká NSN – National Significant Number – národní číslo.

3.3 ČÍSLOVÁNÍ V KLASICKÉ TELEFONII

V klasické telefonii existují dva druhy číslovacích plánů – otevřený a uzavřený. Otevřený plán je takový, kde jednotlivé ústředny nebo dílčí sítě mají přidělena zjevná směrová čísla. Při uskutečnění hovoru mezi dvěma různými ústřednami je třeba použít toto směrové číslo, při hovoru v rámci jedné ústředny není třeba jej použít. Aby bylo rozlišeno zda se volí zjevné směrové číslo a ne číslo ve vlastní ústředně musí se zadat přestupné či rozlišovací číslo zvané prefix, které není součástí číslovací kapacity. Kdysi se za prefix volilo 00, ale dnes jej nahradilo znaménko +, které je součástí každé klávesnice na telefonním přístroji. Pro Českou Republiku toto směrové číslo je dle doporučení E.164 420. Dalším plánem je plán uzavřený jinak též nazývaný skrytý. V tomto plánu není viditelné členění čísla na různě administrativní, geografické nebo fyzické oblasti. Tyto složky mohou být zahrnuty v čísle, avšak volajícímu účastníkovi zůstávají skryty. Nejsou v něm potřeba žádné rozlišovací nebo přestupná čísla, protože volba je vždy jednoznačná. Délka čísla je pevná, což přináší zjednodušení do procesů spojování hovorů, protože je určeno, zda účastník již odvolil všechny číslice. V této době je uzavřený číslovací plán používán v celé České Republice od roku 2002 a délka národního čísla je stanovena na 9 číslic s výjimkou čísel služeb, která jsou kratší např. čísla tísňových volání a jiné.

3.4 ADRESOVÁNÍ V IP TELEFONII

Sítě s protokolem IP se začaly rozvíjet v dobách, kdy telefonní sítě již byly rozšířené a přepracované. Proto se začaly rozvíjet nové standardy a doporučení, které zahrnovaly odlišné a mnohem bohatší možnosti adresace. V zásadě lze IP telefonii rozlišit na modely podle využití. Prvním modelem je propojení hovorů mezi dvěma stanicemi v IP síti, nejčastěji dvou počítačů nebo telefonů pracujících na protokolu IP nebo spojení počítače a IP telefonu. Při takovém spojení jsou počítače (telefony) zaregistrovány k jiným PBX ústřednám a lze využít libovolných metod adresace použitím DNS serverů a fyzické adresace pomocí IP adresy. Druhým modelem je spojování hovorů mezi klasickou a IP telefonní sítí. Lze použít dva způsoby, jeden je nainstalovat do ústředny speciální hardwarovou kartu pro připojení PSTN sítě nebo zaregistrovat svou ústřednu u některého z operátorů. Vlastně lze říci, že tyto prostředky



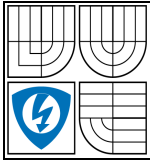
jsou jistými bránami – gateway. Při tomto řešení číslo volaného účastníka musí být voleno ve formátu akceptovatelným klasickou sítí. Tato brána slouží jako jistý překladač čísel, kdy brána sama provádí mapování čísla a pomocí databáze jej přeloží na IP adresu operátora, kde je zaregistrovaný účet, nebo hovor přesměruje na hardwarovou kartu. Třetím modelem je, kdy uživatel volá z klasické PSTN linky uživateli na druhé straně, který také používá klasickou linku, ale hovor je směrován přes IP telefonii. Takový způsob je nejčastěji používán ve volání mezi národy, které geograficky dělí několik států. Tato služba je nejčastějším způsobem volání třeba ze Spojených Států Amerických do České Republiky, kdy uživatel si zakoupí speciální kartu obsahující kód, který zadá před vytočením čísla volaného. Tento to hovor je přesměrován na PSTN – IP bránu, pak je poslán po IP síti na určitou SIP adresu, posléze je přepojen IP – PSTN bránou na klasickou telefonii a směrován na účastnické koncové zařízení volaného. Tímto způsobem lze docílit ušetření peněz, protože uživatel nemusí platit různým operátorům za přepojení hovoru. Tomuto způsobu se často říká „Tranzitní volání přes internet“ nebo „IP Tranzit“.

3.5 ADRESOVÁNÍ V PROTOKOLU SIP

Jelikož nejčastějším protokolem pro uskutečňování hovorů je protokol SIP, je adresování již dané a ne volitelné, popsané v doporučení RFC 3696. Tento protokol má velkou budoucnost, neboť byl zvolen jako signalizační protokol pro síť IMS, které se začínají budovat. SIP protokol používá pro adresaci koncových stanic tzv. formu SIP URI:

sip: uživatelské_jméno@doména

Někdy se zastupuje uživatelské jméno telefonním číslem pro lepší orientaci při volání, názvy a jména se většinou používají jako doplňkové informace. Tento způsob adresace je universální, avšak není kompatibilní s číslováním v klasických telefonních sítích. Lze jej použít, pokud účastník si nainstaluje nebo používá VOIP terminál na svém počítači a může tento tvar adresy lehce zapsat a vytočit. Síť IP telefonie provozované různými poskytovateli telekomunikačních služeb nebo pokročilejší akademické sítě s různými uživateli využívají pokročilejších technologií jako přihlašování účastníků k centrálním prvkům IP sítě. Pro protokol SIP jsou tyto prvky nazývány SIP Proxy, pro protokol H.323 je to tzv. gatekeeper. Účastník se zaregistruje v ústředně a může uskutečňovat hovory v rámci své místní sítě takřka bez omezení. Problém nastane, kdy bude chtít zavolat na jinou adresu volaného, adresu která není



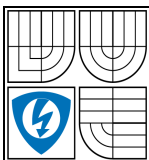
registrována v jeho ústředně. V tomto případě musí obsahovat SIP Proxy údaj o doméně, která specifikuje adresu SIP Proxy operátora, ve které je koncový účastník registrován. SIP Proxy se obrátí na systém DNS a ten adresu převede a vrátí ji zpět. Protože operátor by měl zajistit spolehlivost a účinnost služby VOIP často provozuje více než jeden SIP Proxy. Pro tento účel se používá příkaz SRV definovaný v doporučení RFC 2782, který slouží jako záznam všech adres SIP Proxy. SRV record neboli SRV záznam je jedna z kategorií dat upřesňující informace o dostupných službách, je vyžadován pro protokol SIP i třeba pro protokol XMPP pro rychlé posílání zpráv. SRV záznam lze zobrazit takto:

```
_sip._udp.domena.cz 86400 IN SRV 0 5 5060 sipproxy.domena.cz
```

Kdy tento záznam ukazuje na server sipproxy.domena.cz poslouchajícím na portu 5060 s prioritou 0 a váhou 5. Při přidělování čísel účastníkům by měl operátor uvážit, aby čísla neměla geografický charakter, protože jednou z hlavních vlastností IP telefonie je její mobilita a přenositelnost a mohlo by docházet k určitým nejasnostem v účtování hovorů, např. při cestování uživatele. Kdyby se dávalo důraz na geografické číslování mohlo by taky dojít k omezení procesů registrace účastníka, např. úspěšnou registraci pouze z určitého rozsahu adres. Proto je velmi důležité držet se zásad a efektivně rozpracovat všechny aspekty vyskytující se ve VOIP při vlastní tvorbě IP telefonii.

3.6 PŘEDČÍSLÍ OPERÁTORŮ.

Číslo dle standartu E.164 obsahuje mezinárodní prefix, předčíslí operátora a 6 číslic účastnického čísla. U linky PSTN je délka předčíslí pouze 2-místní, u operátorů mobilních sítí a jiných služeb jako bezplatné linky nebo komerčních čísel. Následující tabulka zobrazuje většinu předčíslí v České Republice(Tab. 3):



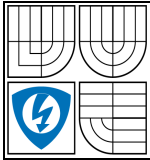
Jihočeský Kraj	38
Jihomoravský Kraj	51, 53, 54
Karlovarský Kraj	35
Královéhradecký Kraj	49
Liberecký Kraj	48
Moravskoslezský Kraj	55, 59
Olomoucký Kraj	58
Pardubický Kraj	46
Plzeňský Kraj	37
Středočeský Kraj	31, 32
Ústecký Kraj	41, 47
Vysočina	56
Zlínský Kraj	57
Praha	2

Tab. 3: Předčíslí pro PTSN linku (podle krajů)

Následující tabulka zobrazuje předčíslí mobilních operátorů (Tab. 4):

T-Mobile	603, 604, 605, 730, 731, 732, 733, 734, 735, 736, 737, 738, 739
Telefónica O2	601, 602, 606, 607, 720, 721, 722, 723, 724, 725, 726, 727, 728, 729
Vodafone	608, 773, 774, 775, 776, 777

Tab. 4: Tabulka předčíslí mobilních operátorů



4. PROTOKOL SIP

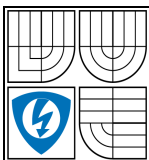
4.1 CO JE TO PROTOKOL SIP

Session Initiation Protokol je signalizační protokol užívající se k řízení multimediálních relací v IP sítích. Obsahuje zprávy pro sestavení, udržení a ukončení hovorů. První verze protokolu byla popsána v doporučení RFC 2543, v současnosti jej definuje dokument RFC 3261. Tento protokol pracuje zároveň s protokoly RTP, který zaručuje vlastní přenos hlasu po síti IP, a detaily o vlastnostech zahajovaného přenosu popisuje protokol UDP přenášený v těle SIP. SIP jako takový je založen na protokolu HTTP, který je dnes nejúspěšnějším a nejpoužívanějším protokolem v Internetu. SIP je navržen na aplikační vrstvě tak, aby byl snadno implementovatelný, rozšířitelný a flexibilní. SIP byl vyvíjen již od poloviny 90. let pracovní skupinou MMUSIC v rámci IETF. V roce 2002 pracovní skupina „SIP“ vypustila hlavní jádro protokolu a jeho celou strukturu. SIP je navržen v souladu s modelem Internetu. SIP je založen na end-to-end koncepci, díky tomu je odlišný od klasické PSTN telefonie, protože veškerá logika je uložena v koncových zařízeních. Tyto koncová zřízení znají jednotlivé stavy komunikace, což zaručuje větší odolnost proti chybám. Nepochybně je tato vlastnost velkým přínosem, protože do protokolu SIP lze implementovat nové služby, které mohou zvýšit využití v jiných systémech. Tyto služby lze těžko nasadit v klasických telefoních.

4.2 METODY SPOJENÍ POMOCÍ SIP

SIP signalizace není implementovaná pomocí elektrických signálů tak jako v PSTN, ale pomocí TCP/IP paketů. Pro vytvoření a řízení relace musí v první řadě SIP účastníka lokalizovat tj. nalezení cílové stanice, pak zjištění stavu účastníka jestli je schopen relaci navázat, zjištění možností účastníka, vlastní navázání spojení pomocí RTP a UDP, řízení spojení a jako poslední ukončení spojení. SIP protokol využívá následující příkazy:

- REGISTER - registrace účastníka na SIP serveru
- INVITE - zahájení komunikace
- ACK - potvrzení zahájení relace
- CANCEL - přerušení relace před jejím spojením

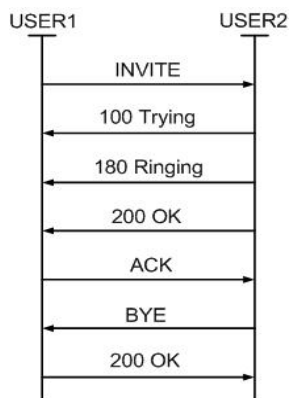


BYE - ukončení relace

Podmínkou druhé strany je odpověď v číselném tvaru. Tyto odpovědi nemají identické označení, ale jsou řazeny do skupin. Tyto odpovědi jsou popsány v RFC dokumentu, ale záleží na vývojové firmě, jak je implementuje do svého produktu. Nejdůležitější odpovědi jsou zde:

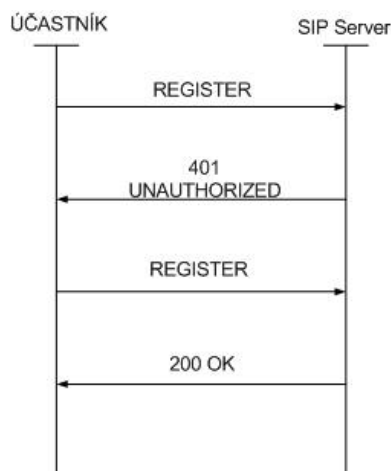
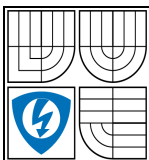
- 1XX - informační zprávy
- 2XX - úspěšné ukončení žádosti
- 3XX - přesměrování
- 4XX - chyba klienta
- 5XX - chyba serveru
- 6XX - fatální chyba

Mezi nejznámější zprávy jsou 100 – Trying kde proxy server zkouší vytvořit spojení, 180 – Ringing, které oznamují vyzvánění volaného. Odpovědi 200 až 299 oznamují, že požadavek byl akceptován a zpracován. Klasický průběh spojení znázorňuje následující obrázek (Obr. 1):



Obr. 1: Průběh spojení hovoru

Uživatele musí svůj účet zaregistrovat na Registrar serveru, aby byli dosažitelnými. Uživatel vyšle příkaz Register a Registrar server vyšle, buď 401 – nezaregistrováno nebo 200 OK – registrace úspěšná. Následující obrázek (Obr. 2) ukazuje registraci:



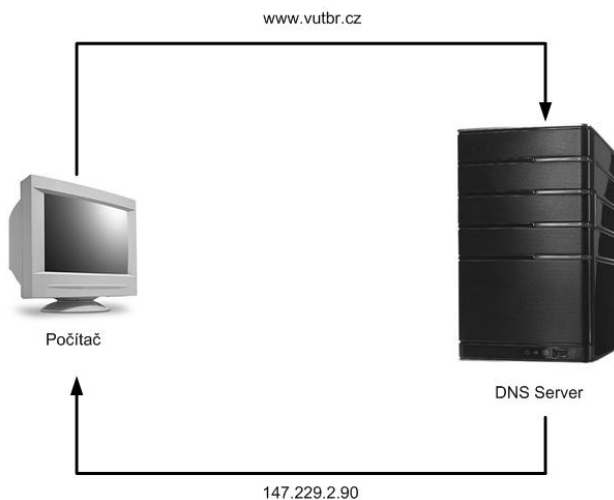
Obr. 2: Registrace účastníka k SIP serveru

Dalším možným způsobem spojení je přes proxy server. Nejdřív je odeslána zpráva od uživatele na jeho proxy server, ten jej pošle na druhý proxy server a ten na účet volaného. Další zprávy po spojení již jsou zasílány přímo, proxy server již nehraje ve spojení roli. Tímto se zrychlí a zjednoduší hovor, jelikož zprávy jsou doručeny s menším zpožděním. Protokol SIP umožňuje velké množství funkcí a služeb, pomocí kterých lze uskutečňovat hovory na IP síti za minimální náklady a výkon celého systému jak proxy serverů tak samotné IP telefonie.

5. ENUM

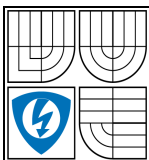
5.1 CO JE TO ENUM

Enum je softwarový systém související se systémem DNS. Domain Name System je hierarchický systém doménových jmen, který se používá k překladu doménových jmen na adresy IP. Prostor doménových jmen tvoří strom, kterého uzly obsahují nějakou část domény a odkazy na své podřízené domény. Kořenem tohoto stromu tvoří kořenová doména, která se zapisuje jako tečka a pod ní se nacházejí domény nejvyšší úrovně TLD. Takové domény mohou být státní jako například cz, sk, nebo komerční – com, nebo vzdělávací – edu. Celý strom lze rozdělit do určitých zón a její správu rozdělit mezi další administrátory. DNS systém vznikl hlavně kvůli tomu, aby si lidé nemuseli pamatovat číselné kombinace adres webových stránek, adres severů atd. V podstatě je DNS jednoduchým překladačem. Funkce DNS si můžeme představit graficky na následujícím obrázku(Obr. 3):



Obr. 3: Princip DNS systému

Uživatel napíše do svého prohlížeče doménový název portálu, počítač tento příkaz zpracuje a odešle jej na server DNS. Ten dotaz přeloží na IP adresu a odešle jej zpátky na počítač uživatele. Jak již bylo řečeno, Enum úzce souvisí s DNS. V podstatě je Enum taky překladačem, jeho hierarchie se ničím neliší od DNS, ale nepřekládá domény na IP adresy, ale překládá telefonní čísla na jiná jména nebo identifikátory sítě. Princip systému Enum ve VoIP spočívá v mapování telefonního čísla podle normy E.164 na řetězec SIP URI, který slouží k identifikaci uživatele VoIP v internetu. Může



také překládat telefonní číslo na jiné zdroje nebo jiné služby jako email, rychlé posílání zpráv nebo odkazy dokumentů atd.

5.2 Z HISTORIE ENUM

Standart „telephone NUmbers Mapping“ jinak ENUM byl popsán v RFC 2916 švédským inženýrem Patrikem Fältströmem, ale jeho začátek jde určit na rok 1999 kdy byla zolžena první pracovní skupina v rámci IETF. V roce 2002 začaly první testovací provozy Enum ve světě. Jelikož protokol Enum vzbudil velký zájem, na základě znalostí a výsledků testovacích provozů byl v roce 2004 popsán nový standart v podobě RFC 3761, který nahradil stávající RFC 2916. Během let 2005 a 2006 se začal nápad mapování čísel rozrůstat ve více zemích a tak byla uvolněna doména e164.arpa pro užívání Enum. Dnes se tento systém vytváří ve spolupráci organizace IETF a mezinárodní telekomunikační unie ITU. V případě ČR je správcem subdomény .0.2.4.e164.arpa je skupina CZ NIC.

5.3 PŘEVOD ČÍSLA NA IDENTIFIKÁTOR URI

Samotné mapování telefonního čísla na URI se provádí, že z mezinárodního telefonního čísla se odstraní všechny přebytečné znaky a zůstanou pouze číslice:

420514149190

Dále pak mezi jednotlivé číslice se vloží tečky:

4.2.0.5.4.1.1.4.9.1.9.0

Číslice se zapíší v opačném pořadí:

0.9.1.9.4.1.1.4.5.0.2.4

A tento tvar se potom vloží do domény e164.arpa a výsledný tvar je:

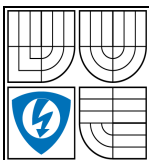
.9.1.9.4.1.1.4.5.0.2.4.e164.arpa

Do systému DNS se odešle dotaz NAPTR se záznamem. Systém nám vrátí příslušnou URI:

sip:420514149190@neco.cz

Pro realizaci je nutné, aby některý z koncových klientů nebo SIP server prováděl ENUM resolving. Jelikož ENUM je pouze standart, systém DNS překládá adresy pro ENUM. DNS musí obsahovat záznam o čísle a SIP URI adrese, například:

`\\IN NAPTR 100 10 „u“ „E2U+sip“ „!^+(.*)$!sip:username@domain.cz!” .`



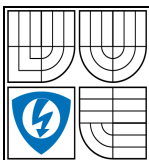
Standart ENUM nám ještě přináší možnost překladu telefonního čísla na jinou služku. Protože SIP protokol je jeden z protokolů pro komunikaci ve VoIP a některé ústředny pracují na jiných protokolech, v systému Enum je definovaná množina dalších protokolů. Pro ústřednu GnuGk – spíše řečeno pro gatekeeper je hlavním protokolem pro sestavení hovorů protokol H.323. Tento protokol je definován v ENUM a proto může uživatel zadat záznam v DNS, který obsahuje URI protokolu H.323:

```
\\IN NAPTR 100 10 „u“ „E2U+h323“ „!^(.*)$!h323:username@domain.cz!“
```

Jak již bylo řečeno, ENUM může být používáno jako překladač telefonních čísel na jiné služby než adresy VoIP, jako email s příkazem „E2U+email:mailto“, jako webová adresa „E2U+web:http“ atd. Vlastně ENUM lze použít jako vyhledávací program, protože na základě identifikace pomocí telefonního čísla lze získat o různých možných přístupech k požadovanému dotazu. Kromě stromu e164.arpa je možno vytvořit i libovolný strom, např. e164.vubtr.cz, ale pouze uvnitř nějaké instituce nebo je třeba doménu zaregistrovat v ITU nebo na serveru CZ NIC. Protože pouze skupina CZ NIC je plným poskytovatelem služby ENUM v České Republice.

5.4 VÝHODY A NEVÝHODY SYSTÉMU ENUM

Možnosti ENUM jsou velmi rozsáhlé a v budoucnosti lze čekat další využití tohoto standartu. V dnešní době je jen malá skupina využívajících tento systém, při rozšíření lze očekávat větší počet lidí, kteří chtějí tento systém zneužít. Jak je známo z historie, všechny otevřené a volně dostupné aplikace a systémy přitahují skupiny „hackerů“ a „crackerů“. V elektronické poště, která je nejvíce podobná IP telefonii se vyskytuje velká nepříjemnost a to nevyžádaná pošta. Lze očekávat, že tento SPAM nějak zapůsobí na VoIP a ENUM jak ve formě textové tak možná i v hlasové.



6. ASTERISK

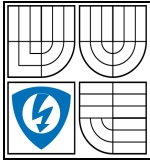
6.1 POBOČKOVÁ ÚSTŘEDNA ASTERISK PBX

Ústředna Asterisk je open source softwarová ústředna pracující na operačních systémech Linux. Oficiálně je Asterisk hybrid TDM a packet voice PBX. Jedná se o platformu IVR (Interactive Voice Response) s funkcí ACD (Automatic Call Distribution). Je vytvářena skupinou Digium, ale protože její kód je dostupný všem některé doplňky jsou programovány všemi lidmi světa. Neznámější verze tohoto systému je 1.4, která přinesla oproti svému předchůdci 1.2 mnoho zajímavých funkcí, jako třeba podporu ENUM a funkci EnumLookup. Tato verze přinesla i podporu jiných operačních systémů jako je BSD, Solaris nebo MacOS. Bohužel systém nepracuje pod systémem Windows, je navržen na Unix. Asterisk přináší spoustu podpory různých hardwarových produktů. Nejpopulárnějším je instalace speciální karty, která slouží pro propojení klasické PSTN linky s pobočkovou ústřednou. Velmi mnoho dalších produktů je podporováno Asterisk ústřednou. Hlavním protokolem, na kterém Asterisk působí je protokol SIP. Je však možno doinstalovat různé doplňky jak sloužící k provozu ústředny, tak i různé podpory jiných protokolů. Spojením Asterisku a OpenH323 lze provozovat i protokol H.323, ale o jeho obsluhu slouží gatekeeper – konkrétně může jím být taky open source GnuGk. Díky integraci protokolu přímo vytvořeného pro Asterisk IAX lze propojit více ústředen dohromady a vytvořit velkou síť VoIP.

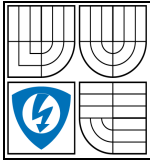
Mezi hlavní funkce Asterisku patří:

- použití jako konferenční server
- použití hlasové schránky
- interaktivní hlasový průvodce
- VoIP gateway pro protokoly MGCP, SIP, H.323
- a jiné

Konfigurace se vytváří pomocí příkazového řádku nebo od verze 1.6 lze používat Web GUI, ale jeho používání nedoporučují přední uživatelé, tato služba je ještě ve fázi vývoje. Základními soubory pro konfiguraci slouží sip.conf po případě oh323.conf pro nastavení samotných uživatelských účtů a jejich správa, pro samotný průběh hovoru slouží soubor extensions.conf. V něm se nastavují všechny stavy od vyvěšení sluchátka po ukončení hovorů. Hlavní základní funkcí je integrace jiných konfiguračních souborů



přímo do `extension.conf` pomocí výrazu „`#include`“. Bohužel toto nastavování funkcí vyžaduje velikou znalost Unix systému, proto je Asterisk určen pro pokročilé uživatele a administrátory sítě. Veliké usnadnění přinesla skupina Trixbox, která přináší možnost stažení Asterisk předinstalovaný na CentOS a plnou podporou Web GUI. Tento systém, ale není určen velkým firmám a uživatelům požadujícím přepracovaný systém s podporou většiny funkcí Asterisku.



7. PERL, AGI A POSTGRESQL.

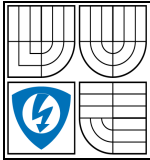
System Asterisk umožňuje integrovat různé skripty a vytvořené programy přímo do funkce ústředny. Pomocí těchto skriptů lze napsat program, který může spravovat hovory, vyhodnocovat různé parametry a průběhy hovorů, jejich směrování a přepínání a jiné. Tyto skripty lze psát v různých jazycích, protože Asterisk se sám nestará o jeho interpretaci nýbrž sám operační systém. Lze je psát v jazyku C nebo C++, ale je potřeba tento program předem zkompileovat, nebo v jiných jazycích, které se kompilují při spuštění. Je pravda, že velmi složitý skript napsaný v jazyce C je oproti jiným jazykům rychlejší, ale pro jednoduché skripty prováděné při vytočení mohou být o pár milisekund delší. Nejlepší alternativou pro jednoduché skripty je Perl nebo Python, který je více flexibilní nežli C.

7.1 PERL

Perl je jazyk, který byl vyvinut pro jednoduchost v programování. Jeho hlavní přednosti jsou přehlednost v definování proměnných a polí, typy proměnných netřeba definovat a rychlost v kompilování. Perl je jazyk vyšší úrovně, byl vytvořen v jazyku C a lze jej přenášet na různé platformy. Pro Asterisk se skript v tomto jazyku hodí, protože perl neobsahuje grafické rozhraní, které není potřeba a jeho implementace v ústředně je z uživatelského hlediska velmi prostá. Jelikož programátor nikdy neví, na kterém systému přímo pracuje ústředna je tento jazyk nejlepší volbou.

7.2 AGI (THE ASTERISK GATEWAY INTERFACE)

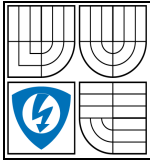
AGI je standardní rozhraní umožňující pomocí externích programů kontrolu a řízení volacího plánu – dialplan. Tyto AGI skripty se nejčastěji používají pro komunikaci s jinými systémy jako je MySQL, PostgreSQL aj. Tímto způsobem lze rozšířit ústřednu Asterisk o nové komplikovanější funkce, které nelze vytvořit v ústředně pouhým nastavováním a konfigurací. Nejčastěji se používají programovací jazyky PERL, PHP nebo Python, jelikož ústředna Asterisk pracuje na Unixových systémech a výše popsané jazyky jsou implementovány a vyvíjeny právě na Unixu. Lze použít i vyšší programovací jazyky, ale jejich složitost může vést k častým chybám. AGI skripty komunikují s Asteriskem přes standardní komunikační kanály Standard Input – standardní vstup a Standard Output – standardní výstup. Existuje i Standard Error – standardní chyba, která určuje jestli se skript provedl bezchybně. Volání skriptu se



provádí v souboru „extensions.conf“ a to způsobem „exten => 123,1,AGI(skript.agi). Pro bezchybnou práci skriptu, je třeba aby byl spustitelný (měl příslušná práva) a nacházel se ve složce /var/lib/asterisk/.

7.3 POSTGRESQL

PostgreSQL je databázový systém pracující na unixových systémech a na Windows. Jeho instalace je náročná, ale jeho výpočetní výkon je vyšší než u jiných systémů. Pro potřeby Asterisku a rozhraní je tento systém výborným řešením. Asterisk nekomunikuje přímo s PostgreSQL, používá speciálního pluginu UnixODBC, který dokáže spojit Asterisk s jakýmkoliv databázovým systémem. Pro potřeby spojení skriptu a PostgreSQL je třeba importovat rozšíření Psql. U programovacího jazyku je tímto rozšířením DBI a DBD::Pg. Pak lze jenom zadávat příkazy do programu a plugin se potom stará o jeho předávání do databáze. Podpora pro PostgreSQL je veliká a proto je tento databázový systém ideální. Příručka nakladatelství O'Reilly – Asterisk, The Future of Telephony popisuje instalaci, konfiguraci Asterisku a PostgreSQL k vzájemnému pracování.



8. ŠETŘÍCÍ AUTOMAT

8.1 AUTOMAT PRO VYHLEDÁVÁNÍ PLATNÝCH ČÍSEL.

Šetřící automat je napsán v jazyce Perl. Jak již bylo řečeno, tento jazyk je přehledný a pro konfiguraci volacího plánu ideální. V šetřícím automatu, dále jen skriptu, bylo třeba propojit Asterisk AGI rozhraní, rozhraní pro připojení k databázi PostgreSQL a jazyk Perl. Popis skriptu:

```
#!/usr/bin/perl -w

use warnings;
use strict;

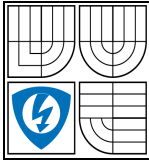
use DBI;
use Asterisk::AGI;
```

Použití balíčku „warnings“ a „strict“ upozorňují na chyby v programu. Balíček DBI je systém pro propojení skriptu s databází. Pro propojení s databází PostgreSQL se používá balíček DBD::Pg, který je implementován v balíku DBI. V něm se také nachází různá další rozšíření hlavně pro připojení s databázemi MySQL, Oracle nebo MS SQL. Další balíček je Asterisk::AGI, v kterém jsou funkce pro čtení z Asterisku a funkce pro řízení volacího plánu.

```
my $AGI = new Asterisk::AGI;
my %input=$AGI->ReadParse();
my $cislo = $ARGV[0];

my $dbh = DBI->connect("DBI:Pg:dbname=providers;host=localhost",
"postgres", "postgres", {'RaiseError'=>1});
```

První řádek předchozího odstavce definuje vytvoření instance AGI k proměnné \$AGI. Do dočasného pole %input vkládá rozhraní AGI všechny informace na vstupu programu z ústředny. Pak lze do proměnné \$cislo vložit první argument z Asterisku, v tomto případě číslo neboli „extenzi“, které účastník vytočil na svém přístroji. Pak do proměnné \$dbh se vloží celý systém připojení k databázi PostgreSQL. Skript se připojí k databázi s následujícími parametry: databáze „providers“, která běží na místním systému localhost, s uživatelským jménem postgres a heslem postgres. Parametr RaiseError nastaven na 1 informuje, že při malé chybě se program „nezhroutl“.



```
my $hash_count = $dbh->prepare("SELECT COUNT(id) FROM providers");
$hash_count->execute();
my $count;
$hash_count->bind_columns(\$count);
$hash_count->fetch();
```

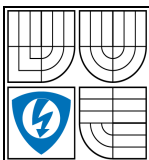
Do dočasné proměnné \$hash_count se запиše příkaz „prepare“ k provedení operace „vyčtení počtu operátorů“ v databázi „providers“. Příkaz „execute“ provede příkaz. Protože v proměnné \$hash_count jsou všechny informace o provedení příkazu, výsledek ale i čas a jiné, je třeba pomocí příkazu „bind_columns“ naplnit do nějaké proměnné pouze výsledek příkazu, v tomto případě do proměnné \$count, a pak jej provést příkazem „fetch“.

Jelikož se operace SELECT do databáze opakuje v celém programu je lepší vytvořit podprogram, který se bude volat v běhu. Vyvaruje se tímto způsobem tvoření chyb v průběhu program a jeho lepší přehlednost.

```
sub select {
    my $m = $_[0];
    my $n = $_[1];
    my $l = $_[2];
    my $k = $_[3];
    my $hashe = $dbh->prepare("SELECT $m FROM $n WHERE $l = $k");
    $hashe->execute();
    my $vysledek;
    $hashe->bind_columns(\$vysledek);
    $hashe->fetch;
    return ($vysledek);
}
```

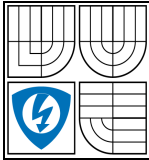
Podprogram „select“ je volán příkazem „&select()“ s parametry v závorce. Konkrétně tento program запиše postupně vstupní parametry do různých proměnných, kde první je název sloupce kde se nachází zjišťovaná hodnota, druhý je název tabulky, z které program hodnotu vyčte, třetí je název porovnávacího sloupce a čtvrtý je hodnota, která se porovná. Funkce vrací pouze hledanou hodnotu. Dále program pokračuje již vyhledáním ceny každého operátora v tabulce a jeho název makra, které je pak použito pro přesměrování hovoru.

```
my $i;
my @cena;
my @makro;
my $num2 = substr($cislo, 0, 3); #promenna num je dvojmistni
predcisli
my $num3 = substr($cislo, 0, 4); #promenna numa je trimistni
predcisli
my $num4 = substr($cislo, 0, 5); #promenna numa je ctyrmistni
```



```
predcisli
my $num5 = substr($cislo, 0, 6); #promenna numa je petimistni
predcisli
my $num6 = substr($cislo, 0, 7); #promenna numa je sestimistni
predcisli
my $num7 = substr($cislo, 0, 8); #promenna numa je sedmimistni
predcisli
my $num8 = substr($cislo, 0, 9); #promenna numa je osmimistni
predcisli
my $emps = $dbh->selectall_arrayref("SELECT table_id, macro FROM
providers",{ Slice => {} });

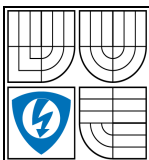
my @table_id;
my @makro_id;
foreach my $emp ( @$emps ) {
    push(@table_id, $emp->{table_id});
    push(@makro_id, $emp->{macro});
}
# Nacteni ceny predcisli kazdeho operatora do pole cena, nacteni
makra kazdeho operatora do pole makra
for ($i = 0; $i<$count; $i++){
    my $cena_sql;
    my $makro_sql;
    my $table_sql = $table_id[$i];
    $makro_sql = $makro_id[$i];
    # pokud se nenajde dvojmistni predcisli v tabulce prejde se
na trimistni
$cena_sql = &select('cena', $table_sql, 'num', $num2) || '7777';
    if ($cena_sql == '7777') {
        # print STDERR "\n predcisli po if 7777: $num\n";
        $cena_sql = &select('cena', $table_sql, 'num', $num3) ||
'7777';
    }
    if ($cena_sql == '7777') {
        # print STDERR "\n predcisli po if 7777: $num\n";
        $cena_sql = &select('cena', $table_sql, 'num', $num4) ||
'7777';
    }
    if ($cena_sql == '7777') {
        # print STDERR "\n predcisli po if 7777: $num\n";
        $cena_sql = &select('cena', $table_sql, 'num', $num5) ||
'7777';
    }
    if ($cena_sql == '7777') {
        # print STDERR "\n predcisli po if 7777: $num\n";
        $cena_sql = &select('cena', $table_sql, 'num', $num6) ||
'7777';}
    if ($cena_sql == '7777') {
        # print STDERR "\n predcisli po if 7777: $num\n";
        $cena_sql = &select('cena', $table_sql, 'num', $num7) ||
'7777';
    }
    if ($cena_sql == '7777') {
        # print STDERR "\n predcisli po if 7777: $num\n";
```



```
        $cena_sql = &select('cena', $table_sql, 'num', $num8) ||  
'9999';    }  
    if ($cena_sql != '9999'){  
        push (@cena, $cena_sql);  
        push (@makro, $makro_sql);  
        print STDERR "\n tabulka: $table_sql, cena: $cena_sql makro:  
$makro_sql\n";  
    }  
}
```

V prvních třech řádcích je definováno proměnná \$i pro „for“ cyklus a pole @cena a @makro, do kterých bude program zapisovat hodnotu ceny a hodnotu makra operátora. Do proměnné \$num se zapíše 2-místní předčísli pro čísla pevné linky a do proměnné \$numa 3-místní předčísli jiných operátorů. Dále následuje cyklus „for“, který do proměnné \$table_id zapíše tabulku předčísli a jejich ceny každého operátora, do proměnné \$makro_sql zapíše hodnotu makra aktuálního operátora. Do proměnné \$cena_sql se vloží hodnota ceny operátora právě vytočeného čísla podle jeho 2-místního předčísli. Pokud se toto předčísli nenachází v tabulce zapíše se to proměnné \$cena_sql hodnota 7777, pak cykl „if“ zapíše do této proměnné cenu operátora podle 3-místního předčísli. Pokud se i toto předčísli nenachází v tabulce to znamená, že předčísli volaného čísla není definováno a přesměruje se na Macro-nedefinováno. Pokud je předčísli nalezeno do pole @cena se přidá hodnota ceny a do pole @makro se přidá makro operátora. Do konzole CLI se vypíše tabulka, cena a makro aktuálního operátora. Následující skript je jádro programu, které již porovnává ceny a určuje nejlevnějšího operátora.

```
my $min;  
my $kon_makro;  
my $delka = @cena;  
my $a = $cena[0];  
my $b;  
my @pole;  
  
for ($i = 0; $i < $delka; $i++){  
    push(@pole, $cena[$i]);  
    push(@pole, $makro[$i]);  
}  
  
for ($i = 0; $i < $delka; $i++){  
    $b = $cena[$i];  
  
    if ($a == $b){  
  
        $min = $a;  
  
    }  
}
```



```
else{
    if ($a < $b){
        $min = $a;
    }
    else{
        $min = $b;
        $a = $b;
    }
}

$delka = @pole;
for ($i = 0; $i < $delka; $i++){
    if ($pole[$i] =~ /$min/){
        $kon_makro = $pole[$i+1];
    }
}

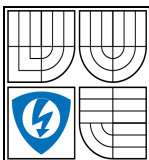
print STDERR "\n vystup z asterisku $cislo, koncovy operator je
$kon_makro\n\n\n";
$AGI->exec('Playback', 'cena');
$AGI->exec('SayDigits', $say_a);
$AGI->exec('Playback', 'korun');
$AGI->exec('SayDigits', $say_b);
$AGI->exec('Playback', 'haleru');
$AGI->exec('Macro', $kon_makro);
```

Do pole @pole se zapíše cena a makro vždy tak, že odpovídající makro operátora je na položce o jeden větší než cena. Tudiž makro vždy následuje cenu. Porovnání se provádí způsobem, že do proměnné \$a se zapíše první prvek z pole @cena, a pak ve „for“ cyklu se do proměnné \$b zapíše druhý prvek z pole @cena. Pokud se ceny rovnají, pak první se zapíše do proměnné \$min. Jinak se do proměnné \$min zapíše nejnižší cena. V cyklu „for“ se v poli @pole najde aktuální cena a jí příslušné makro. Do konzole se vypíše volané číslo a koncový operátor a pak AGI provede příkaz „Macro“ s hodnotou koncového operátora, přes který je hovor přeměrován. Celý program i s komentářem je v zobrazen v příloze A.

8.2 PŘEDČÍSLÍ ZDARMA

U operátora 802.CZ je možnost volání na některá specifická čísla zdarma. Jelikož je tento operátor jedním z největších na českém trhu, je třeba nastavit ústřednu tak aby tuto možnost zvládala. Následující skript nejdřív porovná číslo s databází předčísli zdarma a pokud najde schodu provede patřičnou operaci.

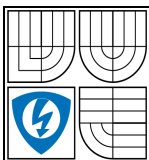
```
sub select {
    my $m = $_[0];
```



```
my $hashe = $dbh->prepare("SELECT makro FROM volne_cisla WHERE
num = $m");
$hashe->execute();
my $vysledek;
$hashe->bind_columns(\$vysledek);
$hashe->fetch;
return($vysledek);
}
```

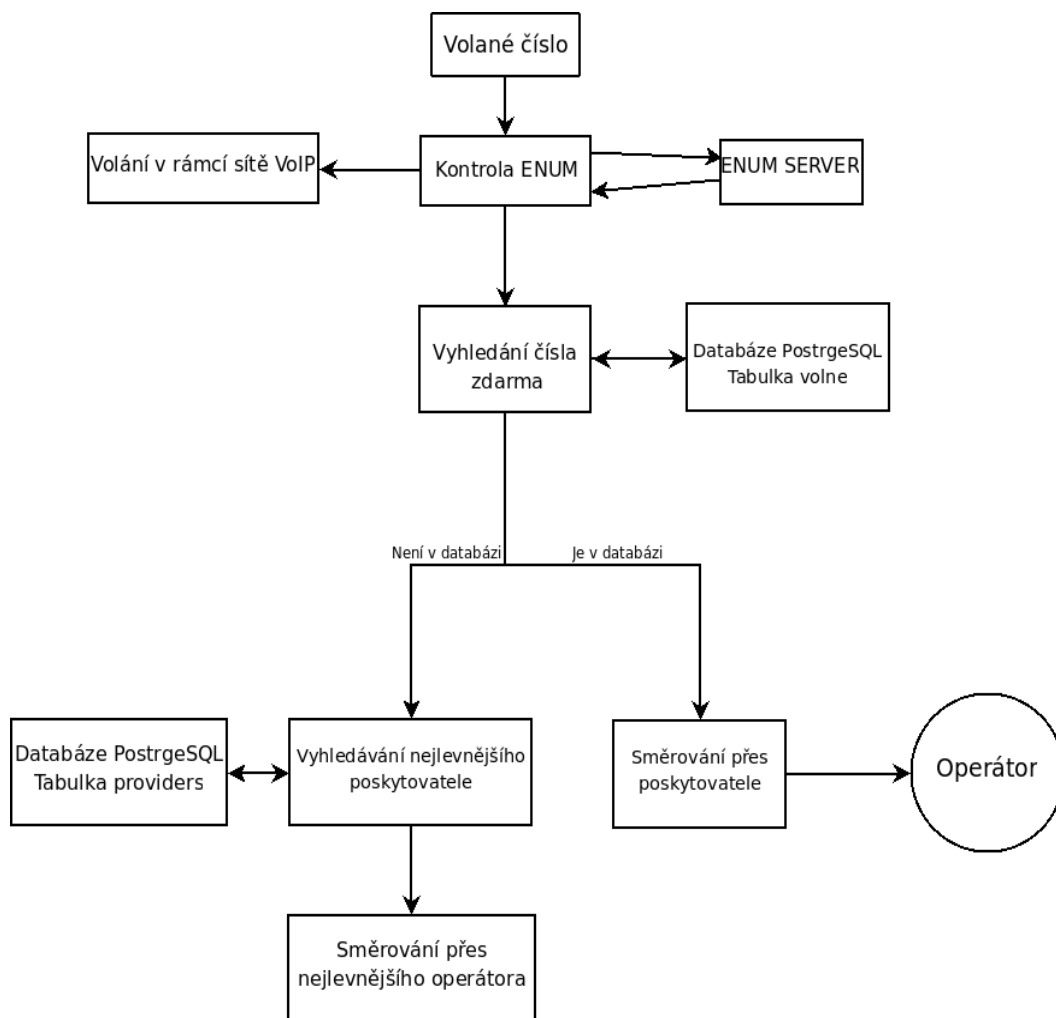
Podprogram select je zde trochu jiný, protože program posílá příkaz do databáze pro vyčtení pouze jednoho parametu „makro“ z tabulky „volne_cisla“. Tato hodnota pak je použita pro volání makra.

```
my $num = substr($cislo, 0, 3);
my $kon_operator = &select($num)|| '999';
print STDERR "1 operator je $kon_operator, cislo je $num\n";
if ($kon_operator == 999) {
    $num = substr($cislo, 0, 4);
    $kon_operator = &select($num)|| '999';
    print STDERR "predcisli je 2 mistne\n";
    print STDERR "operator je $kon_operator\n";
}
if ($kon_operator == 999){
    $num = substr($cislo, 0, 5);
    $kon_operator = &select($num)|| '999';
    print STDERR "predcisli je 4 mistne\n";
    print STDERR "operator je $kon_operator\n";
}
if ($kon_operator == 999){
    $num = substr($cislo, 0, 6);
    $kon_operator = &select($num)|| '999';
    print STDERR "predcisli je 5 mistne\n";
    print STDERR "operator je $kon_operator\n";
}
if ($kon_operator == 999){
    $num = substr($cislo, 0, 7);
    $kon_operator = &select($num)|| '999';
    print STDERR "predcisli je 6 mistne\n";
    print STDERR "operator je $kon_operator\n";
}
if ($kon_operator == 999){
    $num = substr($cislo, 0, 8);
    $kon_operator = &select($num)|| '999';
    print STDERR "predcisli je 7 mistne\n";
    print STDERR "operator je $kon_operator\n";
}
if ($kon_operator == 999){
    $num = substr($cislo, 0, 9);
    $kon_operator = &select($num)|| '999';
    print STDERR "predcisli je 8 mistne\n";
    print STDERR "operator je $kon_operator\n";
}
}
```



```
if ($kon_operator == 999){
    $num = substr($cislo, 0, 10);
    $kon_operator = &select($num)|| '999';
    print STDERR "predcisli je 9 mistne\n";
    print STDERR "operator je $kon_operator\n";
    $AGI->exec('Goto', 20);
}
else{
    print STDERR "\n vysledne cislo je $num\n";
    print STDERR "vysledny operator je $kon_operator\n";
    $AGI->exec('Playback', 'volani_zdarma');
    $AGI->exec('Macro', $kon_operator);
}
```

Výše uvedený program na vstupu upraví předčísli nejdřív na 2-místní a vyhledá jej v tabulce „volne_cisla“. Pokud se předčísli nenachází v tabulce, program přidá do proměnné \$num další číslici a zvýší předčísli na 3-místní. A pokud zase nenajde, zvýší jej na 4-místní atd. Pokud předčísli není vůbec v tabulce pak se provede příkaz AGI, který přejde na prioritu 20 té samé „extanze“. Při nalezení čísla se vypíše do konzole vyhledané předčísli a výsledný operátor. Následující zobrazuje průběh šetřícího automatu.

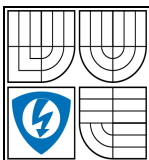


Obr. 4: Diagram zobrazující průběh šetřícího automatu

8.3 ČERNÝ SEZNAM.

Šetřící automat dokáže i blokovat nežádoucí hovory. V tabulce „blacklist“ se nacházejí data o číslech, které administrátor ústředny umístil do databáze, protože jsou nevhodná nebo se označují jako „spam“. Program nejdřív přečte data a porovná je s volacím číslem, pak hovor buď pustí nebo jej zahodí. Program je podobný programu pro hledání čísel zdarma, liší se pouze v tabulce, v které hledá a ve výsledku.

```
my $blacklist = &select($cislo) || '0';
```



```
if ($blacklist != '0'){  
  
$AGI->exec('Playback', 'volani_blacklist');  
$AGI->hangup()  
}  
else {  
$AGI->exec('Goto', '10');  
}
```

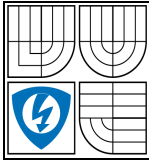
Program při nulovém výsledku z databáze, číslo v tabulce není, provede přesměrování na prioritu 10, při nenulovém přehraje účastníkovi zprávu a pak hovor zavěsí.

8.4 KONFIGURACE VOLACÍHO PLÁNU.

Volací plán se konfiguruje v souboru extensions.conf. Tento plán je připraven pro vyhledávání v VoIP čísel pomocí „enum“. Pokud číslo není VoIP, systém automaticky přejde nejdřív na vyhledávání čísla zdarma nebo pak na šetřící automat. Po vyhledávacím procesu rozhraní AGI spustí makro s hodnotou operátora, přes který je hovor směrován. V příslušném makru se pak hovor uskutečňuje. Modifikace volacího plánu se provede v následujících řádcích. Makra se importují ze souboru „makro.conf“ generovaného přes konfigurační menu. Do volacího plánu se pak přidá položka „#include makro.conf“. Soubor „makro.conf“ vypadá takto.

```
[macro-802]  
exten => s,1,NoOp  
exten => s,n,Playback(oper_802)  
exten => s,n,Dial(SIP/${MACRO_EXTEN}@802, 30)  
exten => s,n,Hangup()  
  
[macro-netphone]  
exten => s,1,NoOp  
exten => s,n,Playback(oper_netphone)  
exten => s,n,Dial(SIP/${MACRO_EXTEN}@netphone, 30)  
exten => s,n,Hangup()  
  
[macro-voipjet]  
exten => s,1,NoOp  
exten => s,n,Playback(oper_voipjet)  
exten => s,n,Dial(SIP/${MACRO_EXTEN}@voipjet, 30)  
exten => s,n,Hangup()
```

Definice makra je jednoduchá. Při zavolání makra se nejdřív nastaví priorita na 1,



pak se přehraje vzkaz uživateli, že volá do dané sítě a pak se hovor přesměruje přes účet definován v souboru „sip_automat.conf“. Pak se hovor zavěsí.

Ve volacím plánu se nejdřív provádí při zahájení hovoru nejdřív vyhledávání v enum serveru. Ve volacím plánu nastavení „enum“ pak vypadá takto.

```
[default]
exten => _420XXXXXXXXXX,1,Set(cislo=${EXTEN:3:9})
exten => _420XXXXXXXXXX,2,Set(enum=${ENUMLOOKUP(+${EXTEN},sip,,1,e164.enum.cz)})
exten => _420XXXXXXXXXX,3,GotoIf($[„${enum}“ = „“]?10:4)
exten => _420XXXXXXXXXX,4,Dial(SIP/${enum}, 30)
```

Nejdřív se do proměnné „cislo“ zapíše volané číslo, ale bez národního prefixu, pak se do proměnné „enum“ zapíše výsledek z vyhledávání „EnumLookup“. Aplikace „GotoIf“ pak porovná jestli je v databázi toto číslo. Pokud ano, přejde systém na prioritu 4 a zahájí hovor, pokud ne přejde systém na prioritu 10.

```
exten => _420XXXXXXXXXX,10,NoOp
exten => _420XXXXXXXXXX,n,AGI(volne.agi, ${cislo})
```

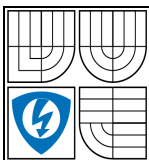
Na prioritě 10 je už spuštění rozhraní AGI a programu pro vyhledávání čísel zdarma s vstupní hodnotou 9-místního volaného čísla. Pokud se nachází v tabulce pro čísla zdarma provede se příslušné makro, pokud ne přeskočí na prioritu 20 pro šetřící automat.

```
exten => _420XXXXXXXXXX,20,NoOp
exten => _420XXXXXXXXXX,n,AGI(script.agi, ${cislo})
exten => _420XXXXXXXXXX,n,Hangup()
```

V příslušné prioritě se provede v rozhraní AGI program „script.agi“, který představuje šetřící automat. Při výskytu chyby se hovor zavěsí.

Nahrání zvukového souboru se provádí pomocí Asterisku zavoláním na číslo 1111 s parametrem, který odpovídá hodnotě „playback“ v tabulce „providers“. Je třeba ji zadat ihned po zavolání čísla 1111.

```
exten => 5555,1,Answer()
exten => 5555,n,Record(oper_${hodnota}:gsm,0,5)
exten => 5555,n,Wait(2)
exten => 5555,n,Playback(oper_${hodnota})
exten => 5555,n,Hangup()
exten => 1111,1,Answer()
exten => 1111,n,WaitExten()
exten => 1111,n,Hangup()
exten => _XX,1,NoOp
```



```
exten => _XX,n,Set(hodnota=${EXTEN})  
exten => _XX,n,Goto(default,5555,1)
```

Kontext pro příchozí hovory „incoming“ obsahuje nastavení pro filrování nežádoucích příchozích hovorů.

```
[incoming]  
exten => s,1,Set(blacklist=${CALLERID(num)})  
exten => s,n,AGI(blacklist.agi)  
exten => s,10,Dial(SIP/${EXTEN} , 30)  
exten => s,n,Hangup()
```

V souboru „sip.conf“ se zaregistrují účty pomocí příkazu „#include sip_automat.conf“ soubor generovaný přes konfigurační menu. Uživatelská jména a hesla jsou určeny operátorem. Při registraci u operátora jsou tyto údaje poslány na email. Registrace v souboru „sip_automat.conf“ se provádí následovně

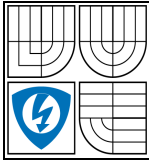
```
[netphone]  
type = friend  
username= XXXXXXXXX  
secret= XXXXXXXXX  
host= sip1.netphone.cz  
port = 5060  
context=default  
; makro = netphone  
; tabulka = prov_netphone  
; informace = www.netphone.cz
```

Položka „netphone“ je název účtu, „type“ je typ účtu, „secret“ je heslo účtu, „username“ je uživatelské jméno, „host“ je SIP server operátora, „context“ je kontext, do které účet patří.

8.5 VSTUPNÍ HODNOTY.

Šetřící automat pracuje se vstupními daty s databáze PostgreSQL, proto je třeba nastavit tabulky a sloupce s parametry. Všechny data se nacházejí v databázi „providers“. Tato databáze obsahuje hlavní tabulku „providers“, ve které jsou hlavní operátoři, jejich makra a odkazy na konkrétní tabulky, a konkrétní tabulky každého operátora. Tabulka „providers“ má hodnoty:

- id – je identifikátor operátora, typ Integer
- name – název operátora, typ Varchar
- macro – hodnota makra, typ Varchar, parametr PRIMARY KEY
- table_id – hodnota tabulky kde se nacházejí ceny předčísli, typ Varchar, parametr UNIQUE



- username – uživatelské jméno, typ Varchar
- password – heslo, typ Varchar
- sipaddr – adresa SIP serveru operátora, typ Varchar
- port – port operátora, typ Integer
- info – doplňující informace o operátorovi, typ Varchar

ID	Name	Macro	table_id	Username	Password	sipaddr	Port	Info
94	VoipJET	voipjet	prov_voipjet	hafan	hafan	sip.voipjet.cz	5060	www.voipjet.cz
96	Netphone	netphone	prov_netphone	934002790	1	sip1.netphone.cz	5060	www.netphone.cz

U parametru „macro“ a „table_id“ je ještě označení „PRIMARY KEY“ nebo „UNIQUE“, které zabraňují výskytu duplicitních hodnot v tabulce.

Každý operátor má v parametru „table_id“ odkaz na tabulku kde se nacházejí všechny předčíslí a jejich ceny daného operátora. Tato tabulka má čtyři parametry:

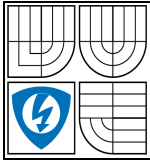
- id_num – je identifikátor pro předčíslí, typ Integer
- num – je hodnota předčíslí, typ Integer
- cena – je cena daného předčíslí, typ Float
- info – je informace komu patří předčíslí, typ Varchar

id_num	num	cena	info
114	58	1	Telefonica
115	46	1	Telefonica

Sloupec „id_num“ a „num“ obsahují i parametr, který brání zapsání toho samého čísla v jedné tabulce. Dále pak ještě databáze obsahuje tabulku prefixů zdarma a ta obsahuje parametry:

- id – je identifikátor, je přírůstková hodnota, typ Integer
- num – je hodnota předčíslí zdarma, typ Integer
- operator – je hodnota operátora, obdoba makra, typ Varchar
- info – je doplňková informace

id	num	operator	info
5	800	netphone	Bezplatna linka
6	934	netphone	Operator netphone



Dále databáze ještě obsahuje tabulku „blacklist“ pro filtrování příchozích nežádoucích hovorů. Tato tabulka má následující parametry.

- id – je identifikátor, přírůstková hodnota, typu Integer
- num – je hodnota nežádoucího čísla, typ Varchar
- info – je doplňková informace, typ Varchar

id	num	info
57	558226445	Spam
58	597445112	spam

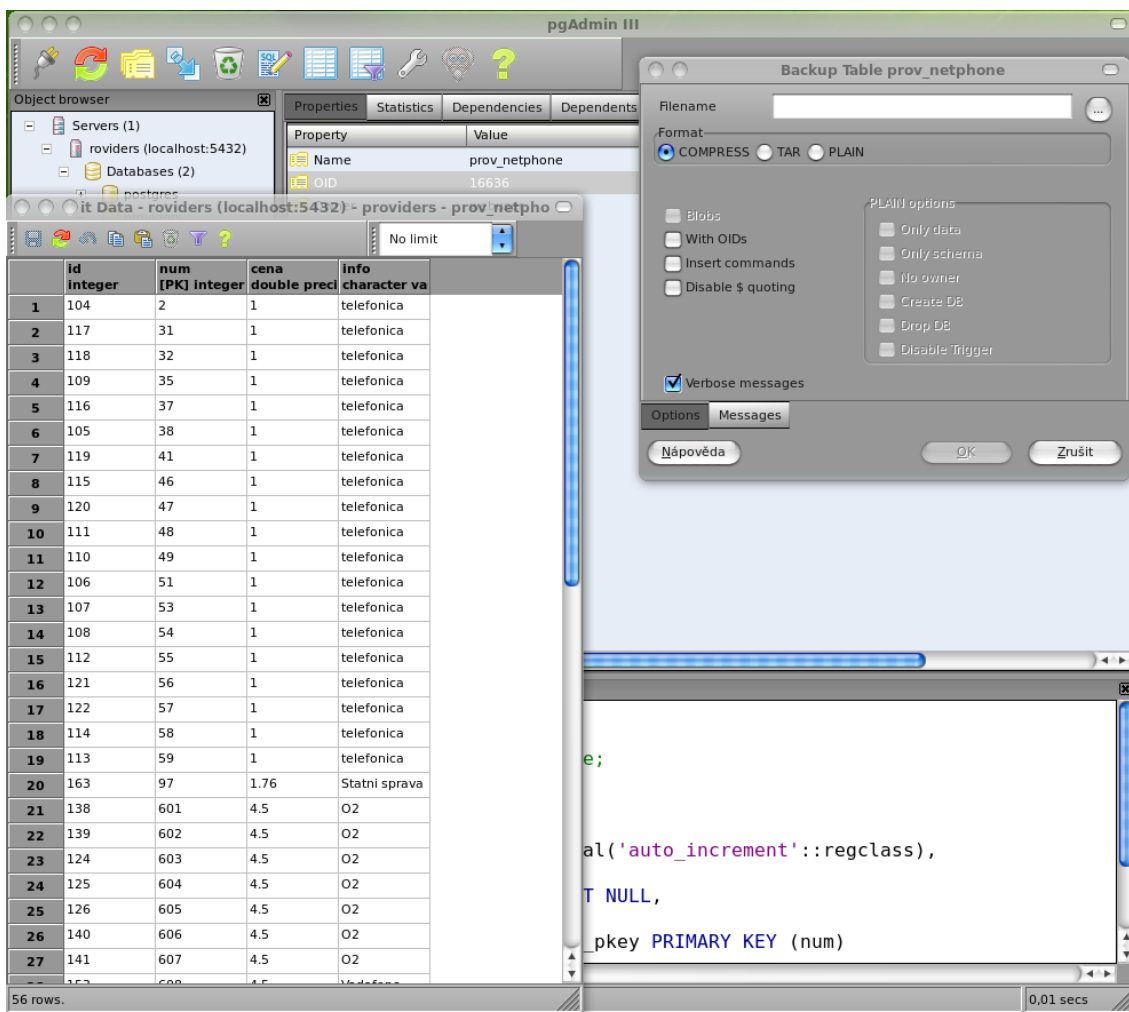
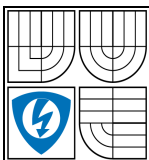
Vstupní hodnoty je lépe spravovat v databázi pomocí databázového systému, který je rychlý a méně problematický. Databázový systém pracujícího na serveru je lepší taky z důvodu, že lze psát skripty v jiných jazycích, popřípadě použít webové stránky pro doplňování a upravování cen z místa klientského počítače. Nebo lze stahovat data přímo ze serverů operátora a zapisovat pomocí programu do databáze. Záloha databáze se provádí příkazem „pg_dump“ v příkazovém řádku uživatele „postgres“ do libovolného souboru.

```
$ pg_dump providers > /tmp/providers_zaloha.dump
```

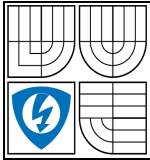
Obnova ze zálohy se pak provede následujícím příkazem.

```
$ psql providers < /tmp/providers_zaloha.dump
```

Editaci lze také provádět i jinými externími programy, např. PgAdmin. Lze v něm lehce provádět úpravy tabulek, úpravy údajů v tabulkách a zálohu databáze. Následující obrázek zobrazuje zálohu tabulky.



Obr. 5: Záloha tabulek v programu PgAdmin

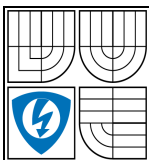


9. KONFIGURAČNÍ MENU ŠETŘÍCÍHO AUTOMATU

Konfigurační menu slouží pro nastavování šetřícího automatu. Pomocí něj lze vložit nová uživatelská jména registračních operátorů, přes které se bude hovor směřovat. Také lze nastavovat ceny předčísli a nezaplatněná čísla, pomocí kterých šetřící automat určuje nejlevnějšího operátora. Menu se skládá z několika podmenu. Je napsané v jazyce PHP a vyžaduje instalaci webového serveru na lokálním počítači. Hlavní menu je na obrázků níže: Vpravo konfiguračního menu se nachází postranní panel, pro navigaci v konfiguračním menu. Menu obsahuje položky, které odkazují na záložky s nastavením. První odkazuje na stránku pro nastavování platných čísel do tabulek vložených operátorů.



Obr. 6: Úvodní stránka konfiguračního menu



Menu obsahuje rozhraní pro vložení nového čísla, ceny a doplňkové informace, rozhraní pro změnu ceny nebo doplňkové informace a smazání čísla. U každé položky se musí vybrat tabulka operátora. Poslední položka odkazuje na výpis čísel operátora.

Konfigurace setřičiho automatu pro PBX Asterisk

Menu

- Operace s čísly
- Operace s operatory
- Císla Zdarma
- Zakazane čísla
- Vyhledávání
- Provest Zalohu
- Home

Vložení nového čísla, jeho ceny a informace

Vyberte Operatora:

Predcisli:

Cena:

Informace:

Zmena ceny nebo informace čísel

Vyberte Operatora:

Zadejte predcisli(*policko musi obsahovat hodnotu):

Zadejte cenu(nebo ponechejte prazdne):

Zadejte info(nebo ponechejte prazdne):

Smazani čísla

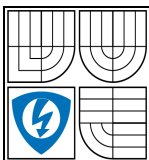
Vyberte Operatora:

Zadejte predcisli(*policko musi obsahovat hodnotu):

Vypis čísel operatora

Vyberte operatora

Obr. 7: Menu pro nastavování platných čísel



Konfigurace setřičiho automatu pro PBX Asterisk

Vypis cen predcisli operatora Netphone

Menu

Operace s cisly

Operace s operatory

Cisla Zdarma

Zakazane cisla

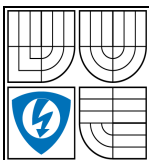
Vyhledavani

Provest Zalohu

Home

ID	Predcisli	Cena	Info
104	2	1 Kc	telefonica
105	38	1 Kc	telefonica
106	51	1 Kc	telefonica
107	53	1 Kc	telefonica
108	54	1 Kc	telefonica
109	35	1 Kc	telefonica
110	49	1 Kc	telefonica
111	48	1 Kc	telefonica
112	55	1 Kc	telefonica
113	59	1 Kc	telefonica
114	58	1 Kc	telefonica
115	46	1 Kc	telefonica
116	37	1 Kc	telefonica
117	31	1 Kc	telefonica
118	32	1 Kc	telefonica
119	41	1 Kc	telefonica
120	47	1 Kc	telefonica

Obr. 8: Výpis cen předčísí operátora



Následující stránka obsahuje položky vložení nového operátora, změna jeho údajů a smazání operátora. U položky „Název“ je třeba psát bez diakritiky a bez teček, protože slouží i pro nastavování jiných parametrů jako makro a jeho tabulku. U názvu tabulek v PostgreSQL se nesmí vyskytovat tečky. Poslední položka odkazuje na výpis operátorů.

Konfigurace setřicího automatu pro PBX Asterisk

Menu

- Operace s čísly
- Operace s operátory
- Čísla Zdarma
- Zakazane čísla
- Vyhledávání
- Provest Zalohu
- Home

Vložení nového operátora

Zadejte všechny položky:

Název(*bez teček a diakritiky):

Uzivateleske jmeno uctu:

Heslo:

Adresa SIP serveru:

Port(cislo - default 5060):

Web Adresa

Hudebni soubor(cislo)

Zmena udaju operátora

Zadejte všechny položky:

Netphone Uzivatelske jmeno

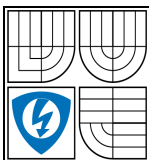
Smazani operátora

Vyberte operátora, ktereho chcete smazat:

Netphone

Vypis operátoru

Obr. 9: Stránka pro konfiguraci operátorů



Konfigurace setřičiho automatu pro PBX Asterisk

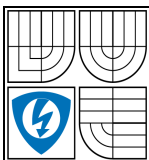
Vypis operatoru definovanych v setřicim automatu

ID	Nazev operatora	Makro	Tabulka v databazi	Uzivatelске jmeno	Heslo	Adresa serveru	Port	Web adresa
94	VoipJET	voipjet	prov_voipjet	hafan	hafan	sip.adresa.cz	5432	www.voipjet.cz

Menu

- Operace s cisly
- Operace s operatory
- Cisla Zdarma
- Zakazane cisla
- Vyhledavani
- Provest Zalohu
- Home

Obr. 10: Výpis operátorů v šetřícím automatu



Následující stránka zobrazuje nastavení bezplatných čísel a předčísli VoIP operátorů, kterým lze volat zdarma. Stránka obsahuje položky pro vložení, editaci, smazání a výpis bezplatných čísel.

Konfigurace setriciho automatu pro PBX Asterisk

!!! Pozor jste na strance definovani volnych cisel !!!

Vlozeni noveho cisla a operatora

Vyberte Operatora:

Predcisli:

Informace:

Zmena ceny nebo informace cisel

Zadejte predcisli(*policko musi obsahovat hodnotu):

Zadejte info(nebo ponechejte prazdne):

Vyberte operatora, kteremu chcete predcisli priradit:

Smazani cisla

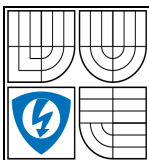
Vyberte cislo zdarma:

Kliknutim vypisete vsechna cisla zdarma

Menu

- Operace s cisly
- Operace s operatory
- Cisla Zdarma
- Zakazane cisla
- Vyhledavani
- Provest Zalohu
- Home

Obr. 11: Menu pro konfiguraci bezplatných čísel



Šetřící automat je schopen i určovat čísla příchozích hovorů a ty pak filtrovat. Pro tuto funkci je definovaná tabulka černý seznam - „blacklist“ v PostgreSQL, kde se nacházejí všechny nežádoucí čísla. Konfigurace se provádí na následující stránce.

Lze vkládat čísla, mazat při chybě a vypisovat z tabulky „blacklist“. Šetřící automat čte hodnoty a na základě nich pak příchozí hovor odhodí.

Konfigurace setřičiho automatu pro PBX Asterisk

Menu

- Operace s čísly
- Operace s operatory
- Čísla Zdarma
- Zakazane čísla
- Vyhledávání
- Provest Zalohu
- Home

Vlozte do blacklist

Cislo: +420

Duvod vlozeni:

Vlozit

Smazte z blacklist

558455621 Smazat

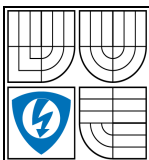
Vypis cisel z blacklist

Provest

Vypis cisel z blacklist

ID	Cislo	Z duvodu
91	+420 558455621	spam

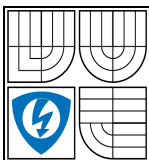
Obr. 12: Konfigurace černého seznamu



Konfigurační menu také umožňuje vyhledávat čísla ve všech tabulkách.

The screenshot displays the configuration interface for an Asterisk PBX. At the top, a black header contains the text "Konfigurace setřičiho automatu pro PBX Asterisk". Below this is a grey sidebar menu with the title "Menu" and several options: "Operace s čísly", "Operace s operatory", "Čísla Zdarma", "Zakazane čísla", "Vyhledavani", "Provest Zalohu", and "Home". The "Vyhledavani" option is selected, and the main content area shows the search interface. It includes the heading "Vyhledavani" and a form with the label "Vlozte cislo:" followed by the number "+420" and an empty input field. To the right of the input field is a button labeled "Vyhledej".

Obr. 13: Vyhledávání čísel v šetřícím automatu

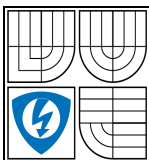


Záloha ze systému se provádí v konfiguračním menu pod odkazem „Provest Zálohu“. Data se exportují do externího souboru. Následující obrázek zobrazuje provedení zálohy.



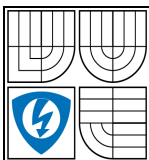
Obr. 14: Provedení zálohy systému.

Konfigurační menu zapisuje také do souborů „sip_automat.conf“ a „makro.conf“ data z databáze ve speciálním formátu odpovídajícím pro ústřednu Asterisk. V souboru „sip_automat.conf“ jsou uloženy informace o uživatelích, heslech a adresy sip serverů operátora. V souboru „makro.conf“ jsou definované makra, ve kterých se provádí přesměrování. Tyto soubory se pak vloží do volacího plánu a do souboru pro konfiguraci účtů příkazem „#include“. Zvukový soubor se označí v položce operace s operátory, nahraje se pak prostřednictvím Asterisku.



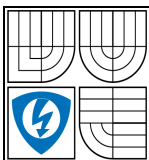
10. ZÁVĚR

Pomocí služby Enum, lze volat v rámci sítě VoIP zdarma s použitím čísla ve formátu mezinárodního číslování. Tímto způsobem je možno propojit ústředny bez použití třetí strany, tzn. poskytovatele. Šetřící automat ve volacím plánu podle volaného čísla je schopen určit nejlevnější cenu a tak nejlevnějšího operátora. Pak přes něj přeměruje hovor a volajícímu účastníkovi pak sdělit cenu hovoru. Pro větší počet uživatelů (nad 500) je výhodné spojit ústřednu přímo se serverem větší společnosti, která je schopná přiřazovat účty dynamicky a disponuje vysokým výpočetním výkonem. Společnost pak může přepojit více hovorů najednou. Šetřící automat je přizpůsoben pro dynamickou změnu cen operátorů a tímto je velice flexibilní. Jeho celý zdrojový kód s komentářem je uveden v příloze A.



11. SEZNAM POUŽITÉ LITERATURY

- [1] MEGGELEN, J.V, Smith, J., Madsen, L., Asterisk™ The Future of Telephony. Sevastopol: O'Reilly Media, Inc., 2005. ISBN 0-596-00962-3.
- [2] BAZALA, D., Telekomunikace a VoIP telefonie I. Praha: BEN – technická literatura, Praha 2006, ISBN 80-7300-201-9.
- [3] WALACE, Z., VoIP bez předchozích znalostí. Brno: Computer Press, 2007. 232 s. ISBN 978-80-251-1458-2.
- [4] VOZŇÁK, M., Signalizace SIP. Teorie a praxe IP telefonie. 2006, č. 2, s. 35-75.
- [5] TROLLER, P., Číslování a adresace telefonních VoIP sítí. Teorie a praxe IP telefonie. 2006, č. 2, s. 117-129.
- [6] TROLLER, P., Číslování a adresování v klasických a IP telefonních sítích. Teorie a praxe IP telefonie. 2004, č. 1, s. 73-81.
- [7] PETERKA, J., Co je ENUM? [online]. 2006 [cit. 2006-09-22]. Dostupný z WWW: <<http://www.lupa.cz/clanky/co-je-enum/>>.
- [8] MOMJIAN, B., PostgreSQL, Praktický průvodce. Brno: Computer Press, Brno 2003. ISBN 80-7226-954-2.
- [9] KYSELA, M., Perl. Praha: Grada Publishing, Praha 2005. ISBN 80-247-1170-2
- [10] LUBOSLAV, L., PHP a MySQL, Hotová řešení. Brno: Computer Press, Brno 2005. ISBN 80-251-0397-8



12. PŘÍLOHA A. - ZDROJOVÝ KÓD ŠETŘÍČÍHO AUTOMATU

12.1 APLIKACE PRO VYHLEDÁNÍ NEJLEVNĚJŠÍ CENY

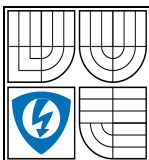
```
#!/usr/bin/perl -w

# Definice balicku a jinych rozhrani
use strict;
use DBI;
use Asterisk::AGI;

# nacteni dat z Asterisku, hlavne vytocene cislo
my $AGI = new Asterisk::AGI;
my %input=$AGI->ReadParse();
my $cislo = $ARGV[0];

# pripojeni k databazi providers do promenne $dbh
my $dbh = DBI->connect("DBI:Pg:dbname=providers;host=localhost", "webuser",
"webuser", {'RaiseError'=>1,'AutoCommit'=> 1});
#####
#####
# Nacteni poctu operatoru z tabulky providers do promenne $count
my $hash_count = $dbh->prepare("SELECT COUNT(id) FROM providers");
$hash_count->execute();
my $count;
$hash_count->bind_columns(\$count);
$hash_count->fetch();
#####
#####
# vytvoreni podprogramu pro ctieni dat z databaze, zamezeni cykleni, vyhodi vzdy 1
vysledek
# vola se &select se ctiry parametry $m volany parametr, $n je tabulka, $l je
porovnavany parametr
# a $k je porovnavaci hodnota
sub select {
    my $m = $_[0];
    my $n = $_[1];
    my $l = $_[2];
    my $k = $_[3];
    my $hashe = $dbh->prepare("SELECT $m FROM $n WHERE $l = $k");
    $hashe->execute();
    my $vysledek;
    $hashe->bind_columns(\$vysledek);
    $hashe->fetch;
    return ($vysledek);
}
#####
#####
# Vyceteni ceny a makra operatora, definice promennych
my $i;
my @cena;
my @makro;
my $num2 = substr($cislo, 0, 3); #promenna num je dvojmistni predcisli
my $num3 = substr($cislo, 0, 4); #promenna numa je trimistni predcisli
my $num4 = substr($cislo, 0, 5); #promenna numa je ctymistni predcisli
my $num5 = substr($cislo, 0, 6); #promenna numa je petimistni predcisli
my $num6 = substr($cislo, 0, 7); #promenna numa je sestimistni predcisli
my $num7 = substr($cislo, 0, 8); #promenna numa je sedmimistni predcisli
my $num8 = substr($cislo, 0, 9); #promenna numa je osmimistni predcisli

my $semp = $dbh->selectall_arrayref("SELECT table_id, macro FROM providers",
```



```
{ Slice => {} });

my @table_id;
my @makro_id;
foreach my $emp ( @$emps ) {
    push(@table_id, $emp->{table_id});
    push(@makro_id, $emp->{macro});
}

# Nacteni ceny predcisli kazdeho operatora do pole cena, nacteni makra kazdeho
operatora do pole makra
for ($i = 0; $i<$count; $i++){
    my $cena_sql;
    my $makro_sql;

    my $table_sql = $table_id[$i];
    $makro_sql = $makro_id[$i];

    # pokud se nenajde dvojmistni predcisli v tabulce prejde se na trimistni

    $cena_sql = &select('cena', $table_sql, 'num', $num2) || '7777';

    if ($cena_sql == '7777') {

        # print STDERR "\n predcisli po if 7777: $num\n";
        $cena_sql = &select('cena', $table_sql, 'num', $num3) || '7777';
    }
    if ($cena_sql == '7777') {

        # print STDERR "\n predcisli po if 7777: $num\n";
        $cena_sql = &select('cena', $table_sql, 'num', $num4) || '7777';
    }
    if ($cena_sql == '7777') {

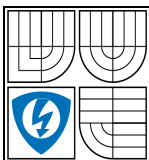
        # print STDERR "\n predcisli po if 7777: $num\n";
        $cena_sql = &select('cena', $table_sql, 'num', $num5) || '7777';
    }
    if ($cena_sql == '7777') {

        # print STDERR "\n predcisli po if 7777: $num\n";
        $cena_sql = &select('cena', $table_sql, 'num', $num6) || '7777';
    }
    if ($cena_sql == '7777') {

        # print STDERR "\n predcisli po if 7777: $num\n";
        $cena_sql = &select('cena', $table_sql, 'num', $num7) || '7777';
    }
    if ($cena_sql == '7777') {

        # print STDERR "\n predcisli po if 7777: $num\n";
        $cena_sql = &select('cena', $table_sql, 'num', $num8) || '9999';
    }

    if ($cena_sql != '9999'){
        push (@cena, $cena_sql);
        push (@makro, $makro_sql);
        print STDERR "\n tabulka: $table_sql, cena: $cena_sql makro: $makro_sql\n";
    }
}
#####
#####
# Definice promennych a vyhledavaci proces
my $min;
my $kon_makro;
my $delka = @cena;
my $a = $cena[0];
my $b;
```



```
my @pole;

# do pole se prida cena a za ni hned makro, do promenne a se zapise prvni prvek z
pole cena
for ($i = 0; $i < $delka; $i++){
    push(@pole, $cena[$i]);
    push(@pole, $makro[$i]);
}

# vyhledavaci proces, vrati promennou $min s nejnijszi cenou
for ($i = 0; $i < $delka; $i++){
    $b = $cena[$i];
    if ($a == $b){
        $min = $a;
    }
    else{
        if ($a < $b){
            $min = $a;
        }
        else{
            $min = $b;
            $a = $b;
        }
    }
}

# Vyhledavaci proces pro vyhledani makra, porovna kazdy prvek z pole pole s
hodnotou promenne $min
$delka = @pole;
for ($i = 0; $i < $delka; $i++){
    if ($pole[$i] =~ /$min/){
        $kon_makro = $pole[$i+1];
    }
}

my $say_cena = $min;
my $say_a = substr($say_cena, 0, 1);
my $say_b = substr($say_cena, 2, 2);

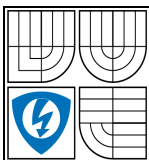
# Vypis vysledku do konzole CLI a provedeni makra operatora
print STDERR "\n vystup z asterisku $cislo, a koncove makro je $kon_makro\n\n\n";
$AGI->exec('Playback', 'cena_je');
$AGI->exec('SayDigits', $say_a);
$AGI->exec('Playback', 'korun');
$AGI->exec('Playback', 'a');
$AGI->exec('SayDigits', $say_b);
$AGI->exec('Playback', 'haleru');
$AGI->exec('Playback', 'volate_pres');
$AGI -> exec('Macro', $kon_makro);
```

12.2 ZDROJOVÝ KÓD APLIKACE PRO VYHLEDÁVÁNÍ ČÍSEL ZDARMA

```
#!/usr/bin/perl -w

# Definice balicku a jinych rozhrani
use strict;
use DBI;
use Asterisk::AGI;

# nacteni dat z Asterisku, hlavne vytocene cislo
my $AGI = new Asterisk::AGI;
my %input = $AGI->ReadParse();
my $cislo = $ARGV[0];
```

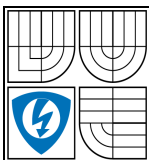


```
# pripojeni k databazi providers do promenne $dbh
my $dbh = DBI->connect("DBI:Pg:dbname=providers;host=localhost",
"postgres","postgres", {'RaiseError' => 1, 'AutoCommit'=> 1});

#####
#####
# vytvoreni podprogramu pro cteni dat z databaze, zamezeni cykleni, vyhodi vzdy 1
vysledek
# vola se &select se ctiry parametry $m volany parametr, $n je tabulka, $l je
porovnavany parametr
# a $k je porovnavaci hodnota
sub select {
    my $m = $_[0];
    my $hashe = $dbh->prepare("SELECT operator FROM volne_cisla WHERE num = $m");
    $hashe->execute();
    my $vysledek;
    $hashe->bind_columns(\$vysledek);
    $hashe->fetch;
    return($vysledek);
}

# Prohledavani tabulky pro cislo $num, pro vsechny predcisli.
print STDERR "Prohledavani cisla...prosim cekejte...\n\n";

my $num = substr($cislo, 0, 3);
my $kon_operator = &select($num)|| '999';
print STDERR "1 operator je $kon_operator, cislo je $num\n";
if ($kon_operator == 999) {
    $num = substr($cislo, 0, 4);
    $kon_operator = &select($num)|| '999';
    print STDERR "predcisli je 2 mistne\n";
    print STDERR "operator je $kon_operator\n";
}
if ($kon_operator == 999){
    $num = substr($cislo, 0, 5);
    $kon_operator = &select($num)|| '999';
    print STDERR "predcisli je 4 mistne\n";
    print STDERR "operator je $kon_operator\n";
}
if ($kon_operator == 999){
    $num = substr($cislo, 0, 6);
    $kon_operator = &select($num)|| '999';
    print STDERR "predcisli je 5 mistne\n";
    print STDERR "operator je $kon_operator\n";
}
if ($kon_operator == 999){
    $num = substr($cislo, 0, 7);
    $kon_operator = &select($num)|| '999';
    print STDERR "predcisli je 6 mistne\n";
    print STDERR "operator je $kon_operator\n";
}
if ($kon_operator == 999){
    $num = substr($cislo, 0, 8);
    $kon_operator = &select($num)|| '999';
    print STDERR "predcisli je 7 mistne\n";
    print STDERR "operator je $kon_operator\n";
}
if ($kon_operator == 999){
    $num = substr($cislo, 0, 9);
    $kon_operator = &select($num)|| '999';
    print STDERR "predcisli je 8 mistne\n";
    print STDERR "operator je $kon_operator\n";
}
}
# Pokud se cislo nenachazi v tabulce provede se context s prioritou 20
if ($kon_operator == 999){
    $num = substr($cislo, 0, 10);
    $kon_operator = &select($num)|| '999';
}
```



```
print STDERR "predcisli je 9 mistne\n";
print STDERR "operator je $kon_operator\n";
$AGI->exec('Goto', 20);
}

# Provedeni Makra
else{
print STDERR "\n vysledne cislo je $num\n";
print STDERR "vysledny operator je $kon_operator\n";
$AGI->exec('Playback', 'volate_zdarma');
$AGI->exec('Playback', 'volate_pres');
$AGI->exec('Macro', $kon_operator);
}
```

12.3 APLIKACE PRO BLOKOVÁNÍ PŘÍCHOZÍCH HOVORŮ

```
#!/usr/bin/perl -w

# Definice balicku a jinych rozhrani
use strict;
use DBI;
use Asterisk::AGI;

# pripojeni k databazi providers do promenne $dbh
my $dbh = DBI->connect("DBI:Pg:dbname=providers;host=localhost", "postgres",
"postgres", {'RaiseError'=>1,'AutoCommit'=> 1});

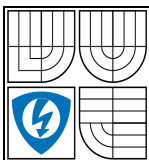
# nacteni dat z Asterisku, hlavne vytocene cislo
my $AGI = new Asterisk::AGI;
my %input=$AGI->ReadParse();
my $cislo = $ARGV[0];

#####
#####
# vytvoreni podprogramu pro cteni dat z databaze, zamezeni cykleni, vyhodi vzdy 1
# vysledek
# vola se &select se ctymi parametry $m volany parametr, $n je tabulka, $l je
# porovnavany parametr
# a $k je porovnavaci hodnota
sub select {
my $m = $_[0];
my $hashe = $dbh->prepare("SELECT id FROM blacklist WHERE num = $m");
$hashe->execute();
my $vysledek;
$hashe->bind_columns(\vysledek);
$hashe->fetch;
return($vysledek);
}

my $blacklist = &select($cislo) || '0';
if ($blacklist != '0'){
$AGI->exec('Playback', 'volani_blacklist');
$AGI->hangup()
}
else {
$AGI->exec('Goto', '10');
}
```

12.4 INSTALACE ŠETŘÍČÍHO AUTOMATU

Vytvoření databáze „providers“ příkazem:



```
$ createdb providers -U postgres
```

Vytvoření uživatele pro šetřící automat:

```
providers=# CREATE USER asterisk WITH PASSWORD asterisk;
```

Vytvoření uživatele webuser:

```
providers=# CREATE USER webuser WITH PASSWORD webuser;
```

Povolení práv uživateli webuser:

```
providers=# GRANT ALL ON providers TO webuser;
```

Nastavení ručně:

Vytvoření sekvence auto_increment:

```
CREATE SEQUENCE auto_increment;
```

Vytvoření tabulky providers:

```
CREATE TABLE providers (  
    id INTEGER DEFAULT nextval('auto_increment'),  
    name VARCHAR,  
    macro VARCHAR PRIMARY KEY,  
    table_id VARCHAR UNIQUE,  
    username VARCHAR,  
    password VARCHAR,  
    sipaddr VARCHAR,  
    port INTEGER,  
    info VARCHAR);
```

Vytvoření tabulky volne:

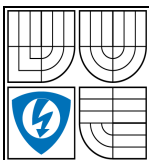
```
CREATE TABLE volne (  
    id INTEGER DEFAULT nextval('auto_increment'),  
    num INTEGER UNIQUE,  
    operator VARCHAR,  
    info VARCHAR);
```

Vytvoření tabulky blacklist:

```
CREATE TABLE blacklist (  
    id INTEGER DEFAULT nextval('auto_increment'),  
    num INTEGER UNIQUE,  
    info VARCHAR);
```

Zkopírování konfiguračního menu ze složky „Konfigurační menu“ do složky „/var/www/“. Pak v prohlížeči stačí napsat „localhost“.

Nastavení asterisku:



Do volacího plánu vložit následující řádky:

```
[default]
exten => _XXXXXXXX,1,Set(cislo=${EXTEN:3:9})
exten => _XXXXXXXX,2,Set(enum=${ENUMLOOKUP(+${EXTEN},sip,,1,e164.enum.cz)})
exten => _XXXXXXXX,3,GotoIf("${enum}" = ""?10:4)
exten => _XXXXXXXX,4,Dial(SIP/${enum}, 30)

exten => _XXXXXXXX,10,NoOp
exten => _XXXXXXXX,n,AGI(volne.agi, ${cislo})

exten => _XXXXXXXX,20,NoOp
exten => _XXXXXXXX,n,AGI(script.agi, ${cislo})
exten => _XXXXXXXX,n,Hangup()

[incoming]
exten => s,1,Set(blacklist=${CALLERID(num)})
exten => s,n,AGI(blacklist.agi)

exten => s,10,Dial(SIP/${EXTEN} , 30)
exten => s,n,Hangup()
```

a vložení řádku:

```
#include „/var/www/macro.conf“
```

Do souboru pro konfigurování uživatelů se vloží příkaz:

```
#include „/var/www/sip_automat.conf“
```

Nastavení pomocí programu instalace.pl (v příkazovém řádku):

```
$ sudo perl instalace.pl
```