

Integrace VOIP serveru Asterisk s analogovou telefonní sítí

Integration of the Asterisk VOIP server with Analogue Telephone Services

Bc. Martin Ludík

Diplomová práce
2012



Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně

Fakulta aplikované informatiky

akademický rok: 2011/2012

ZADÁNÍ DIPLOMOVÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení: **Bc. Martin LUDÍK**

Osobní číslo: **A09512**

Studijní program: **N 3902 Inženýrská informatika**

Studijní obor: **Informační technologie**

Téma práce: **Integrace VOIP serveru Asterisk s analogovou telefonní sítí**

Zásady pro vypracování:

1. Analyzujte možnosti integrace ústředny Asterisk s analogovou telefonní sítí.
2. Zaměřte se zejména na AMI, AGI (Asterisk media/gateway interface), propojení přes protokoly E1 (PRI, BRI, VoIP) a využití Asterisk DB.
3. Navrhněte způsob implementace pokročilých nových služeb pobočkové telefonní ústředny Asterisk, připojené k analogové síti, např. editovatelný seznam telefonních čísel, který umožňuje zobrazení jména místo čísla na VoIP terminálech, přímé vytáčení (kliknutím na číslo v seznamu v prohlížeči), zpětné volání (call back) atd.
4. Implementujte sadu nových funkcí pobočkové ústředny tak, aby byly dostupné i na původních analogových telefonech.

Rozsah diplomové práce:

Rozsah příloh:

Forma zpracování diplomové práce: **tištěná/elektronická**

Seznam odborné literatury:

1. **SIMIONOVICH, Nir. Asterisk gateway interface 1.4 and 1.6 programming: design and develop Asterisk-based VoIP telephony platforms and services using PHP and PHPAGI. 1st ed. Birmingham: Packt Publishing, 2009, 200 s. ISBN 978-184-7194-466.**
2. **VOŽŇÁK, Miroslav. Voice over IP. 1. vyd. Ostrava: VŠB – Technická univerzita Ostrava, 2008, 176 s. ISBN 978-802-4818-283.**
3. **VOŽŇÁK, Miroslav. Spojovací systémy. 1. vyd. Ostrava: Vysoká škola báňská – Technická univerzita Ostrava, 2009, 196 s. ISBN 978-802-4819-617.**
4. **Asterisk: the definitive guide. 3rd ed. Sebastopol, CA: OReilly Media, Inc. ISBN 978-059-6517-342.**
5. **WINTERMEYER, Stefan a Stephen BOSCH. Practical Asterisk 1.4 and 1.6. Upper Saddle River: Addison-Wesley, 2009, 793 s. ISBN 978-032-1525-666.**
6. **DIGIUM, Inc. Dashboard: Asterisk Project [online]. Huntsville, [2011] [cit. 2012-01-24]. Dostupné z: <https://wiki.asterisk.org/wiki/dashboard.action>**

Vedoucí diplomové práce:

Ing. Tomáš Dulík

Ústav informatiky a umělé inteligence

Datum zadání diplomové práce:

24. února 2012

Termín odevzdání diplomové práce:

21. května 2012

Ve Zlíně dne 24. února 2012



prof. Ing. Vladimír Vašek, CSc.
děkan



doc. Mgr. Roman Jašek, Ph.D.
ředitel ústavu

ABSTRAKT

Práce volně rozšiřuje bakalářskou práci autora z roku 2009 a představuje další možnosti využití softwarové ústředny Asterisk. Zaměřuje se především na popis a využití moderních telefonních služeb VoIP s možností integrace do starších TDM systémů, propojení ISDN a VoIP telefonie atd. Praktická část práce se zaměřuje na jednu z mnoha možností, především na ovládání telefonních služeb přes webové rozhraní.

Klíčová slova:

Asterisk, internetová telefonie, VoIP, webové služby, ISDN, telefonní ústředna, SIP, telefonní brána, AMI, AGI.

ABSTRACT

This work expands bachelor thesis written by the author in 2009, and introduces additional possibilities of usage of Asterisk PBX software. It focuses primarily on the description and use of advanced VoIP telephony with possibilities of integration with older TDM systems, as well as connecting ISDN and VoIP telephony and so on. The practical part of this work is focused on one of many prospects, especially on the control of telephone services thru the Web.

Keywords:

Asterisk, internet telephony, VoIP, Web services, ISDN, PBX, SIP telephone gateway, AMI, AGI.

Děkuji Ing. Tomáši Dulíkovi za vedení této práce.

Děkuji své rodině, jmenovitě Pavlíně, Sebastianovi, Stele a Sofii, za čas, který mi pro psaní této práce a celé studium poskytli.

Prohlašuji, že

- beru na vědomí, že odevzdáním diplomové/bakalářské práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že diplomová/bakalářská práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk diplomové/bakalářské práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně a jeden výtisk bude uložen u vedoucího práce;
- byl/a jsem seznámen/a s tím, že na moji diplomovou/bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – diplomovou/bakalářskou práci nebo poskytnout licenci k jejímu využití jen s předchozím písemným souhlasem Univerzity Tomáše Bati ve Zlíně, která je oprávněna v takovém případě ode mne požadovat přiměřený příspěvek na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše);
- beru na vědomí, že pokud bylo k vypracování diplomové/bakalářské práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky diplomové/bakalářské práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem diplomové/bakalářské práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji,

- že jsem na diplomové práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně

.....
podpis diplomanta

OBSAH

ÚVOD	9
I TEORETICKÁ ČÁST	10
1 PBX - ÚSTŘEDNA	11
1.1 HISTORIE ÚSTŘEDEN.....	11
2 VEŘEJNÁ TELEFONNÍ SÍŤ PSTN	12
2.1 ROZHRANÍ TELEFONNÍCH SÍTÍ.....	12
2.2 SIGNALIZACE	13
2.2.1 Rozdělení signalizací	13
2.3 ISDN – ZÁKLADNÍ VLASTNOSTI	14
2.3.1 Základní přípojka BRI.....	14
2.3.2 Rozšířená přípojka PRI – E1	14
2.3.3 ISDN se signalizací DSS1 v modelu OSI.....	15
2.3.4 Některé signalizační zprávy DSS1	16
2.3.5 Služby v ISDN	18
2.4 SIGNALIZACE QSIG A SS7	19
2.5 SIGNALIZAČNÍ PROTOKOL SIP.....	19
2.5.1 SIP základní vlastnosti	20
2.5.2 Prvky SIP	21
2.5.3 SIP servery	21
2.5.4 Požadavky a odpovědi protokolu SIP	22
2.5.5 Typy požadavků a opovědí	23
2.5.6 SIP transakce a dialog	24
2.6 SPOJENÍ Z VOIP PŘES ISDN NA ANALOGOVÝ TERMINÁL S U-ROZHRANÍM.....	25
3 NEJEN TELEFONNÍ ÚSTŘEDNA ASTERISK	27
3.1 VLASTNOSTI ASTERISKU	27
3.2 ARCHITEKTURA ASTERISKU	29
3.2.1 Kanál SIP	30
3.2.2 Kanál DAHDI	30
3.2.3 Kanál LOCAL	30
3.3 DIALPLAN	31
3.3.1 Kontext.....	31
3.3.2 Extension.....	32
3.3.3 Priority.....	33
3.3.4 Aplikace	33
3.3.5 Řetězce a porovnání vzorů v dialplanu	33
3.3.6 Makra v dialplanu	34
3.4 ASTERISK DATABÁZE ASTDB	35
3.4.1 Ukládání dat do databáze	35
3.4.2 Načtení dat z databáze.....	36
3.4.3 Mazání dat databáze	36
3.5 ASTERISK MANAGER INTERFACE AMI.....	36
3.5.1 Připojení AMI pomocí protokolu TCP	37
3.5.2 AMI správce zpráv	37

3.6	ASTERISK GATEWAY INTERFACE AGI	38
3.6.1	Základní varianta AGI.....	39
3.6.2	EAGI (Enhanced AGI).....	40
3.6.3	DeadAGI	40
3.6.4	FastAGI AGI přes protokol TCP	40
3.6.5	Asynchronní AGI kontrolované AMI	40
3.7	TAB. 3-20 - ASYNC AGI—AMI-CONTROLLED AGIINTEGRACE RELAČNÍCH DATABÁZÍ.....	40
3.8	WEBOVÉ SLUŽBY PRO SPOLUPRÁCI S ÚSTŘEDNOU ASTERISK	41
3.8.1	Jazyk HTML	41
3.8.2	Jazyk PHP	41
3.9	ASTERISK WEB INTERFACE	41
3.10	DALŠÍ OPEN VOIP SIP PROJEKTY	42
II	PRAKTICKÁ ČÁST	43
4	INTEGRACE VOIP SERVERU.....	44
4.1	POPIS ZAPOJENÍ SYSTÉMU.....	44
4.2	STAVEBNÍ PRVKY SYSTÉMU NUA	46
4.3	WEBOVÉ SLUŽBY, WEBOVÉ ROZHRAŇÍ	46
4.3.1	Integrovaný telefonní seznam	47
4.3.1.1	Struktura dat v AstDB telefonního seznamu	48
4.3.1.2	Zobrazení dat telefonního seznamu	48
4.3.1.3	Uložení dat telefonního seznamu.....	52
4.3.1.4	Editace dat telefonního seznamu	54
4.3.1.5	Mazání dat telefonního seznamu	54
4.3.2	Web telefon	56
4.3.3	Zpětné volání - Call back	59
4.3.4	Speciální konfigurační soubor.....	59
4.4	PŘÍKLAD INTEGRACE SYSTÉMU I – ANALOGOVÝ UŽIVATEL	60
4.5	PŘÍKLAD INTEGRACE SYSTÉMU II – VOIP UŽIVATEL	61
	ZÁVĚR	63
	ZÁVĚR V ANGLIČTINĚ.....	64
	SEZNAM POUŽITÉ LITERATURY.....	65
	SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK.....	67
	SEZNAM OBRÁZKŮ	69
	SEZNAM TABULEK.....	70
	SEZNAM PŘÍLOH.....	72

ÚVOD

Dnešní doba přináší neuvěřitelně rychlý vývoj všech technologií. Tento pokrok se nevyhnul ani telekomunikacím. Tak jako digitalizace byla určitý milník evoluce telefonie, tak i VoIP představuje podobný krok. VoIP telefonie, díky její otevřenosti a dostupnosti, má nesporně mnoho výhod, protože jako přenosové medium používá celosvětovou síť Internet. Nevýhodou této otevřenosti je zatím její zranitelnost.

Vývoj VoIP telefonie byl tak rychlý, že při nasazování telefonních systémů vyšší generace založené na TDM ho novější technologie VoIP předběhla. Nahrazení stávajícího systému systémem TDM je proto ekonomicky nevýhodné. Tato práce si dala za cíl popis integrace nových pokročilých služeb, které bezesporu VoIP přináší, do telefonních systémů vývojově starších.

Práce je dělena na dvě části. Teoretická část popisuje principy přepínaných telefonních služeb, signalizací, protokolu SIP a telefonní VoIP ústředny Asterisk. Rozsáhlé softwarové možnosti telefonní ústředny Asterisk jsou také rozvedeny v teoretické části. V podstatě jsou v této části popsány stavební prvky integrace systému, použité v praktické části.

Praktická část seznamuje již s konkrétním řešením. Sestavuje jednotlivé prvky popsané v první části, kde popisuje principy komunikace a ukazuje, jakými programovými kroky lze toho dosáhnout.

Název „analogová telefonní síť“ v názvu práce nesignalizuje jen typický analogový systém, ale představuje zde celý telefonní systém, který má vedle rozhraní analogového i digitální. Jde o systémy nižší generace než je VoIP řešení.

I. TEORETICKÁ ČÁST

1 PBX - ÚSTŘEDNA

PBX (Private Branch Exchange) je telefonní pobočková ústředna. V dřívějších dobách propojoval tento systém telefonní přístroje společnosti s veřejnou telefonní sítí (PSTN). V dnešní moderní době navíc zastává funkci komunikačního řešení propojující nejenom hlasové služby, ale i mobilní a datová komunikační zařízení s napojením do informačních systémů vlastníka zařízení. Hlavní výhodou pobočkové ústředny je úspora nákladů na interní telefonní hovory. S rostoucí popularitou začali PBX nabízet i takové služby, které nejsou k dispozici v síti operátorů.

1.1 Historie ústředen

Původní telefonní systémy ústředen byly založeny na přepojování okruhů (circuit switching). Znamenalo to fyzické propojení linkové cesty od zdroje (volající účastník) až k příjemci (volaný účastník). Mezi ně patří ústředny 1. generace – systémy s krokovými voliči,

2. generace – systémy s křížovými spínači, 3. generace – telefonní ústředny řízeny procesory s elektromechanickým či elektronickým spojováním. 4. generace využívá rozprostřené programové řízení a digitální spojování s využitím digitálních okruhů ISDN. Masivní růst datových sítí a zvýšení porozumění veřejnosti využití datových okruhů (Internet-TCP/IP) vedl k novému typu PBX systémů 5. generace, kde je spojení uskutečněno pomocí technologie přepojování paketů. Datové sítě společností a dostupnost internetu jako přenosového média tak vedly k rozvoji VoIP [7].

2 VEŘEJNÁ TELEFONNÍ SÍŤ PSTN

Veřejná komutovaná telefonní síť PSTN je celosvětová síť veřejných přepínaných okruhů. Je složena ze vzájemně provázaných telefonních kabelů, vláken optických kabelů, mikrovlnných spojů, celulárních sítí, komunikačních satelitů a podmořských telefonních kabelů atd., což umožňuje komunikovat s kýmkoliv na světě. Původně se jednalo o síť pevných linek analogových telefonních systémů, ale nyní je již téměř výhradně digitální a zahrnuje tak mobilní i pevné telefony.

Technický provoz sítě PSTN využívá standardy, které zastřešuje organizace ITU-T. Jedno z doporučení má označení E.164. Toto doporučení definuje mezinárodní veřejný telekomunikační číslovací plán používaný k telefonování a pravidla pro některé datové sítě. Zároveň definuje formát telefonních čísel. Mezinárodní číslo E.164 může mít maximálně 15 číslic a obvykle používá předponu +, nazývanou prefix. Tento standard umožňuje bezproblémové vzájemné propojení telefonních sítí v různých zemích [8].

2.1 Rozhraní telefonních sítí

Starší systémy pobočkových ústředen používají analogové telefonní linky (POTS), s rozhraním FXS, FXO, E&M a další.

- FXS – rozhraní pro připojení analogového telefonu
- FXO – rozhraní pro připojení telefonní linky operátora
- E&M – rozhraní pro vzájemné propojení telefonních pobočkových ústředen

Další používaná rozhraní jsou již digitální. Jedná se ISDN BRI, ISDN PRI, PCM, atd. Stále používají technologii přepojování okruhů, ale podporují různé doplňkové služby.

- ISDN BRI – základní digitální přípojka se dvěma hovorovými kanály a jedním signalizačním, 2B + D (2 x 64 kbit/s + 1 x 16 kbit/s).
- ISDN PRI – rozšířená digitální přípojka s třiceti hovorovými kanály a jedním signalizačním, 30B + D (30 x 64 kbit/s + 1 x 64 kbit/s).
- PCM 30/32 – Digitalizace a multiplexace hovorového signálu sloužící jen pro přenos mezi zdrojovým a koncovým bodem. 30 přenosových kanálů – již se nepoužívá.

Open-source projekty poskytující funkce telefonních ústředen devadesátých let, rozšířené o nové služby. Tyto projekty se prvotně zaměřují na VoIP, ale poradí si se staršími druhy

rozhraní a poskytují extrémní pružnost a funkce. Moderní rozhraní VoIP využívá některý z mnoha protokolů SIP, IAX2, H323, MGCP, atd.

- SIP – nejrozšířenější a jednoduchý textově orientovaný protokol podobný HTTP
- IAX2 – binární protokol nesoucí signalizaci ale i media na stejném portu
- H323 – binární multimediální protokol využívající několik různých portů, složitá implementace
- MGCP – protokol v systémech obvykle vzájemně propojujících VoIP a PSTN

2.2 Signalizace

Signalizace slouží k sestavení spojení, dohledem nad spojením po celou dobu jeho trvání, k spojení a uvolnění spojovacích cest použitých v přenosovém řetězci. Každé rozhraní používá určitý druh signalizace dané technologickou vyspělostí v době vzniku [3].

2.2.1 Rozdělení signalizací

Dle druhu přenosu:

- Analogový přenos, značky jsou reprezentovány napět'ovými úrovněmi, směrem protékajícího proudu ve vodičích nebo pomocí tónů DTMF.
- Digitální, tok bitů, pomocí signalizačních zpráv.

Analogová - dle druhu volby:

- Pulzní – sled napět'ových nebo proudových impulzů.
- DTMF – přiřazení znaku tónu o dvou frekvencích.

Digitální - dle použitého signalizačního kanálu:

- CAS – signalizace přidružená přenosovému kanálu – každý hovorový kanál má přidruženou signalizaci.
- CCS – signalizace společným kanálem – více kanálů používá jeden signalizační kanál.

Dle binární reprezentace signalizačního protokolu (VoIP)

- Textový – znakově orientovaný (vychází z HTTP), čitelný pro člověka
- Binární – přenos digitální informace, nečitelné pro člověka

2.3 ISDN – základní vlastnosti

Jedná se o telekomunikační službu navrženou pro hlas, video a data. Původ vzniku je v Německé republice. V Evropě se ISDN linky stále využívají, zatímco v USA nenaplnily očekávání. ISDN využívá několik účastnických signalizací. Signalizaci DSS1 určenou pro připojení koncových zařízení a pobočkových ústředěn, SS7 pro signalizaci mezi ústřednami telekomunikačních operátorů a QSIG pro vzájemné propojování pobočkových ústředěn [3], [9]. Existují dva druhy ISDN linek dělené dle přístupu a kapacity přenosu na PRI a BRI, zmíněné v kapitole 2.1.

Pro přenos užitečné informace se používají B kanály s přenosovou rychlostí 64kbit/s. Pro přenos signalizačních informací se používá D kanál s různou přenosovou rychlostí, závisící na typu přípojky. Jedna strana sítě je vždy *master* a druhá *slave*, tj. strana sítě a strana účastníka.

2.3.1 Základní přípojka BRI

Základní digitální přípojka, označená jako BRI, obsahuje tedy dva hovorové kanály a jeden signalizační, 2B + D (2 x 64 kbit/s + 1 x 16 kbit/s). Celková přenosová rychlost je včetně řídicích bitů 192kbit/s. Na tuto přípojku lze připojit až 8 koncových zařízení (telefon, fax G4, ...) [3]. BRI přípojka existuje ve dvou variantách:

- konfigurace PTP (Point to Point) pro připojení pobočkových ústředěn (někdy označovaná jako typ "D")
- konfigurace PMP (Point to Multipoint) a sběrnice pro připojení několika koncových zařízení paralelně (někdy označovaná jako typ "A").

U připojení pro pobočkové ústředny lze volit mezi provolbou DID a vícenásobným číslem MSN, případně sérií. U sběrnice připojení jiná varianta, než vícenásobné číslo MSN, neexistuje [10]. Vysvětlení pojmů DID, MSN bude popsáno dále v tomto textu.

2.3.2 Rozšířená přípojka PRI – E1

Rozšířená digitální přípojka, označená jako PRI má 30 B kanálů a jeden signalizační D kanál s vyšší přenosovou rychlostí než u základní přípojky BRI, 30B + D (30 x 64 kbit/s + 1 x 64 kbit/s). Celková přenosová rychlost je 2048 kbit/s. PRI přípojka se prakticky poskytuje pouze pro připojení pobočkových ústředěn s provolbou (DDI) v konfiguraci PTP [10]. Přípojky mají i své národní specifikace: Evropský standard se označuje někdy také

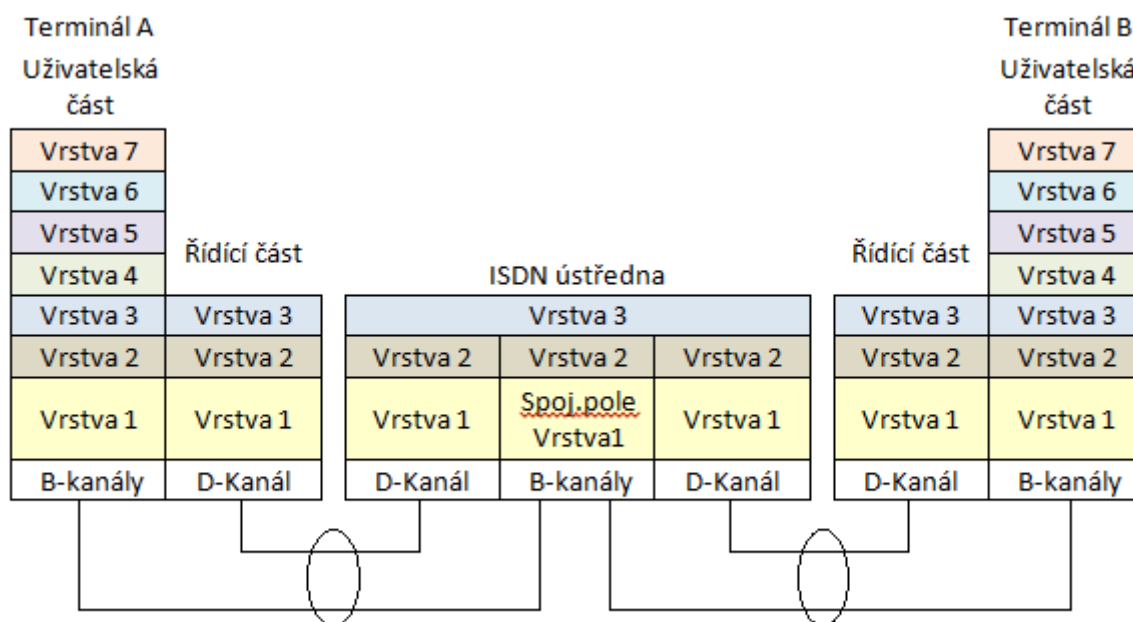
jako E1, v USA jde o T1, a v Japonsku o J1. Liší se hlavně počtem B-kanálů v základní úrovni a přenosové rychlosti na vyšších úrovních pospojování linek za účelem zvýšení kapacity přípojky.

E1	T1	J1
2.048 Mbit/s, 32 channels	1.544 Mbit/s, 24 channels	1.544 Mbit/s, 24 channels
8.448 Mbit/s (128 Ch.) (E2)	6.312 Mbit/s (96 Ch.) (T2)	7.786 Mbit/s (120 Ch.) (J2)

Tab. 2-1 - Porovnání linek E1, T1, J1

2.3.3 ISDN se signalizací DSS1 v modelu OSI

Z obecného referenčního modelu OSI vyplývá, že účastnická signalizace DSS1 používá pro svoji činnost první tři vrstvy. Obr. 2-1 ukazuje síťovou vrstvu 3 - protokol Q.931, spojovací vrstvu 2 - protokol Q.921, fyzickou vrstvu - protokol I.431 - PRI a I.430 – BRI.



Obr. 2-1 Model OSI - ISDN

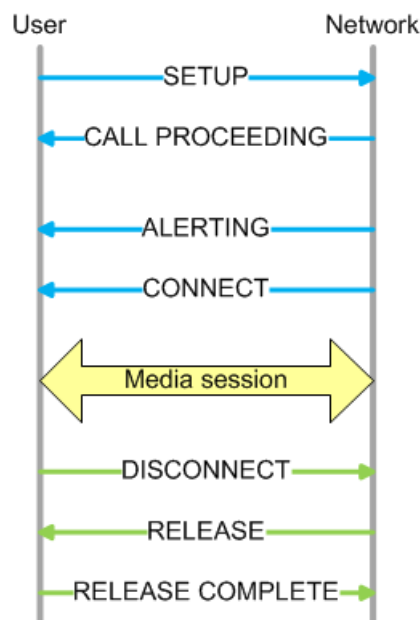
Model obsahuje vedle uživatelské části, která řídí aktivity na B kanálech, i řídicí část, která řídí signalizační proces. Toto oddělení má výhodu v tom, že ústředna nemusí vyhodnocovat informační tok B kanálů, ale pouze jej propojuje. Propojení B kanálů se uskutečňuje ve spojovacím poli ústředny [3].

- **Fyzická vrstva** je prezentována tokem bitů se stanovenou rychlostí (timing), kódováním a napětíovou úrovní. Kromě vlastního přenosu B a D kanálů probíhá na fyzické vrstvě aktivace a deaktivace stavu, kdy je možné přenášet data.

- **Spojová vrstva** se na úrovni L2 stará o zabezpečení přenášených dat a signalizace proti přenosovým chybám a o jednoznačnou identitu terminálu TEI. TEI je přiděleno každému terminálu v rozsahu 0-126, pro broadcast 127 a 0 pro PTP.
- **Síťová vrstva** se stará o popis procedur pro řízení spojení, procedur pro služby přenosu rámců a popis kódování informačních prvků k definování kompatibility různých služeb. Začátek zprávy je složen ze tří částí:
 - Typ protokolu
 - Call Reference – označení všech zpráv patřící jedné signalizační aktivitě
 - Message Type – procedura řízení spojení (např. Connect, Setup, ...) [3]

2.3.4 Některé signalizační zprávy DSS1

Tak jako ostatní typy rozhraní (SIP, H323, ale i FXO, FXS,...), tak i ISDN používá své vlastní specifické zprávy pro komunikaci dotaz-odpověď. Zprávy se vyměňují na třetí vrstvě pomocí protokolu Q. 931, který je zhruba srovnatelný s TCP protokolem. Průběh výměny zpráv při sestavení spojení lze zobrazit pomocí sekvenčního diagramu viz Obr. 2-2, podobně jako se zobrazuje průběh spojení např. u protokolu SIP.



Obr. 2-2 Průběh výměny zpráv při sestavení spojení

Nejčastěji používané zprávy DDS1:

- SETUP – žádost o sestavení spojení
- ALERTING – indikace vyzvánění volaného

- CALL PROCEEDING – sestavení volání, přidělen B-kanál
- CONNECT – signalizace přihlášení volaného, příchozí volání bylo přijato
- SETUP ACKNOWLEDGE – potvrzení zprávy SETUP, při neúplné volbě
- CONNECT ACKNOWLEDGE – potvrzení propojení na zprávu CONNECT, spojení bylo navázáno
- DISCONNECT – výzva k rozpadu spojení, zavěšení
- RELEASE – odpověď na DISCONNECT, uvolnění B kanálu
- RELEASE COMPLETE – potvrzení RELEASE
- INFORMATION – přenos dodatečných informací mezi terminálem a ústřednou
- PROGRESS – označuje určitou fázi sestavování spojení
- FACILITY – pro přenos doplňkových služeb

Některé další zprávy DSS1: RESTART, HOLD, RETRIEVE, SUSPEND, RESUME [3].

```

Q931/CORNET-N 26B9H ORIG SETUP inicializace spojení
W-elem: (A1H) SENDING COMPLETE volané číslo kompletní (volba v bloku)
W-elem: ( 4H) BEARER CAPABILITY
  Laenge 3
  Octett 3: Coding Standard: CCITT
            Transfer cap : 3.1kHz AUDIO
  Octett 4: Transfer Mode : Circuit mode
            Transfer Rate : 64kbit/s
  Octett 5: Layer ID : LAYER 1
            L1 Protocoll : A-Law
W-elem: (18H) CHANNEL ID druhý B-kanál
  Laenge 3
  Octett 3: B-Chl selection : EXCLUSIVE
            Channel : as indicated
            ID presentation : IMPLICITLY IDENTIFIED
            Type : other, e.g. primary
            D-Chl Indicator : THIS IS NOT THE D-CHL
  Octett3.2: Coding Standard : CCITT
            MAP/NUMBER : NUMBER
            Channel Type : B-CHANNEL UNITS
  Octett3.3: Channel number : 2H
W-elem: (1EH) PROGRESS INDICATOR volající je NON-ISDN
  Octett 3: Coding Standard : CCITT
            Location : PUBLIC NETWORK SERVING THE LOCAL USER
  Octett 4: Descriptor : ORIG ADDRESS NON ISDN
W-elem: (6CH) CALLING PARTY NUMBER volající 581216321
  Octett 3: Type of Number : UNKNOWN typ čísla neznámý
            Numbering Plan : UNKNOWN
  Octett 3a: Presentation : PRESENTATION ALLOWED
            Screen ind : USER PROVIDED, VERIFIED AND PASSED
  Octett 4: Number : 581216321
W-elem: (70H) CALLED PARTY NUMBER číslo volaného 43927
  Octett 3: Type of Number : UNKNOWN typ čísla neznámý
            Numbering Plan : ISDN
  Octett 4: Number : 43927

```

Obr. 2-3 Příklad zprávy SETUP u DSS1

2.3.5 Služby v ISDN

Služby v ISDN jsou rozděleny na tři části: Nosné služby, teleslužby, doplňkové služby. **Nosné služby** přenášejí informaci přes síť a dělí se na služby, které používají pro přenos přepojování okruhů a přepojování paketů [3].

- Přepojování kanálů
 - **Unrestricted mode**, 64kbit/s neomezený přenos – zvuk, text, data
 - **3,1 KHz Audio**, pro 3,1 kHz audio v kanále 64 kbit/s (fax, modem, ...)
- Přepojování paketů

- Přenos dat v B-kanále (max 64kbit/s)
- Přenos dat v D-kanále (X.25, do 9,6 kbit/s)

Teleslužby využívají pro přenos všech 7 vrstev modelu OSI.

- Telefon s šířkou pásma 3,1 kHz nebo 7kHz
- Teletex nebo telefax s rychlostí 64 kbit/s
- Videotelefonie
- Datová komunikace

Doplňkové služby (výběr jen pro potřeby této práce)

- CLIP – zobrazení identifikace volajícího
- CLIR – zamezení identifikace volajícího
- DDI – provolba, poslední číslice volaného čísla se shodují s interním číslovacím plánem. Provolba umožňuje přímé spojení na volaného účastníka bez použití obsluhy (volaný účastník nemusí být přepojen na volaného operátorem). Poskytují se bloky 10, 100, 1000 a 10000 čísel provolby.
- MSN – Vícenásobné číslo: Umožňuje nadefinovat až 8 různých čísel pro jednu přípojku ISDN. Každému připojenému zařízení je pak dáno jeho číslo, takže lze volat např. cíleně fax, ISDN telefon, videokameru apod.
- TP – přenositelnost terminálů

2.4 Signalizace QSIG a SS7

Pro ucelenou informaci o ISDN budou jen okrajově zmíněny další signalizace, které se ale v této práci nevyskytují.

QSIG je signalizační systém v referenčním bodě Q, což je bod mezi dvěma PBX. Signalizace je nezávislá na struktuře propojení, je symetrická, tzn. nemá stranu účastníka a stranu sítě.

Signalizační systém č. 7 (SS7) se používá v ISDN a GSM sítích, které umožňují vytváření hovorových spojení, nasazení inteligentních sítí, práci s databázemi, přenositelnost telefonních čísel, podporu SMS, atd. [3].

2.5 Signalizační protokol SIP

Asi nejrozšířenější protokol pro IP telefonii, vznikl v roce 1999 a byl standardizován jako RFC 2543. V roce 2002 došlo k rozšíření a aktualizaci protokolu popsaného ve standardě

RFC 3261. Jedná se o textově orientovaný signalizační protokol, který pracuje na aplikační vrstvě. Pro úplnou implementaci VoIP ovšem nestačí samotný protokol SIP, ale další protokoly, jako jsou RTP a SDP.

K protokolu SIP existuje mnoho dokumentace dostupné zejména na síti Internet. Pro potřeby této práce bude proto popis tohoto protokolu zaměřen jen na základní vlastnosti bez velkých detailů a budou zdůrazněny jen speciální potřebné vlastnosti. Pro více informací o SIP protokolu je doporučeno nahlédnout do dokumentace [2], [12], [13].

2.5.1 SIP základní vlastnosti

SIP je end-to-end orientovaný signalizační protokol, což znamená, že veškerá logika je uložena v koncových zařízeních (vyjma směrování SIP zpráv). Koncové zařízení zná i jednotlivé stavy komunikace, tím je zvýšena odolnost komunikace proti chybám. Nepochybně stojí za zmínku, že end-to-end koncept SIPu je významná odlišnost od klasického řešení PSTN, kde logika je uložena v síti a koncová zařízení jsou podstatně jednodušší. Cílem SIPu je zajistit stejnou funkcionalitu jakou mají klasické PSTN, ale end-to-end návrh umožní SIP sítím vyšší výkonnost a otevře implementaci nových služeb, které mohou být jen obtížně nasazeny v klasických PSTN. Komunikace v protokolu SIP probíhá výměnou dvou typů zpráv, požadavků a odpovědí. Klient i server jsou logickou částí jednoho prvku [2]. SIP je v síti identifikován použitím SIP URI:

sip:user:password@host:port;uri-parameters?headers

SIP URI se skládá z části user a z části host, obě části jsou oddělené znakem @. SIP URI je podobná e-mailové adrese. Část user identifikuje uživatele v doméně prezentované v části host, která může být zadána pomocí jména nebo IP adresy. Pokud není uvedeno číslo portu, tak se předpokládá použití všeobecně známého portu 5060 na UDP. Parametry mohou nést další volitelné informace. Případné parametry se oddělují středníkem, údaje pro zadání parametrů hlavičky se uvádí v URI za otazníkem [2]. SIP URI používá číselné identifikátory (telefonní číslo), ale může použít i jmenný identifikátor.

sip:petr.karolin@server.pbx.net *jmenný identifikátor*

sip:598245262@server.pbx.net *číselný identifikátor*

Avšak pro další směrování hovorů napříč různorodými sítěmi (VoIP, PSTN, ...) je lepší použít číselný identifikátor, který umožní dynamické směrování na základě daného číslovacího plánu ústředn. K převodu identifikátorů může pomoci i služba ENUM.

2.5.2 Prvky SIP

V nejjednodušší konfiguraci je možné použít dva UA posílající si navzájem SIP zprávy. Typická SIP síť však bude obsahovat více než jeden typ prvků. Základními SIP prvky jsou:

- user agents (UA)
- proxies, registrars, and redirect servers.

Jednotlivé servery jsou většinou prezentovány jako logické části SIP serveru, jelikož je často efektivní provozovat je společně na jednom HW. Koncové body sítě Internet, které používají SIP ke vzájemnému spojení, jsou nazývány jako user agents. Každý UA je během komunikace buď User Agent Server (UAS) nebo User Agent Client (UAC).

- UAC – část vysílající požadavky a přijímající odpovědi
- UAS – část přijímající požadavky a odesílající odpovědi.
- B2BUA – speciální typ UA, vytváří dvě spojení, může se chovat jako brána mezi dvěma UA, možnost modifikace paketů, vložení dalších služeb (změna identifikace, nahrávání hovorů, ...).

Požadavek a odpověď jsou dva základní typy SIP zpráv. Protože koncové zařízení téměř vždy obsahuje UAC a UAS, tak používáme pouze označení UA namísto UAC a UAS [2], [12], [13].

2.5.3 SIP servery

SIP servery se nazývají SIP Proxy. Mají za úkol směrování žádostí o spojení, provádí autentizaci, kontrolují umístění user agentů a další služby (např. přesměrování). Dva základní typy serverů SIP Proxy:

- stateless (bezstavový)
- stateful (drží informaci o stavu transakce)

Stateless – jednoduchý server, který přeposílá zprávy nezávisle na jejich vazbách. Nekontroluje zprávy, nezachytí chybu v komunikaci. Jsou velmi rychlé. Využívají se pro balanční servery, pro překládání zpráv a směrování.

Stateful – drží záznam o stavu transakce, odhalí chyby komunikace, protože zná stav dialogu. Může poskytnout účtování, větvení a jiné doplňkové služby [2], [12], [13].

Jedním z mnoha proxy serverů umožňující tyto služby, je i softwarová ústředna Asterisk popsána v této práci.

2.5.4 Požadavky a odpovědi protokolu SIP

Komunikace je tvořena zprávami. Zpráva se skládá z hlavičky (header) a může obsahovat tělo (body), popisující média. Hlavička a tělo jsou většinou odděleny volným řádkem.

V prvním řádku je identifikátor typu zprávy, který může nabývat dvou hodnot:

- Žádost
- Odpověď

```

Request-Line: INVITE sip:333@192.168.55.22:5066 SIP/2.0
Via: SIP/2.0/UDP 62.134.102.250:5060;branch=z9hG4bK6fcdd620;rport
Max-Forwards: 70
From: "Ludik Martin" <sip:188@82.114.193.60>;tag=as66880790
To: <sip:333@192.168.55.22:5066>
Contact: <sip:188@62.134.102.250:5060>
Call-ID: 58ac1f186ac0a6fd44d635a27b9f97dc@62.134.102.250:5060
CSeq: 102 INVITE
User-Agent: Asterisk PBX 1.8.3.3
Date: Fri, 30 Mar 2012 13:36:35 GMT
Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, SUBSCRIBE, NOTIFY, INFO, PUBLISH
Supported: replaces, timer
Content-Type: application/sdp
Content-Length: 281

v=0
o=user 88975990 88975990 IN IP4 62.134.102.250
s=Asterisk PBX 1.8.3.3
c=IN IP4 62.134.102.250
t=0 0
m=audio 13080 RTP/AVP 8 3 0 101
a=rtpmap:8 PCMA/8000
a=rtpmap:3 GSM/8000
a=rtpmap:101 telephone-event/8000
a=fmtp:101 0-16
a=ptime:20
a=sendrecv

```

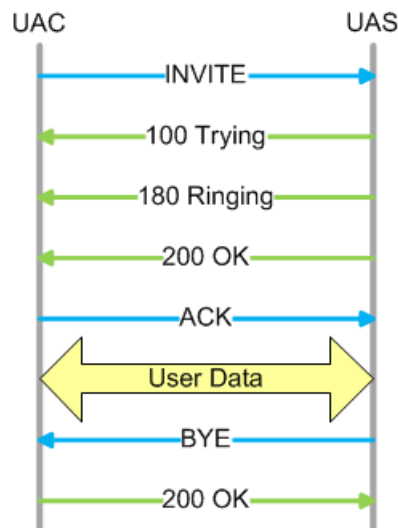
Tab. 2-2 - SIP hlavička (červená) a tělo (zelená) zprávy

Žádosti jsou obvykle užívány k inicializaci procedury (sestavení, ukončení spojení) nebo oznamují příjemci požadavek na něco. Odpovědi jsou užívány k potvrzení, že žádost byla přijata a zpracována a obsahuje stav zpracování. Tab. 2-2 ukazuje typickou žádost SIP.

První řádek ukazuje žádost INVITE k sestavení spojení. Položka *Via* znázorňuje cestu pro zaslání SIP odpovědi. Položek může být více. Pole *From* a *To* identifikuje iniciátora volání

- **6xx**: Globální chyba. Požadavek nelze provést na žádném serveru

Odpovědi s kódem 200 a výše jsou konečné, jejich přijetí ukončuje transakci [2], [12], [13].

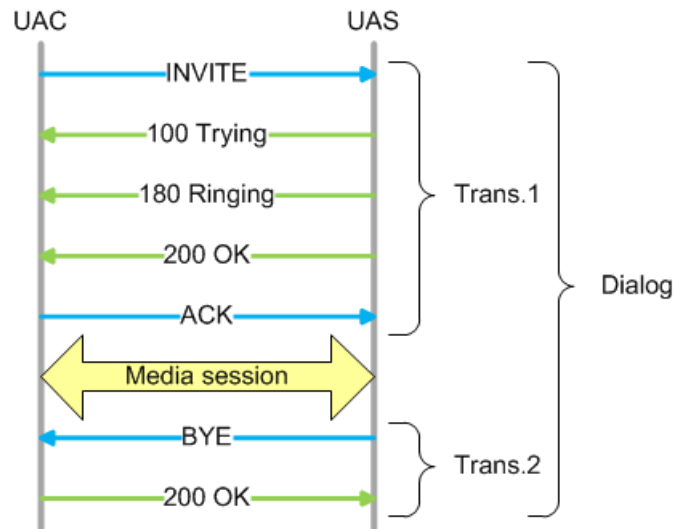


Obr. 2-5 Výměna zpráv v dialogu

2.5.6 SIP transakce a dialog

Transakce je sekvence SIP zpráv vyměňovaných mezi síťovými SIP prvky. Obsahuje jednu žádost a všechny odpovědi. Výměna zpráv stejné transakce musí mít stejný identifikátor nazývaný *branch* umístěný v poli hlavičky SIP.

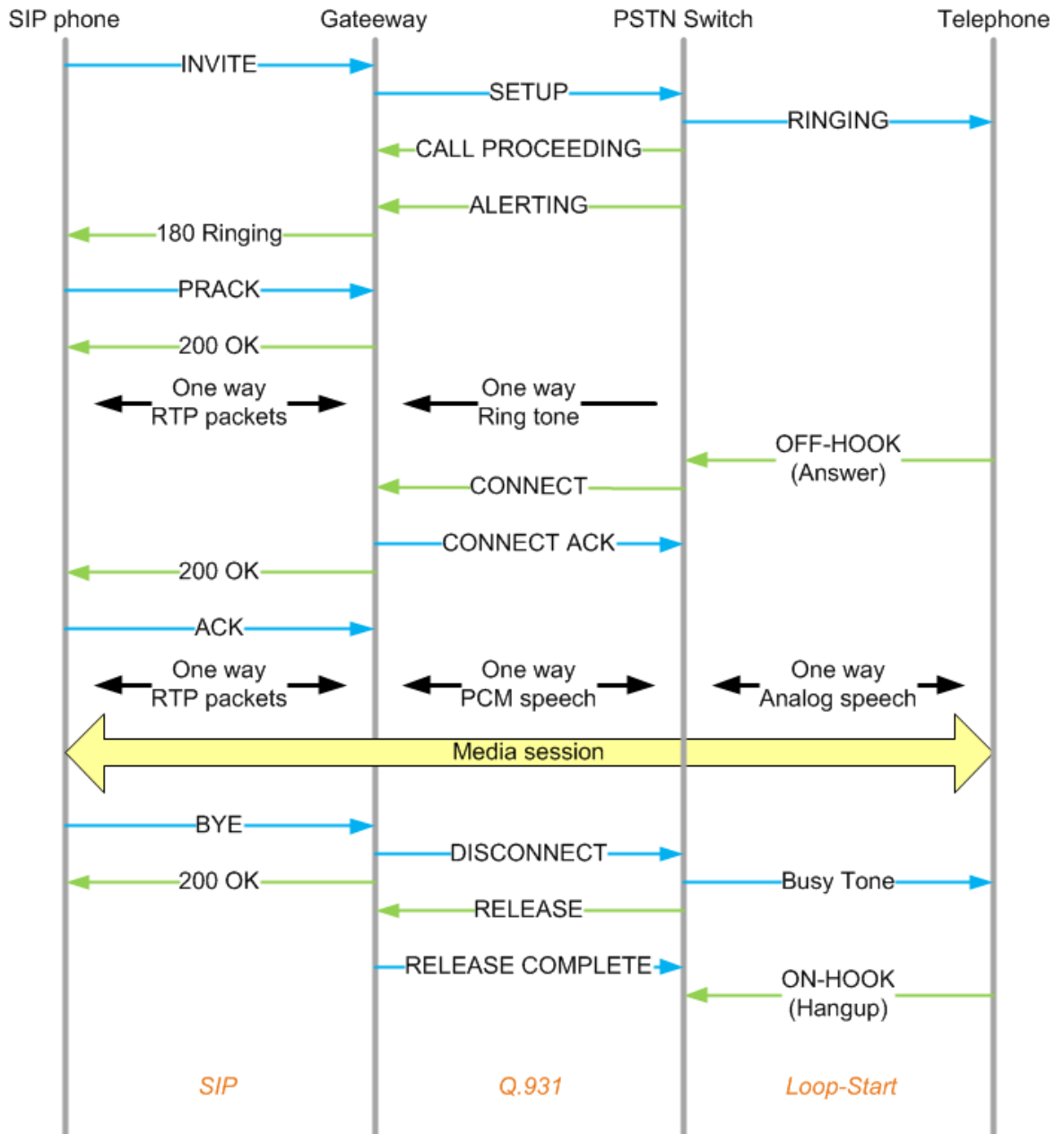
Transakce, které spolu souvisí, se nazývají dialog. Dialog je širší pojem pro kompletní hovor. Identifikace dialogu v hlavičkách protokolu SIP je pomocí pole *Call-ID*, *From (tag)*, *To (tag)*. Pokud jsou tyto identifikátory stejné, náleží jednomu dialogu. Názorné rozdělení transakcí a dialogu ukazuje Obr. 2-6.



Obr. 2-6 Transakce a dialog SIP protokolu

2.6 Spojení z VoIP přes ISDN na analogový terminál s U-rozhraním

Spojení z VoIP terminálu na klasický analogový telefon, musí být realizováno pomocí zařízení, které rozumí více signalizacím a protokolům. Takovéto zařízení se nazývá brána, ale častěji se používá anglický název *gateway*. Na ukázce je spojení, realizované na straně VoIP je pomocí protokolu SIP, síťová část propojená linkou ISDN mezi bránou a koncovou telefonní ústřednou se využívá protokol Q.931 a spojení mezi telefonní ústřednou a koncovým analogovým terminálem na U-rozhraní zajišťuje signalizace Loop-Start.



Obr. 2-7 Tok zpráv při síťování mezi SIP, Q.931, Loop-Start

3 NEJEN TELEFONNÍ ÚSTŘEDNA ASTERISK

Asterisk je jedna z mnoha celosvětově oblíbených, volně dostupných softwarových aplikací, řešících funkci telefonní ústředny. Vlastností ústředny jsou nejenom rozsáhlý provoz systému VoIP, ale i možnost připojení různorodého telekomunikačního rozhraní (FXO, FXS, ISDN), za pomoci vhodného podporovaného hardware, pracujícího jako brána (gateway) mezi TDM a VoIP částí. Instaluje se na operační systém Linux, což umožňuje integraci dalších volně šiřitelných aplikací pracujících s tímto operačním systémem a telefonní ústřednou. Asterisk umožňuje jednoduše vytvořit a implementovat celou řadu telefonních aplikací a služeb, včetně IP PBX a VoIP brány, Call centra a ACD IVR systémů. Asterisk je uvolněn jako open source, pod licencí GNU General Public License (GPL) a je k dispozici ke stažení zdarma.

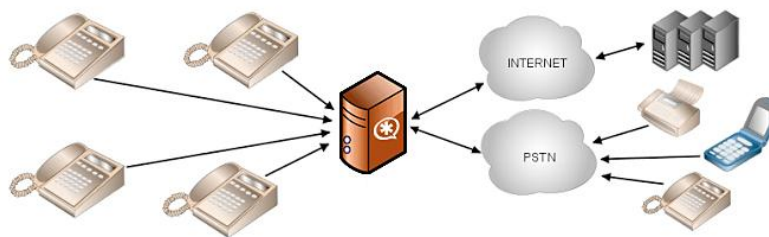
3.1 Vlastnosti Asterisku

Jde vlastně o stavebnici srovnatelnou se systémem „Lego“. Jednotlivé funkční bloky se spojují v jeden celek dle požadavků na druh budoucí komunikace. Asterisk zahrnuje všechny stavební kameny potřebné pro vytvoření telefonní ústředny s jakýmkoliv druhem řešení komunikace. Možné části této „stavebnice“ jsou:

- Ovladače pro různé protokoly VoIP (SIP, IAX, MGCP, H323, SCCP, ...)
- Ovladače pro HW karty pro rozhraní PSTN (FXO, FXS, ISDN)
- Směrování hovorů různými algoritmy
- Práce s médii (nahrávání, hlasové syntézy, rozpoznání řeči, ...)
- Tarifkace hovorů
- Převod telekomunikačních protokolů (jako brána mezi systémy)
- Převod kodeků
- Integrace s relačními databázemi
- Integrace webových služeb
- Integrace s LDAP adresářovým systémem
- Vlastní skriptovací jazyk pro zpracování hovorů (Dialplan)
- Externí řízení ústředny pomocí skriptovacího jazyku AGI a AMI

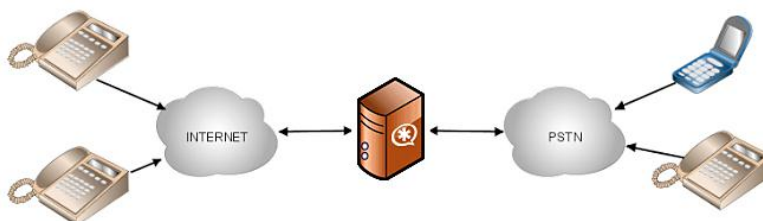
Tato kombinace funkcí umožňuje rychle vytvářet různorodé hlasové aplikace. Otevřená povaha Asterisku znamená, že neexistuje žádný pevný limit toho, co lze udělat [14].

- Asterisk jako ústředna – s možností analogového (POTS) nebo digitálního (ISDN) připojení



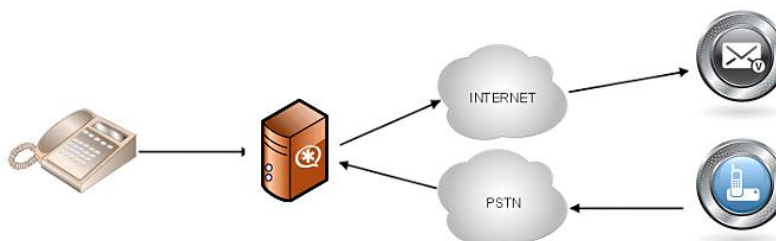
Obr. 3-1 Asterisk jako ústředna

- Asterisk jako brána – umožňuje převod mezi velkou škálou komunikačních protokolů a mediálních kodeků.



Obr. 3-2 Asterisk jako brána

- Asterisk jako call centrum – flexibilní řešení pro vybudování ACD. Asterisk také oživuje stávající řešení ACD přidáním pokročilých vlastností směrování hovorů, možností napojení na informační systém atd.
- Asterisk voicemail – nahrazuje starší řešení služby zanechání vzkazu a rozšiřuje je o nové funkce, s nízkými pořizovacími náklady.



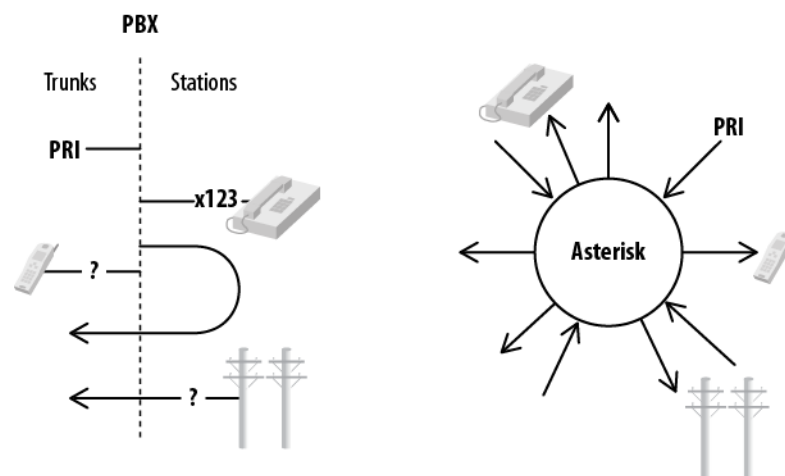
Obr. 3-3 Asterisk voicemail

- Asterisk jako konferenční server – umožňuje vytváření sofistikovaných konferenčních místností pomocí jednoduchého skriptu v „Dialplanu“, bez větších omezení a limitů počtu uživatelů.

- Asterisk jako ACD – funkce automatické spojovatelky s nízkými náklady na HW serveru a telefony [14].

3.2 Architektura Asterisku

Tradiční telefonní ústředny rozlišují mezi interními stanicemi (stations - inputs) a vnějšími linkami (trunks - outputs). Například to znamená, že jen omezeně lze používat na portech interních stanic externí brány (mezičlánky pro propojení do jiných telekomunikačních sítí). U nich nelze směřovat volání, aniž by bylo nejprve voleno nějaké číslo, například číslo interní stanice, atd. A naopak nelze na vnější porty připojit koncový terminál [4].



Obr. 3-4 Porovnání architektury standardní PBX a Asterisku

Všechny druhy terminálů a telefonních linek (přípojek) zahrnuje Asterisk pod pojmem kanál. Ústředna je velmi odlišná od ostatních tradičních pobočkových ústředen, protože zpracovává všechny vstupní i výstupní kanály stejným způsobem. Možné kanály jsou SIP, DAHDI, LOCAL, IAX, MGCP, atd. Asterisk je sestaven z modulů. Modul nese komponentu, která poskytuje určité funkce, jako je například ovladač kanálu (např. SIP má ovladač *chan_sip.so*), nebo zdroj, který umožňuje připojení k vnější technologii (např. databáze *odbc* má ovladač *func_odbc.so*). Každý modul proto zastupuje nějakou funkci ústředny [14]. Konfigurace se provádí změnou údajů v konfiguračních souborech Asterisku. Např. pro konfiguraci kanálů SIP se použije konfigurační soubor *sip.conf*, pro nastavení načítání modulů je to *modules.conf*, pro správu připojení AMI/AGI soubor *manager.conf*, atd. Pro více informací o modulech, konfiguračních souborech a jejich seznam, lze najít v Asterisk dokumentaci [14].

3.2.1 Kanál SIP

Neumožňuje jen připojení koncových terminálů VoIP na protokolu SIP, ale i registraci k VoIP účtům telefonních poskytovatelů. Konfiguruje se pomocí konfiguračního souboru *sip.conf*. Protože se jedná o signalizační protokol, musí nutně spolupracovat s protokolem, který přenáší média (hlas), a tím je protokol RTP konfigurovaný v souboru *rtp.conf*.

3.2.2 Kanál DAHDI

Kanál DAHDI umožňuje připojení „analogové telefonní sítě“ (FXO, FXS, BRI, PRI linek) pomocí vhodného hardware. Tento hardware se stará o připojení těchto rozhraní na fyzické vrstvě (hlídá napěťové úrovně signálu, jeho časování atd.). Dodává se ve formě HW karet určených do PCI nebo PCIe slotů. Před použitím se do hostitelského operačního systému musí nainstalovat správné ovladače. Konfigurace kanálu DAHDI v Asterisku se provádí pomocí konfiguračního souboru *chan_dahdi.conf*.



Obr. 3-5 HW karta Sangoma



Obr. 3-6 HW karta Digium

3.2.3 Kanál LOCAL

Lokální kanál LOCAL, je metoda v Asterisku k rozšíření dialplanu, někdy také nazývána pseudo-kanál. LOCAL kanál umožňuje větvení nebo rekurzivní směrování s možností vrácení zpět do dialplanu po dokončení hovoru. Například volání na dvě destinace (SIP, DAHDI, atd.) s různým nastavením parametrů (časové, identifikace ID, atd.). V Tab. 3-1 je ukázka větvení dialplanu pomocí lokálního kanálu. Princip zápisu dialplanu bude vysvětlen dále v této práci.

```
[dev]
exten => 333,1,NoOp(4,Volani pevny telefon a mobilni se zpozdenim)
exten => 333,n,Dial(Local/pevny-333@ext&Local/mobile-333@ext,30)
exten => 333,n,Hangup()

[ext]
; Volba pevneho telefonu
exten => pevny-333,1,Verbose(4,Volani na pevny telefon 333)
exten => pevny-333,n,Dial(SIP/333)

; Volba mobilni stanice se 6s zpozdenim
exten => mobile-333,1,Verbose(4,Dialing cellphone of extension 201)
exten => mobile-333,n,Verbose(4,-- Waiting 6 seconds before dialing)
exten => mobile-333,n,Wait(6)
exten => mobile-333,n,Dial(DAHDI/g0/606606606)
```

Tab. 3-1 - Ukázka větvení dialplanu pomocí lokálního kanálu

3.3 Dialplan

Dialplan je základem Asterisku. Všechny kanály, které vstoupí do systému ústředny, jsou zpracovány prostřednictvím dialplanu, který určuje způsob zpracování příchozích volání. Změna konfigurace, se provádí změnou zápisu v konfiguračním souboru. Zápis pravidel může být napsán jedním ze tří způsobů [4]:

- Pomocí tradiční syntaxe v souboru *extensions.conf*
- Pomocí zápisu rozšířené logiky (AEL) v souboru *extensions.ael*
- Pomocí speciálního LUA zápisu v souboru *extensions.lua*

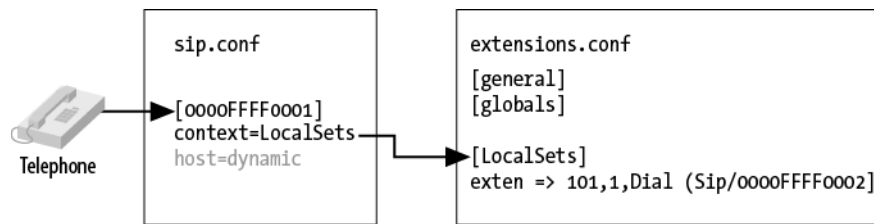
Dialplan se skládá ze čtyř hlavních pojmů: *context*, *extension*, *priority* a *aplication*. Následně si vysvětlíme tyto stavební prvky, podle zápisu konfiguračního souboru *extension.conf* [4].

3.3.1 Kontext

Dialplan je rozdělen do celků, zvaných kontexty. Kontexty udržují různé části dialplanu v interakci mezi sebou. Kontexty jsou definovány názvem kontextu uvnitř hranatých závorek [muj_kontext]. Název může být vytvořen z písmen od „a“ do „Z“, čísel 0 až 9 a podtržítka. Všechny instrukce umístěné za názvem kontextu jsou součástí této definice, až do definice následujícího kontextu [14].

Definice kanálu probíhá v konfiguračních souborech (*sip.conf*, *iax.conf*, *chan_dahdi.conf*, atd.), a jeden z požadovaných parametrů v každém kanálu je definice kontextu. Kontext je

bod v dialplanu, kde začne připojení tohoto kanálu. Vztah mezi konfiguračními soubory kanálů a kontextů v dialplanu znázorňuje Obr. 3-7[4].



Obr. 3-7 Vztah kontextu a dialplanu

3.3.2 Extension

Extension definuje jedinečný postup (pravidlo), jehož prostřednictvím bude ústředna zpracovávat hovor (provádět danou aplikaci). Jedná se o filtr, který kontroluje, budou-li se shodovat znaky vstupující do kontextu se znaky *name* v dialplanu. Při shodě se provede následující aplikace. Názvy extension mohou být libovolné kombinace čísel a písmen, řízené určitými pravidly nazvaných „Pattern-matching“. Zápis extension je slovo *exten*, a následuje znak rovná se, se znakem větší než, tvořící šipku, a to takto [4]:

```
exten =>
```

Tab. 3-2 - Zápis extension

Každý krok na řádce extension se skládá ze tří složek:

- Name - název (text nebo číslo)
- Priority - každá extension může zahrnovat více kroků, priority určují pořadí zpracování skriptů
- Application (další volby) – příkaz, který se uskuteční v tomto kroku

```
exten => name,priority,application()
```

Tab. 3-3 - Tři položky *exten*, odděleny čárkami

```
exten => 581123456,1,Answer()
```

Tab. 3-4 - Příklad zápisu extension

3.3.3 Priority

Každá extension může mít více kroků, nazývané priority. Priority jsou číslovány sekvenčně, počínaje číslem 1, kde každý krok provede jednu konkrétní aplikaci.

```
exten => 581123456,1,Answer()  
exten => 581123456,2,Hangup()
```

Tab. 3-5 - Ukázka odpovědi a zavěšení

3.3.4 Aplikace

Aplikace jsou stěžejním krokem dialplanu. Každá aplikace provede konkrétní akci v aktuálním kanálu, např. přehrávání zvuku, přijetí tónová volby, práci s databází, volba kanálu, zavěšení hovoru atd. V předchozím příkladu byly představeny dvě jednoduché aplikace: Answer() a Hangup(), které nepotřebují vysvětlení, jejich funkce vychází ze samotného názvu. Aplikace mohou být rozšířeny o funkce. Některé vybrané aplikace s jednoduchým popisem:

- Answer() – propojí zvonící kanál
- Playback() – přehraje zvukový soubor
- Hangup() – ukončí všechny akce, odpojí připojený kanál
- Goto() – skok v dialplanu
- Background() – přehraje zvukový soubor s možností volby DTMF v kanále
- Wait() – čeká nastavenou dobu v dialplanu před skokem na další prioritu
- Dial() – nejdůležitější aplikace Asterisku, volá a propojuje kanály, aplikace má mnoho dalších voleb

```
exten => 5437,1,NoOp(Dial user: SIP/${EXTEN})  
exten => 5437,2,Playback(welcome)  
exten => 5437,3,Dial(SIP/${EXTEN:1},15,tTr)  
exten => 5437,4,Hangup()
```

Tab. 3-6 - Ukázka dialplanu jednoho kontextu a konkrétní extensions

3.3.5 Řetězce a porovnání vzorů v dialplanu

Jména extension nejsou omezena jen na jednotlivé specifické čísla, nebo jména (vzory). Jednotlivé extensions mohou rovněž odpovídat vzorům. Porovnávání vzoru umožňuje vytvořit jednu příponu v dialplanu, která odpovídá více různým číslům (vzorům).

V souboru *extension.conf* se jméno *extension* stává vzorem, jestliže začíná znakem podtržení. V *extension* vzorech mají následující znaky speciální smysl [12]:

```
X - odpovídá některé z číslic 0 - 9
Z - odpovídá některé z číslic 1 - 9
N - odpovídá některé z číslic 2 - 9
. - odpovídá jednomu nebo více znakům
! - odpovídá žádnému nebo více znakům
```

Tab. 3-7 - Znaky vzorů se speciálním významem

```
${123456789:1} - vrací řetězec 23456789
${123456789:-4} - vrací řetězec 6789
${123456789:0:3} - vrací řetězec 123
${123456789:2:3} - vrací řetězec 345
${123456789:-4:3} - vrací řetězec 678
```

Tab. 3-8 - Syntaxe řetězců vracejících hodnotu danou parametrem za dvojtečkou

```
exten => _[67]ZXX,1,Dial(IAX/${EXTEN:3})
```

Tab. 3-9 - Ukázka práce se vzory a řetězci

Vzor *extension* v Tab.3-9 znamená, že se očekává volba čtyřmístného čísla začínajícího znakem 6 nebo 7. Na druhé pozici se nesmí objevit číslo 0. Syntaxe „(*\${EXTEN:3}*)” provede odříznutí prvních tří čísel. Pravidlu tedy vyhoví čísla 6200-6999 a 7200-7999. Aplikace *Dial* propojí kanál *IAX* s číslem v rozsahu 00-99 [4], [12].

3.3.6 Makra v dialplanu

Makra jsou velmi užitečné konstrukce s cílem vyhnout se opakování v dialplanu. Pomáhají také při provádění změn dialplanu. Pro představu systém se stovkou uživatelů - nastavení pravidel pro každého uživatele by vyžadovalo mnoho kopií v dialplanu. Neúměrně by se zvětšoval konfigurační soubor a při editaci by jistě vznikaly chyby. Místo toho se může definovat makro, které obsahuje seznam kroků a pravidel v dialplanu. Ostatní volby se pak odkazují na toto makro. Vše, co je potřeba změnit, je makro a všechna pravidla v dialplanu se změni také. Makro je aplikace dialplanu vypadající jako kontext.

```
Macro(macroname,arg1,arg2...)
```

Tab. 3-10 - Správná syntaxe makra

Makro má první parametr povinný, a to název volaného makra, kde jsou definována pravidla dialplanu. Následují předávané argumenty oddělené čárkou. Vlastní makro vypadá jako kontext, ale v názvu musí být slovo „macro-“ a vlastní název makra. V těle makra se už neobjevují jednotlivé hodnoty přímo, ale pracuje se s argumenty nesoucí potřebné informace o proměnných. Makro názorně ukazuje následující tabulka [4].

```
[level_0]
;=====
exten => _4XX,1,Macro(voipUA,${EXTEN})
exten => _5XX,1,Macro(voipUA,${EXTEN})

[macro-voipUA]
;=====
exten => s,1,NoOp(5, Dial user: SIP/${ARG1})
exten => s,2,Dial(SIP/${ARG1},,tT)
```

Tab. 3-11 - Ukázka práce s makrem

3.4 Asterisk databáze AstDB

Asterisk poskytuje mocný nástroj pro ukládání a zpracování hodnot označovaný jako Asterisk databáze (AstDB). AstDB je jednoduchý způsob, jak lze v dialplanu zpracovávat data. Nejedná se o tradiční relační databázi, ale databázi Berkeley DB verze 1. Data se ukládají podle pravidla *klíč/hodnota data* (*key/value data*), ale v Asterisku je tohle pravidlo popsáno trochu jiným způsobem a to: *<family> <key> <value>*. Tato databáze je použita ve všech verzích Asterisku až do verze 1.8. Ve verzi Asterisku 1.10 se již používá databáze SQLite3 [4].

3.4.1 Ukládání dat do databáze

Pro uložení nové hodnoty do AstDB, se používá aplikace Set() s parametrem DB(), která obsahuje vlastní další volby. Tímto parametrem je klíč k uložení dat.

```
exten => _5492,1,Set(DB(cislo/mobil)=602082435)
```

Tab. 3-12 - Uložení dat do databáze

Tab. 3-12 ukazuje uložení hodnoty 602082435 do AstDB na pozici *family=číslo* a klíč *key=mobil*. Hodnoty lze také přidávat pomocí příkazového řádku Asterisku spuštěním příkazu *databáze put <family><key><value>* [4].

3.4.2 Načtení dat z databáze

Při načtení je vhodné použít načtení do lokální proměnné, ale není to podmínka. Opět bude použita aplikace Set(). V ukázce Tab. 3-13 je načtena hodnota databáze do proměnné s názvem TEST, a následně je volajícím přečteno číslo pomocí aplikace SayNumber().

```
exten => _5492,1,Set(DB(cislo/mobil)=602082435)
exten => _5492,2,Set (TEST=${DB(cislo/mobil)})
exten => _5492,3,Set,SayNumber(${TEST})
```

Tab. 3-13 - Načtení dat z databáze

Získání dat pomocí příkazového řádku Asterisku se provede příkazem *database get <family> <key>* [4].

3.4.3 Mazání dat databáze

Pro odstranění dat se může použít funkce DB_DELETE. Následující aplikace nedělá nic, vypíše jen text v závorkách do příkazového řádku, a při tom funkce smaže hodnotu v databázi.

```
exten => 5493,1,Verbose(Hodnota byla: ${DB_DELETE(cislo/mobil)})
```

Tab. 3-14 - Odstranění dat z databáze pomocí funkce

Celou skupinu *family* lze také odstranit pomocí aplikace DBdeltree().

```
exten => 5494,1,DBdeltree(cislo)
```

Tab. 3-15 - Odstranění dat z databáze pomocí aplikace

Pro odstranění dat z AstDB prostřednictvím rozhraní příkazového řádku lze použít příkaz *database del <family> <key>* nebo *database deltree <family>* [4].

3.5 Asterisk Manager Interface AMI

Asterisk Manager Interface (AMI) je monitorovací a řídicí rozhraní Asterisku. To umožňuje online sledovat události (manager events), které se vyskytují v systému, ale také komunikovat s Asteriskem (manager actions). K dispozici je široké spektrum aplikací. Mnoho zajímavých bylo vytvořeno za účelem rozšíření funkcionality Asterisku. AMI aplikace je klientem ústředny Asterisk, kde je služba AMI spuštěna. Konfigurace se opět

provádí v konfiguračních souborech. Pro AMI je to soubor *manager.conf*, v kterém se především nastavují přístupová práva, ale i bezpečnostní a jiná nastavení [4].

3.5.1 Připojení AMI pomocí protokolu TCP

K rozhraní AMI se vnější aplikace připojují pomocí TCP socketu. Na příkladu si ukážeme připojení AMI pomocí služby telnet.

1. Připojení AMI přes TCP na portu 5038
2. Přihlášení akcí Login
3. Provedené příkazu (např. Ping)
4. Odhlášení a odpojení Logoff

```
$ telnet localhost 5038
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
Asterisk Call Manager/1.1
Action: Login
Username: strom
Secret: jedle
Response: Success
Message: Authentication accepted

Action: Ping
Response: Success
Ping: Pong
Timestamp: 9256398652.925531

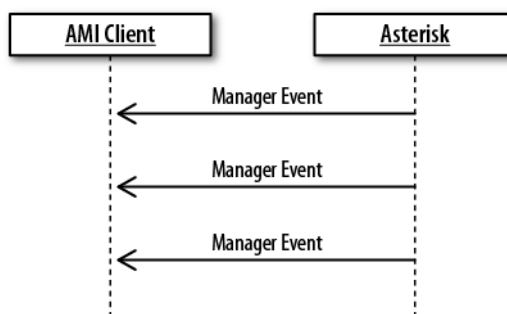
Action: Logoff
Response: Goodbye
Message: Thanks for all.
```

Tab. 3-16 - Ukázka komunikace AMI

3.5.2 AMI správce zpráv

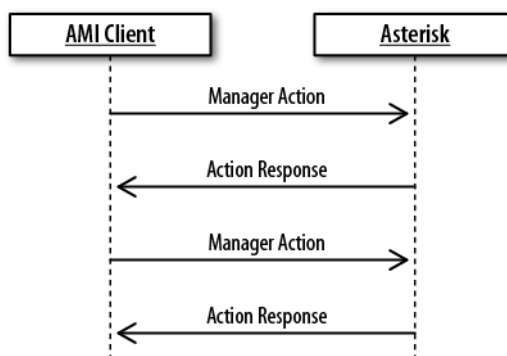
Existují dva hlavní typy zpráv pro AMI: správce událostí (manager events) a správce akcí (manager actions).

Správce událostí zobrazuje jednosměrné zprávy směřující z Asterisku k AMI klientovi a zobrazuje stav systému. Události jsou vyvolány samotnou ústřednou, např. probíhající volání (stav kanálu).



Obr. 3-8 Správce událostí AMI

Správce akcí rozšiřuje funkcionalitu správce událostí o schopnost události vyvolávat. Znamená to, že AMI klient odešle zprávu do Asterisku, ten ji zpracuje (provede žádanou akci) a vrátí zpět související událost AMI klientovi např. vytvoření kanálu (hovoru), získání dat z AstDB, atd.



Obr. 3-9 Správce akcí AMI

Asterisk reaguje jen na zprávy (příkazy), které zná. Dostupné příkazy lze zjistit v příkazovém řádku ústředny zadáním příkazu *manager show command*. Některé používané příkazy [4]:

- Command Provádí příkazy do CLI Asterisku
- Originate Aktivuje telefonní hovor
- Hangup Zavěsí kanál (ukončí hovor)
- Logoff Odhlásí klienta AMI
- SIPpeers Vrátí seznam SIP uživatelů

3.6 Asterisk Gateway Interface AGI

Asterisk dialplan se programuje vlastním jednoduchým programovacím jazykem. Avšak mnoho uživatelů, speciálně programátorů, preferuje svůj programovací jazyk. Použití vlastního programovacího jazyka usnadňuje integraci do jiného IT systému. AGI rozhraní

proto umožňuje vývojářům kontrolovat (ovládat) ústřednu Asterisk programovacím jazykem, dle vlastního výběru.

```
exten => 500,1,AGI(ahoj-svete.sh)
```

Tab. 3-17 - Volání jednoduchého AGI skriptu "Ahoj světe" z dialplanu

Existuje několik variant AGI, které se liší především v metodě komunikace s Asteriskem.

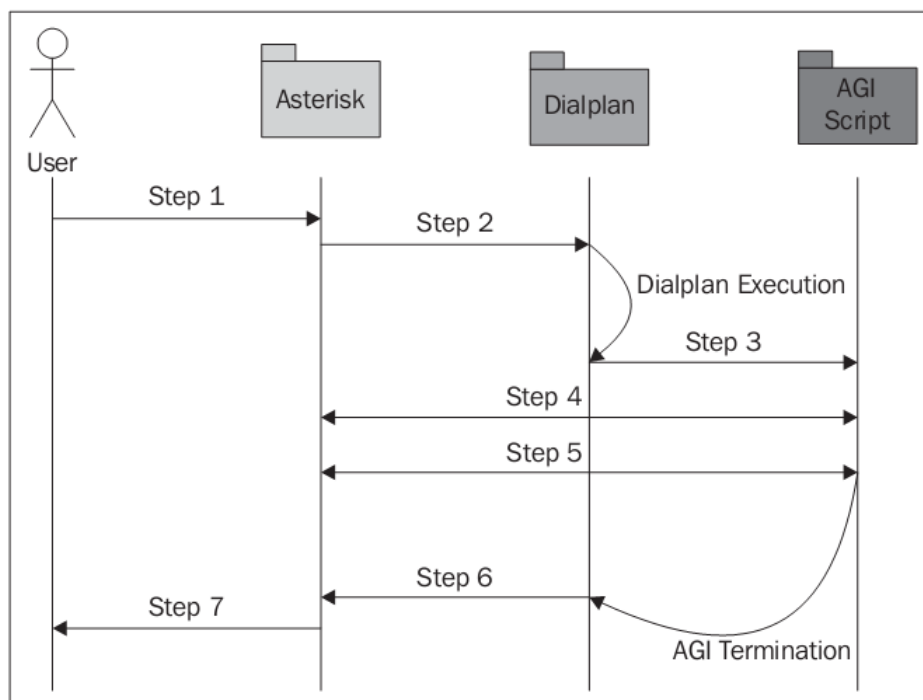
3.6.1 Základní varianta AGI

Nejjednodušeji varianta, zmíněná hned na začátku této kapitoly. AGI skript je spuštěn v dialplanu Asterisk. Zázpis syntaxe skriptu ukazuje Tab. 3-18. Komunikace mezi Asteriskem a aplikací se uskutečňuje přes *stdin* a *stdout* [4].

```
AGI(command[,arg1[,arg2[,...]])
```

Tab. 3-18 - Syntaxe AGI()

Tato varianta je nejméně efektivní formou AGI s ohledem na spotřebu zdrojů.



Obr. 3-10 Jak pracuje AGI

3.6.2 EAGI (Enhanced AGI)

EAGI oproti AGI poskytuje kromě *stdin* a *stdout* i jednosměrný audio tok.

3.6.3 DeadAGI

Starší verze AGI. Již se nepoužívá. Služba pracovala s již ukončeným hovorem v Asterisku označován jako *exten=>h*.

3.6.4 FastAGI AGI přes protokol TCP

FastAGI se používá pro AGI přes protokol TCP. Rozdíl proti předchozím variantám je, že pro nové spojení (další hovor), není otevřen nový proces. FastAGI má mnohem větší možnosti použití, ale složitější implementaci.

```
exten => 1234,1,AGI(agi://192.168.12.101,arg1,arg2,arg3)
```

Tab. 3-19 - FastAGI()

3.6.5 Asynchronní AGI kontrolované AMI

Async AGI—AMI-Controlled AGI je novější způsob použití AGI, který se používá od verze Asterisku 1.6.0. Účelem asynchronního AGI je asynchronně řadit požadavky (vytvářet frontu) požadavků AGI příkazů v kanálu. To může být užitečné, pokud se již v Asterisku používá rozhraní AMI a je požadavek použít vlastní programové skripty pro zpracování a řízení volání. Zápis syntaxe skriptu ukazuje tab. 3-17.

```
exten => 1234,1,AGI(agi:async)
```

Tab. 3-20 - Async AGI—AMI-Controlled AGI

3.7 Integrace relačních databází

AstDB je velice výhodná a výkonná databáze pro přímou práci v dialplanu Asterisku. Není to však jediná databáze, s kterou Asterisk spolupracuje. Asi nejnámější, dobře zdokumentované a výborně integrovatelné databáze jsou populární PostgreSQL a MySQL. Pro komunikaci mezi Asteriskem a databází se používá rozhraní ODBC. Integrace Asterisku s relačními databázemi je jedním ze základních stavebních kamenů pro vybudování velkých systémů. Schopnosti databáze umožní sdílení informací nebo integraci s webovými službami.

Konfigurace ODBC rozhraní se nachází v konfiguračním souboru *res_odbc.conf*. Soubor nastavuje parametry, pro různé moduly Asterisku, které se používají k připojení k databázi. Mezi související moduly, patří *cdr_odbc*, *cdr_adaptive_odbc*, *func_odbc*, *func_realtime*, *pbx_realtime*, *res_config_odbc* a *res_odbc* [4].

3.8 Webové služby pro spolupráci s ústřednou Asterisk

Všechny předešlé poznatky lze integrovat v uživatelsky příjemném prostředí. Aby bylo možné skripty spouštět a kontrolovat, musí s nimi nějaký programovací jazyk spolupracovat. Jedna z možností je použití jazyka PHP, který je bez potíží určen k použití s ústřednou Asterisk a jeho rozhraními, jako je AGI nebo AMI. PHP je přímo předurčeno pro spolupráci s HTML jazykem a odtud už je jen krůček k webovému rozhraní pro ovládání telefonní ústředny Asterisk. PHP skript v této práci je zpracován serverem Apache.

3.8.1 Jazyk HTML

HyperText Markup Language, označovaný zkratkou HTML, je značkovací jazyk pro hypertext. Je jedním z jazyků pro vytváření stránek v systému World Wide Web, který umožňuje publikaci dokumentů na Internetu [15]. Pomocí CSS stylů lze provádět formátování html stránek.

3.8.2 Jazyk PHP

PHP je skriptovací programovací jazyk. Je určený především pro programování dynamických internetových stránek a webových aplikací, například ve formátu HTML. PHP lze použít i k tvorbě konzolových a desktopových aplikací. Při použití PHP pro dynamické stránky jsou skripty prováděny na straně serveru – k uživateli je přenášén až výsledek jejich činnosti. Syntaxe jazyka je inspirována několika programovacími jazyky (Perl, C, Pascal a Java). PHP je nezávislý na platformě, rozdíly v různých operačních systémech se omezují na několik systémově závislých funkcí. Skripty lze většinou mezi operačními systémy přenášet bez jakýchkoliv úprav.

3.9 Asterisk Web Interface

Existuje několik verzí grafického uživatelského rozhraní (GUI) pro ústřednu Asterisk. Toto rozhraní umožňuje administrátorům zobrazit, upravit a změnit různé parametry Asterisku prostřednictvím webového rozhraní. Společnost Digium nabízí rozhraní *Asterisk-GUI*

napsané v javaskriptu. Bohužel není toto rozhraní podporováno novější verzí Asterisku (1.8 a vyšší). Existují ale i jiné GUI, jako je například *FreePBX*, *AsteriskNOW*, *Trixbox* nebo *PBX in a Flash* a další.

Rozhraní jsou určena k základní konfiguraci ústředny a uživatel nemusí mít příliš velké zkušenosti s konfigurací. Lze s nimi provést jen taková nastavení Asterisku, které jsou do tohoto konfiguračního rozhraní zahrnuta (proto je výhodné provádět skoro neomezenou konfiguraci pomocí konfiguračních souborů).

3.10 Další open VoIP SIP projekty

Ústředna Asterisk není jediné VoIP zařízení pro zpracování SIP protokolu. Na Internetu lze nalézt mnoho podobných open projektů.



Asi dva nejznámější Kamailio a OpenSIPS, vycházející z projektu SIP Express Router (SER). Konfigurační soubory mají stejnou strukturu, proto je lze mezi servery přenášet. Jedná se o velice výkonné SIP proxy servery, schopné zpracovat tisíce spojení za sekundu. Proto se často používají k vyrovnaní zátěže mezi servery. Mezi další vlastnosti patří: zabezpečení komunikace prostřednictvím TLS protokolu, LCR, stavový i bezstavový proxy server, účtování hovorů, napojení na Postgres, Oracle, MySQL, LDAP, atd.

Nevýhody těchto řešení jsou, že zpracovávají jen signalizační protokol SIP a neřeší zpracování přenosu medií (např. RTP), pracují jen s protokolem SIP a nelze je použít jako bránu mezi TDM a VoIP.

Proto se tyto servery často používají současně se softwarovou ústřednou Asterisk, kdy se vzájemně tato řešení doplňují.

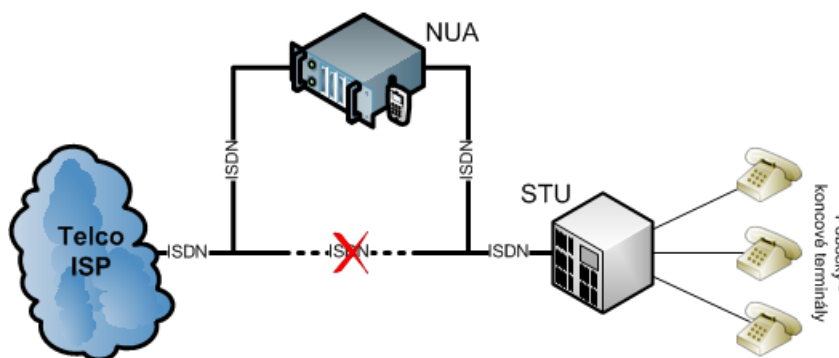
II. PRAKTICKÁ ČÁST

4 INTEGRACE VOIP SERVERU

Jak již bylo zmíněno v úvodu této práce, bude popis integrace VoIP serveru zaměřen na rozšíření analogového telefonního systému starší generace o VoIP možnosti a služby s tím související. Předchozí poznatky budou spojeny v jeden celek s ukázkou, jakým způsobem lze tuto integraci VoIP serveru provést. Toto řešení bude postaveno na softwarové telefonní ústředně Asterisk.

4.1 Popis zapojení systému

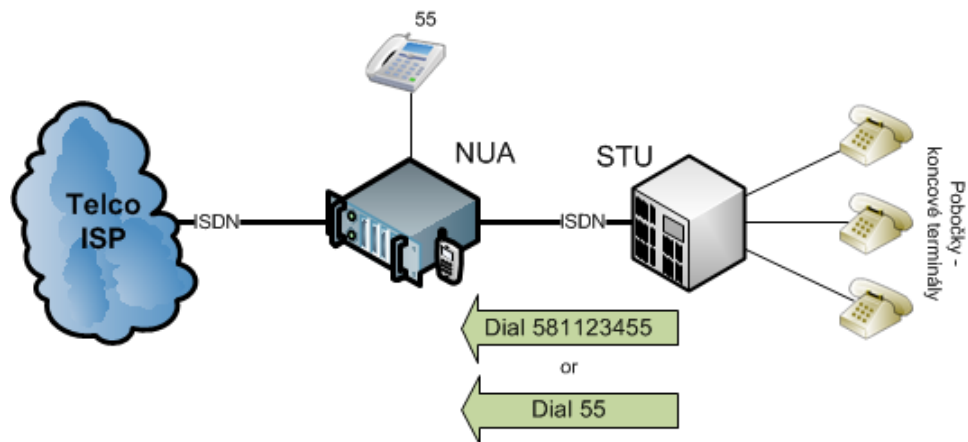
Rozšíření stávající telefonní ústředny (STU) o nové funkce a služby se provede jednoduše vložením nové VoIP telefonní ústředny Asterisk (NUA) mezi stávající telefonní ústřednu na porty vstupních linek a zakončení telefonních linek poskytovatele s vhodnou signalizací. Obvykle se jedná o linku ISDN (BRI, PRI, E1), což umožní plně integrovat NUA do nového telefonního systému. Více na Obr. 4-1.



Obr. 4-1 Zapojení NUA do nového telefonního systému

Např. STU má připojeno 40 poboček (koncových terminálů) a používá telefonní číslo 581 123 4xx s provolbovým blokem DDI ISDN se 100 čísly (tj. čísla 00 až 99). NUA s v tomto zapojení chová jako poskytovatel, to umožňuje pomocí provolby DDI volbu jakéhokoliv koncového účastníka STU. V tomto zapojení je samozřejmě vyřešeno i připojení (obsloužení) DDI provolby telekomunikačního operátora (připojené linky Telco ISP). Díky programovým možnostem NUA se systém chová jako původní ústředna, kde navíc rozšiřuje její možnosti. Konektivita telefonních poboček (koncových terminálů) STU pomocí DDI provolby nastavené v této ústředně je tedy zachována. Přirovnáním k síti LAN se z NUA stalo zařízení *bridge* na telefonní lince ISDN. Nyní lze k systému NUA připojit libovolné telekomunikační rozhraní. Zapojením byla získána kontrola nad telefonními linkami telekomunikačního operátora, ale i podřízeného systému STU. Číslovací plán telefonní přípojky nám dovoluje zřízení dalších 60 koncových terminálů

VoIP již na systému NUA. Telefonní spojení ze systému STU vyžaduje drobnou úpravu konfigurace číslovacího plánu, aby bylo možné volat koncové terminály ze STU do systému NUA. Pokud tato konfigurace není možná, může nastavení STU zůstat zachováno, jen přijdeme o komfort volby nových koncových terminálů systémem STU. Jako například uživatel STU volá pobočku 55. Bez úpravy konfigurace, musí uživatel STU volit 0 581 123 455 (0 = přestupný znak do sítě PSTN).



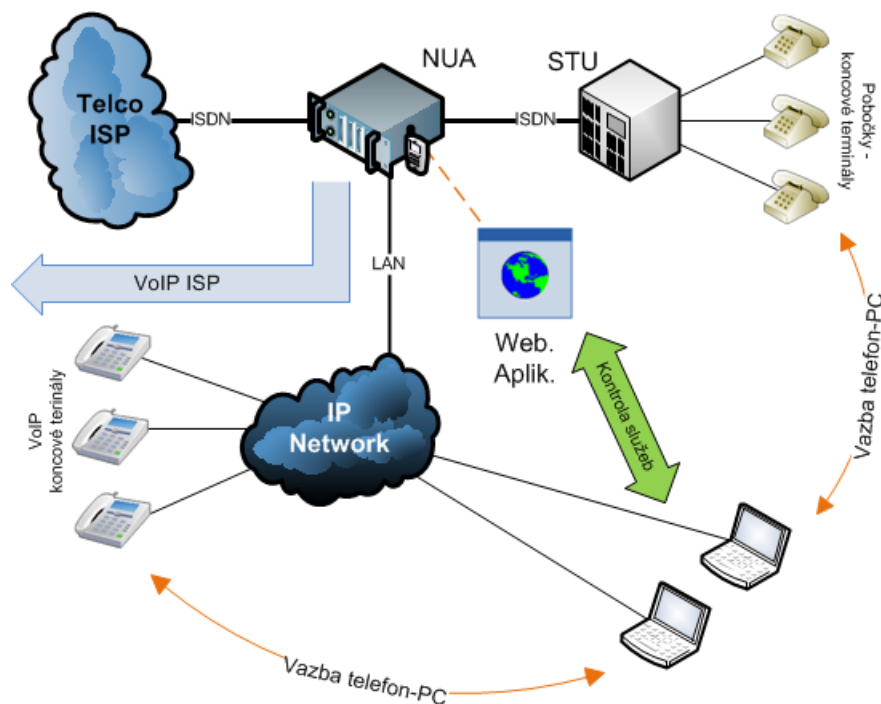
Obr. 4-2 Volba čísla 581123455 nebo 55

V systému NUA lze tedy akceptovat oba způsoby komunikace. Zpracování obou druhů volby se provede v dialplanu. Tab. 4-1 ukazuje možnost zpracování dlouhého čísla.

```
exten => _5811234XX,1,NoOp(Show ID: ${EXTEN}) ; Show ID: 581123455
exten => _5811234XX,2,NoOp(Show ID: ${EXTEN:7}) ; Show ID: 55
exten => _5811234XX,3,Dial(SIP/${EXTEN:7},,tT) ; Call to SIP/55
```

Tab. 4-1 - Úprava a použití čísla v národním formátu

Tímto způsobem zapojení je dosaženo plného ovládní STU. To umožňuje využití dalších služeb, které systém STU neposkytoval. Např. sestavení volání použitím webového integrovaného seznamu čísel, použití webového telefonu, volání přes VoIP operátora atd., viz Obr. 4-3.



Obr. 4-3 Úplná integrace STU a webové služby NUA

4.2 Stavební prvky systému NUA

Celý systém je sestaven z jednotlivých prvků, vzájemně propojených, kde jeden prvek může vyvolat interakci jiného prvku:

Telefonní ústředna – stará se o standardní komunikaci mezi všemi koncovými terminály a komunikačními linkami operátorů. Zajišťuje všechny hlasové služby.

Webový server – umožňuje spravovat (zobrazit, přidat a odstranit) databázový telefonní seznam, volit telefonní čísla pomocí webového prohlížeče pomocí funkce *click to call*.

VoIP SIP klient – ústředna se při sestavování jednotlivých hovorů chová jako SIP klient (UA), sestavuje jednotlivá požadovaná volání, a při dosažení spojení je vzájemně propojí.

Gateway telefonní komunikace – slouží jako brána (převodník) mezi různými druhy signalizací, protokolů a kodeků.

Databázový telefonní seznam – slouží ke správě hodnot telefonního seznamu pro telefonní ústřednu a webový server.

4.3 Webové služby, webové rozhraní

Webový server v této práci zajišťuje dvě funkce. První funkcí je integrovaný telefonní seznam, pomocí něhož lze spravovat databázi telefonních čísel a přiřazených jmen, ale i

kliknutím na ikonku telefonu sestavit volání mezi požadovaným účastníkem a telefonem iniciátora volání. Tento seznam dokáže také přiřadit k příchozímu volanému číslu jmenný identifikátor (jméno na displeji telefonu). Druhou funkcí webového serveru je webový telefon, kde zadáním čísla volaného do určeného pole formuláře a potvrzením opět dojde ke spojení mezi požadovaným číslem a iniciátorem volání.

4.3.1 Integrovaný telefonní seznam

Webová stránka telefonního seznamu je tvořena jazykem HTML a PHP. Statické řešení stránky HTML se stará jen o vzhled samotného webu. V první části telefonního seznamu se nachází formulářové pole pro zadání hodnot do seznamu. Pod touto částí se nachází výpis samotného telefonního seznamu generovaného jazykem PHP. Data jsou uložena v databázovém systému ústředny Asterisk AstDB a komunikace mezi web serverem a Asteriskem je zajištěna pomocí AMI. Výměna dat mezi formulářem a databází probíhá otevřením socketu, výměnou dat, ukončením (zavřením) socketu a zpracování načtených dat.



The screenshot shows the VoIPMachine web interface. At the top, there is a navigation bar with 'Home', 'Intranet', 'Záznam hovorů', and 'Help' on the left, and 'Přihlášení' and 'Odhlášení' on the right. The main content area is divided into a left sidebar with menu items like 'Základní služby', 'Telefonní seznam', and 'Web telefon', and a main panel. The main panel contains a form with three input fields: 'Číslo rychlovolby:', 'Telefonní číslo:', and 'Jméno stanice:', followed by an 'Uložit' button. Below the form, a table displays a list of contacts with columns for 'Rychlovolba', 'Číslo', 'Jméno', 'Volat', 'Editace', and 'Smazat'.

Rychlovolba	Číslo	Jméno	Volat	Editace	Smazat
8001	5646131531	dsfvghshdf			
8002	261301333	Servis 2N			
8003	724269926	Ludik mobil			
8004	5454564212	Pokus4527			
8005	687434654	dsjbfdsfjnhdsjbsv kdsvj			
8007	68546549684	hfvdsfhvdfjv			
8008	3213213213	dsgdfgfdfgd			
8010	23123131	DSFfdgdfDFG			

Obr. 4-4 Integrovaný telefonní seznam

Telefonní webový seznam obsahuje 6 sloupců hodnot:

- Rychlovolba – obsahuje indexové číslo záznamu, které lze použít i pro volbu telefonního čísla pomocí zkrácené volby
- Číslo – telefonní číslo
- Jméno – jméno stanice přiřazené k telefonnímu číslu

- Volat – ikona telefonu slouží k sestavení volání na požadované telefonní číslo
- Editace – umožňuje editovat záznam telefonního seznamu
- Smazat – odstraní záznam telefonního seznamu

4.3.1.1 Struktura dat v AstDB telefonního seznamu

Data telefonního seznamu jsou uložena v AstDB, kdy je jednou v hodnotě $\langle family \rangle$ $\langle key \rangle$ uloženo telefonní číslo, podruhé telefonní jméno. Tímto způsobem jsou uloženy v databázi tři hodnoty: rychlovolba, telefonní číslo, jméno stanice. Na příkladu je ukázáno uložení čísla „800123456“ se jménem „O2 telefonica“ pod index (rychlovolbu) „8888“. Skutečným klíčem databázového záznamu je telefonní číslo.

$\langle family \rangle$	$\langle key \rangle$	$\langle value \rangle$
speeddial	8888	800123456
spedname	800123456	O2 telefonica

Tab. 4-2 - Závislosti klíčů databáze

4.3.1.2 Zobrazení dat telefonního seznamu

Telefonní seznam načítá data ze systému NUA a zobrazuje je v HTML tabulce. Programový kód zobrazení telefonního seznamu je uložen v souboru `../speed/speed.php`.

```

$socket = fsockopen("127.0.0.1","5038", $errno, $errstr, $timeout);
    // Otevreni spojeni AMI pres soket
fputs($socket, "Action: Login\r\n"); // Akce prihlaseni
fputs($socket, "UserName: $man_name\r\n"); // Vlozeni uziv. jmena
fputs($socket, "Secret: $man_pword\r\n\r\n"); // Vlozeni uziv. hesla
fputs($socket, "Action: Command\r\n"); //Zadavani prikazu do Aster.
fputs($socket, "Command: database show speeddial\r\n\r\n"); // Spec. dotaz
na vycteni databaze
fputs($socket, "Action: Logoff\r\n\r\n"); // Odhlaseni od ustredny

while (!feof($socket)) {
    $dbentries1 .= fread($socket, 8192); // Cteni vstupu ze soketu a
ulozeni do promenne $dbentries1
}
$array1 = preg_split("/\n/", $dbentries1, -1, PREG_SPLIT_NO_EMPTY);
    // Rozdeleni prom. $dbentries1 do pole řetěz. $array1 dle parametru
for ($i = 1; $i < 7; $i++) {
    array_shift($array1); // Ostraneni prvnych "i" radku bez info.
}
fclose($socket); // Zavreni soketu

```

Tab. 4-3 - Programový kód – komunikace ústřednu pomocí soketu

Při běhu skriptu nejprve dojde k otevření soketu. IP adresa 127.0.0.1 (localhost) signalizuje provádění skriptu na serveru Apache, který je na stejném fyzickém stroji, jako je spuštěn i

server Asterisk. Použit je standardní port AMI 5038. Následuje akce přihlášení a vložení jména a hesla. Přihlašovací informace jako je port, jméno, heslo a uživatelská práva jsou uloženy v konfiguračním souboru Asterisku *manger.conf*.

```
[general]
enabled = yes
port = 5038
bindaddr = 0.0.0.0

[john]
secret = password
read = system,call,log,verbose,command,agent,user
write = system,call,log,verbose,command,agent,user
```

Tab. 4-4 Ukázka konfiguračního souboru *manager.conf*

Po proceduře přihlášení dojde k odeslání akce a příkazu na server Asterisk. Jedná se o akci *Command* a příkaz *database show speeddial*. *Speeddial* je název hodnoty *<family>* databáze nesoucí všechna dostupná data o telefonních číslech seznamu. V dalším kroku proběhne odpojení od Asterisku a uložení dat ze soketu do proměnné *\$dbentries1*. Data obsahují mnoho různých hodnot a nejsou odřádkována. Nepotřebné údaje se odstraní, potřebné se uloží jako prvky pole *\$array1*.

```
Asterisk Call Manager/1.1 Response: Success Message: Authentication
accepted Event: FullyBooted Privilege: system,all Status: Fully Booted
Response: Follows Privilege: Command /speeddial/8001 : 564613153
/speeddial/8002 : 261301333 /speeddial/8003 : 724269926 /speeddial/8004 :
554564212 /speeddial/8005 : 687434654 /speeddial/8007 : 685469684
/speeddial/8008 : 321321321 /speeddial/8010 : 230123131 /speeddial/8011 :
0013215456321231 /speeddial/8012 : 001371765763715 /speeddial/8013 :
746225646 /speeddial/8014 : 606639778 /speeddial/8282 : 987965313
/speeddial/8324 : 436541656 /speeddial/8888 : 800123456 /speeddial/8998 :
816411651 16 results found. --END COMMAND-- Response: Goodbye Message:
Thanks for all the fish
```

Tab. 4-5 - Obsah proměnné *\$dbentries1*

```
/speeddial/8888 : 800123456
```

Tab. 4-6 - Obsah pole proměnné *\$array1[18]*

Tento krok se opakuje, i pro získání telefonních jmen. Hodnota *<family>* se v tomto případě jmenuje *speedname*. Údaje se ukládají do proměnné *\$dbentries2*, a po zpracování do proměnné *\$array2*. Dalším krokem je z uložených polí *\$array1*, *\$array2* získat požadované číslo, jméno a index.

```

for ($a=$start_a;$a<=$stop_a;$a++) // Citac x hodnot dle „$stop_a“
{
$speed_no_line=$array1[$a]; // Nacteni radku x z pole dle „$a“
if (ereg("$Result.*",$speed_no_line, $status_dial)) {
//Jestlize = "result" v prohled. radcich tak stop podminky
$a=$stop_a;// Nastaveni end hodnoty pro ukoceni
print "Data zpracována.";
}
else {
$cut_line = explode ("/",$speed_no_line);
// Rozdel. retezce na casti pres „/“, vystup do pole
ereg("[0-9][0-9][0-9][0-9]", $cut_line[2], $out_no);
// Vyber druhé casti pole z $cut_line, jen ctyrmistne cislo
ereg("[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]*",
$cut_line[2], $out_dial_no);
// Vyber tel. cisla delky > 9 znaku, ulozeni do $out_dial_no

for ($x=0;$x<999;$x++){// Hledani priraz. jmena k tel číslu
$speed_name_line=$array2[$x]; // Jmeno z rad. pole $array2
$cut_name_line = explode ("/",$speed_name_line);
// Rozdel. retezce na casti pres „/“, vystup do pole
$text_name = $cut_name_line[2];
// Ulozeni promene radku z promenne "$cut_name_line"
$result_name = strstr($text_name, $out_dial_no[0]);
// Hledani hodnoty tel. cisla v "text_name"
if ($result_name > "") { // Jestlize li je vysledek vetsi jak 0
$result_name_array[$a]=$result_name;
// Ulozeni vysledku radku s jmenem
}
}
ereg(":[A-z].*", $result_name_array[$a], $out_name_1);
// Uprava nalezene hodnoty pred zobrazenim
ereg("[A-z].*", $out_name_1[0], $out_name);

```

Tab. 4-7 - Skript pro získání požadovaných hodnot

Skript začne cyklem *for{}*, který funguje jako čítač, a prochází jednotlivé hodnoty pole *\$array1*. Maximální počet zobrazených řádků určuje proměnná *\$stop_a*. Pomocí regulárních výrazů a příkazů pro práci s textem v PHP se získá:

1. hodnota indexu, představující pořadové číslo a číslo rychlovolby
2. hodnota telefonního čísla
3. vnořený cyklus *for{}* vyhledá ke konkrétnímu číslu telefonní jméno.

Proměnná	Hodnota
\$out_no	Index, rychlovolba
\$out_dial_no	Telefonní číslo
\$out_name	Telefonní jméno

Tab. 4-8 - Závislost proměnné a hodnoty

Zajímavé je ukončení běhu skriptu. Objeví-li se při zpracování textového řetězce slovo „END“ nebo „results“ (závisí od verze Asterisku) v proměnné *\$Result*, znamená to, že bylo dosaženo konce seznamu a vyhodnocení bylo ukončeno. Než se prvotní cyklus *for{}* inkrementuje (tj. zvýší svou hodnotu o 1), musí dojít k tisku získaných hodnot do HTML tabulky.




```

if ($a/2 != floor($a/2)) { // Test lichého čísla na obarvení radku
    $color="#E6E6E6"; // liche číslo
}
else {
    $color="#A5B9FE"; // sude číslo
}
print "<tr>\n";
print "<td bgcolor=\""$color\""><div
align=\"center\"><b>$out_no[0]</b></div></td>";
print "<td bgcolor=\""$color\""><b>$out_dial_no[0]</b></td>";
print "<td bgcolor=\""$color\""><b>$out_name[0]</b></td>";
print "<td width=\"60\" bgcolor=\""$color\""><b><div
align=\"center\">
<a href=\"../phone/phone_dial.php?phonenumber=$out_dial_no[0]\">
<img src=\"../img/contact16.png\" alt=\"edit\" title=\"Volat\" />
</a></b></div></td>";
print "<td width=\"60\" bgcolor=\""$color\""><div align=\"center\">
<a href=\"speed.php?speednumber=$out_no[0]&
phonenumber=$out_dial_no[0]&phonename=$out_name[0]\">
<img src=\"../img/edit16.png\" alt=\"edit\" title=\"Editovat\" />
</a></div></td>";
print "<td width=\"60\" bgcolor=\""$color\""><div align=\"center\">
<a href=\"speed_delete.php?speednumber=$out_no[0]&
phonenumber=$out_dial_no[0]&phonename=$out_name[0]\"
onclick=\"return confirm('Opravdu smazat záznam $out_no[0]?')\">
<img src=\"../img/delete16.png\" alt=\"Smazat\" title=\"Smazat\"
/></a></div> </td>";
print "</tr>\n";
} //End else
} //End for

```

Tab. 4-9 - Tisk hodnot do HTML tabulky

Hodnoty nesoucí užitečnou informaci, jsou v Tab. 4-9 zvýrazněny tučně, ostatní údaje slouží jen ke grafické úpravě a zobrazení ikoněk.

8888	800123456	O2 telefonica			
------	-----------	---------------	---	---	---

Obr. 4-5 Příklad jednoho řádku telefonního seznamu HTML

Takové zpracování dat přes jednu (respektive dvakrát) otevřený socket s následným zpracováním textových řetězců je mnohem efektivnější, než načítání jednotlivých hodnot z databáze, které obnáší otevření a zavření socketu. Např. při telefonním seznamu o 1000 hodnotách by muselo proběhnout 2000 otevření a zavření socketu, která by neúměrně zvyšovala zátěž serveru.

4.3.1.3 Uložení dat telefonního seznamu

Pro odeslání dat do skriptu pro uložení údajů telefonního seznamu, slouží jednoduchý HTML formulář. Formulář je uložen v souboru `../speed/speed.php`.

```
<form action="../speed/speed_status.php" method="post" >
  <table width="408" cellpadding="2">
    <tr>
      <td width="184"><strong> Číslo rychlovolby: </strong></td>
      <td width="208"><input name="SpeedNumber"
        value="<?php echo $_GET["speednumber"];?>"
        size="30" />
      </td>
    </tr>
    <tr>
      <td><strong> Telefonní číslo: </strong></td>
      <td><input name="PhoneNumber"
        value="<?php echo $_GET["phonenumber"];?>"
        size="30" />
      </td>
    </tr>
    <tr>
      <td><strong>Jméno stanice:</strong></td>
      <td><input name="PhoneName"
        value="<?php echo $_GET["phonename"];?>"
        size="30" />
      </td>
    </tr>
    <tr>
      <td></td>
      <td>
        <div align="right">
          <strong><input type="submit" value="Uložit" /></strong>
        </div>
        <div align="right"></div>
      </td>
    </tr>
  </table>
</form>
```

Tab. 4-10 - HTML formulář pro uložení dat telefonního seznamu

V úvodu skriptu jsou nastaveny parametry odesílání formuláře metodou POST. Údaje z formulářových polí jsou předány hodnotou `<input name>`. Předávají se hodnoty *SpeedNumber*, *PhoneNumber* a *PhoneName*. Hodnota *value* těchto polí obsahuje krátký PHP skript pro vyplnění pole při editaci záznamů. Princip editace bude vysvětlen následně. Stisknutím odesílacího tlačítka se předají výše zmíněné hodnoty do skriptu umístěném v souboru `/speed/speed_status.php`.

```

$BackLink = $_SERVER['HTTP_REFERER'];
// Ziskani dat z formulare metodou POST
$PhoneNumber = $_POST['PhoneNumber'];
$PhoneName = $_POST['PhoneName'];
$SpeedNumber = $_POST['SpeedNumber'];

if ($PhoneNumber <> "" && $PhoneName <> "" && $SpeedNumber <> "")// Test
{
    $socket = fsockopen("127.0.0.1","5038", $errno, $errstr, $timeout);
    fputs($socket, "Action: Login\r\n");
    fputs($socket, "UserName: $man_name\r\n");
    fputs($socket, "Secret: $man_pword\r\n\r\n");
    fputs($socket, "Action: Command\r\n");
    // Akce zadavani prikazu do Asterisku
    fputs($socket, "Command: database put speeddial $SpeedNumber
$PhoneNumber\r\n\r\n");
    fputs($socket, "Action: Command\r\n");
    fputs($socket, "Command: database put speedname $PhoneNumber
\"$PhoneName\"\r\n\r\n");
    fputs($socket, "Action: Logoff\r\n\r\n");

    while (!feof($socket)){
        $response .= fread($socket, 8192);
        // Cteni vstupu ze soketu a ulozeni do promenne $response
    }
    fclose($socket);// Zavreni soketu socket
    if (strpos($response, 'Updated database successfully')) {
        // Kontrola uspesne odpovedi od Asterisku
        print("<p><strong>Vaše data byla úspěšně uložena do
databáze.</strong></p>\r\n");
        print("<p><h2><a href=\"\$BackLink\">Zpět</a></h2</p>");
    }
    else print(" Error....");
}
else{ // Nejsou li zadana spravna vstupni data
    print "<p><strong>Někde je chyba!</strong></p>\n";
    print "<p><strong>Prosíme zkontrolujte správné vstupní
údaje!</strong></p>";
    print("<p><h2><a href=\"\$BackLink\">Zpět</a></h2</p>");
}

```

Tab. 4-11 - Skript `speed_status.php` pro uložení hodnoty do databáze

Metodou POST jsou předány hodnoty, které byly v polích formuláře, a následně uloženy do proměnných. Po kontrole, jestli není některá proměnná prázdná, dojde k otevření socketu a spuštění příkazu Asterisku *Command* k uložení dat do databáze dle struktury dat AstDB:

Command: database put speeddial \$SpeedNumber \$PhoneNumber\r\n\r\n

Command: database put speedname \$PhoneNumber \"\$PhoneNumberName\"\r\n\r\n

Je-li příkaz vykonán, pokračuje skript k uzavření socketu, a ke kontrole kladné odpovědi Asterisku. Podle výsledku podmínek se tiskne na obrazovku úspěch, popřípadě neúspěch akce. Jako doplněk je na stránce tlačítko zpět, jehož hypertextový odkaz ukazuje na předchozí stránku pomocí systémové proměnné PHP \$_SERVER['HTTP_REFERER'].

4.3.1.4 Editace dat telefonního seznamu

Editace telefonního seznamu je založena na jednoduchém principu předání všech dříve zobrazených dat (všech proměnných) pod obrázek editace (obrázek tužky) jako hypertextový parametr. Jedná se o vlastnost HTML, ale v tomto případě zpracovávána PHP skriptem příkazem *print*. Při kliknutí na tento obrázek se automaticky vyplní políčka HTML formuláře. V Tab. 4-12 jsou předávané proměnné zvýrazněny tučně a skript je odřádkován, aby byl čitelnější, ve skutečnosti je vše na jednom programovém řádku.

```
print "
<td width=\"60\" bgcolor=\"\$color\">
<div align=\"center\">
<a href=\"speed.php?
speednumber=\$out_no[0]&
phonenummer=\$out_dial_no[0]&
phonename=\$out_name[0]\">
<img src=\"../img/edit16.png\" alt=\"edit\" title=\"Editovat\" />
</a>
</div>
</td>";
```

Tab. 4-12 - Kód tlačítka *edit*

4.3.1.5 Mazání dat telefonního seznamu

Při stisku tlačítka pro smazání (obrázek křížku) na základní stránce s telefonním seznamem dojde k několika akcím. První akce spustí mini skript *onclick*, který zobrazí okno s dotazem pro potvrzení smazání položky. Druhá akce je obdobná jako u tlačítka editace, s tím rozdílem, že se proměnné předávají metodou GET do souboru *../speed/speed_delete.php*.

```

print "
<td width=\"60\" bgcolor=\"\${color}\">
<div align=\"center\">
<a href=\"speed_delete.php?
speednumber=\$out_no[0]&
phonenumber=\$out_dial_no[0]&
phonename=\$out_name[0]\"
onclick=\"return confirm('Opravdu smazat záznam \$out_no[0]?')\">
<img src=\"../img/delete16.png\" alt=\"Smazat\" title=\"Smazat\" />
</a>
</div>
</td>";

```

Tab. 4-13 - Kód tlačítka *delete*

Předaná data jsou v souboru `../speed/speed_status.php` zpracována obdobně jako při uložení dat. Použitá metoda předání dat je GET.

```

$PhoneNumber = $_GET['onenumber'];
$PhoneNumber = $_GET['onenumber'];
$SpeedNumber = $_GET['speednumber'];

if ($PhoneNumber <> "" && $PhoneNumber <> "" && $SpeedNumber <> ""){
    $socket = fsockopen("127.0.0.1","5038", $errno, $errstr, $timeout);
    fputs($socket, "Action: Login\r\n");
    fputs($socket, "UserName: $man_name\r\n");
    fputs($socket, "Secret: $man_pword\r\n\r\n");
    fputs($socket, "Action: Command\r\n");
    fputs($socket, "Command: database del speeddial $SpeedNumber
\r\n\r\n"); // Mazani cisla
    fputs($socket, "Action: Command\r\n");
    fputs($socket, "Command: database del speedname $PhoneNumber
\r\n\r\n"); // Mazani jmena
    fputs($socket, "Action: Logoff\r\n\r\n");

    while (!feof($socket)){
        $response .= fread($socket, 8192);
    }
    fclose($socket);
    if (strpos($response, 'Database entry removed')){
        print("<p>Data byla úspěšně vymazána.</p>\r\n");
        print("<p><h2><a href=\"\${BackLink}\">Zpět</a></h2></p>");
    }
    else print("Error....");
}
else {
    print "<p><strong>Někde je chyba!</strong></p>";
    print("<p><h2><a href=\"\${BackLink}\">Zpět</a></h2></p>");
}

```

Tab. 4-14 -Skript *speed_delete.php* pro smazání hodnoty z databáze

Po otevření socketu, příkaz Asterisku *Command* v tomto skriptu dle struktury AstDB smaže data:

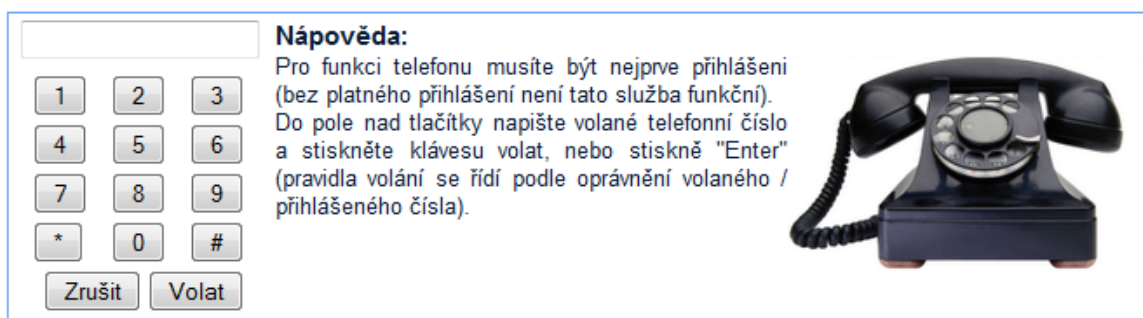
Command: database del speeddial \$SpeedNumber

Command: database del speedname \$PhoneNumber

Následuje zavření soketu a kontrola úspěšnosti s vypsáním výsledku mazání dat.

4.3.2 Web telefon

Web telefon je aplikace, která umožňuje propojit koncový terminál uživatele s požadovaným volaným číslem. Nezáleží, zdali je koncový uživatel systému STU nebo NUA.



Obr. 4-6 Webový telefon

Volba čísla se provádí buď pomocí formuláře web telefonu, nebo stiskem ikony telefonu na telefonním seznamu u požadovaného čísla. Telefon je navržen tak, aby se nejprve spojil s koncovým terminálem iniciátora volání. Jakmile ten hovor přijme, začne telefonní systém sestavovat spojení na požadované číslo. Aby systém věděl, který iniciátor volání požaduje, musí se nejdříve systému sdělit svou identifikaci. Pro přihlášení slouží odkaz „Přihlášení“ v pravém horním rohu, zobrazený na všech web stránkách telefonního serveru.

Obr. 4-7 Přiřazení web telefonu ke koncovému terminálu uživatele

Přihlášení je v podstatě služba „cookies“, která uloží data z proměnné *Exten* předanou metodou POST z přihlašovacího formuláře do proměnné *\$ExtenCookie*. Proměnná zůstane uložena až do doby zavření prohlížeče (dobu trvání lze modifikovat - *\$expire*).

```
<?php
    $ExtenCookie = $_POST['Exten'];
    setcookie("ExtenCookVar", $ExtenCookie,$expire = 0, $path = '/');
?>
```

Tab. 4-15 - Uložení cookies s ID uživatele

Po úspěšném přihlášení lze již web telefon plně používat. V následujícím výpisu kódu bude vysvětlen princip sestavení volání. Jako v předchozích popisech komunikace s Asteriskem, je opět otevřen soket, provedeny příkazy a zavřen soket. Nyní se zaměříme na samotnou akci Asterisku. Pro přehlednost bylo odstraněno co nejvíce PHP kódu.

```
Action: Login
UserName: $man_name
Secret: $man_pword
Action: Originate
Channel: local/$ExtenCookie@$LocContext
Context: $VarContext
Exten: $PhoneNumber
Priority: 1
Timeout: 30000
Variable: var1=$PhoneNumber
Callerid: $PhoneNumber
Action: Logoff
```

Tab. 4-16 - Akce Asterisku, sestavení spojení

Metodou GET je předána proměnná nesoucí informaci o telefonním čísle. Tato proměnná může být odeslána z web telefonu, ale i z telefonního seznamu. Je uložena v proměnné *\$Phonenumber*, což je požadovaná volba čísla. Po otevření soketu je odeslán do Asterisku příkaz pro ověření uživatele a následuje akce *Originate* („založení hovoru“). Tato akce spustí sestavování telefonního hovoru. Akce *Originate* má několik voleb:

Channel: local/\$ExtenCookie@\$LocContext	Volá lokálním kanálem iniciátorovi volání.
Context: \$VarContext	Kontext pro odchozí hovor (nastavení oprávnění).
Exten: \$PhoneNumber	Požadované volené číslo.
Priority: 1	Priorita v kontextu volání.
Timeout: 30000	Čas pro vyzvednutí volaného.
Variable: var1=\$PhoneNumber	Pomocná proměnná v dialplanu „var1“.
Callerid: \$PhoneNumber	Nastavení identifikace volaného.

Tab. 4-17 - Volby akce *Originate*

\$ExtenCookie	Hodnota čísla volajícího (iniciátora volání) koncového terminálu
\$LocContext	Informace o kontextu kde začne spojení volajícího (iniciátora volání)
\$VarContext	Nastavení kontextu pro odchozí hovor (lze nastavit oprávnění hovoru)
\$PhoneNumber	Hodnota čísla volaného

Tab. 4-18 - Proměnné v akci *Originate*

Akce volání začne sestavením hovoru na iniciátora volání (koncový terminál), pomocí lokálního kanálu Asterisku. Například uživatel 333 bude volat z webového telefonu na číslo 800123456. Kanál spojení, při dosazení proměnných, bude vypadat takto:

Channel: local/333@\$local_ami

Znamená to tedy, že se v kontextu *local_ami* bude hledat *exten=333*. V dialplanu tomuto pravidlu odpovídá *_XXX*. V tomto kontextu dojde k úpravě SIP hlavičky (bude popsáno dále) a skoku do kontextu, kde lze provést volání na stanici 333 (*level_0* – *macro-voipUA* – *CallerDB* – *Dial SIP/333*).

```
[local_ami]
;=====
exten => _XXX,1,SIPAddHeader(Call-Info: answer-after=1)
exten => _XXX,n,Goto(level_0,${EXTEN},1)

[level_0]
;=====
exten => _XXX, hint, SIP/${EXTEN}
exten => _XXX,1,Macro(voipUA,${EXTEN})
exten => _39X,1,Goto(services,${EXTEN},1)
exten => _X.,1,Macro(outgoing,${EXTEN})

[macro-voipUA]
;=====
exten => s,1,NoOp(2, Dial user: SIP/${ARG1})
    same => n,Gosub(CallerDB,s,1)
    same => n,Dial(SIP/${ARG1},,tT)

[macro-outgoing]
;=====
exten => s,1,NoOp(4, Calling ${ARG1})
    same => n,Dial(DAHDI/g0/${ARG1},,L(900000))
    same => n,Hangup()

[CallerDB]
;=====
exten => s,1,NoOp(3, CALLERID number: ${CALLERID(num)})
exten => s,n,NoOp(3, CALLERID name: ${CALLERID(name)})
exten => s,n,NoOp(3, AstDB speedname: ${DB(speedname/${CALLERID(num)})})
exten => s,n,Set(CALLERID(name)=${DB(speedname/${CALLERID(num)})})
exten => s,n,NoOp(3,new CALLERID name: ${CALLERID(name)})
exten => s,n,Return()
```

Tab. 4-19 - Dialplan Asterisku pro volání z web rozhraní

Než iniciátor volání hovor přijme, nastaví se identifikace volaného pomocí kontextu *CallerDB*. Znamená to, zobrazení požadované volané stanice, včetně jména, na displeji koncového terminálu uživatele, který hovor vyvolal. V tomto příkladu se na displeji stanice 333 objeví číslo 800123456 a jméno *O2 Telefonica*. Při tomto hovoru, dojde k druhému kroku akce *Originate* a to již spojení na požadované číslo. Začátek volání v dialplanu Asterisku určuje kontext *\$VarContext*, což v tomto příkladu znamená kontext *level_0*. V kontextu *level_0* se tedy bude hledat *exten=80012345*. V dialplanu tomu odpovídá pravidlo *_X*. Pokud nedojde po časovou dobu nastavenou parametrem *Timeout: 30000 (30s)* k vyzvednutí hovoru, všechny předchozí akce se ukončí.

4.3.3 Zpětné volání - Call back

Výše popsáný způsob sestavení hovoru lze použít i jako funkci zpětného volání. Například když iniciátor volání může být potencionální zákazník, který u zajímavé nabídky vyplní své telefonní číslo a odešle jej. Ihned začne zvonit telefon obchodníkovi, a ten po vyzvednutí již slyší vyzváněcí tón při sestavování spojení na potencionálního zákazníka. Služba v tomto podání by obnášela jen změnu registrace obchodníka, jako příjemce zpětného volání a pár drobných úprav programového kódu web stránek.

4.3.4 Speciální konfigurační soubor

Protože tato práce řeší implementaci skoro k jakémukoli systému Asterisk a vývoj této softwarové ústředny jde stále dopředu, bylo potřeba vytvořit konfigurační soubor, pomocí něhož se nastaví nějaké globální parametry (konkrétně proměnné) k dané HW a SW instalaci systému, a logovací údaje k systému AMI. Tento soubor je v kořenovém adresáři webového rozhraní a nazývá se *evoma_conf.inc* a propojení s PHP skripty se provádí příkazem `include "../evoma_conf.inc";`. Popis jednotlivých položek je v komentářích.

```
// Pro ASTERISK 1.6 (0-999) pro ASTERISK 1.8 (4-1004)
$start_a = 4;
$stop_a = 1004;

// Rozsah pobocek celého systému(cislovaci plan)
$start_exten = 330;
$stop_exten = 399;

// Nastaveni kontextu $VarContext
$VarContext = "level_1";

// Nastaveni lokalniho kontextu $LocContext
$LocContext = "local_ami";

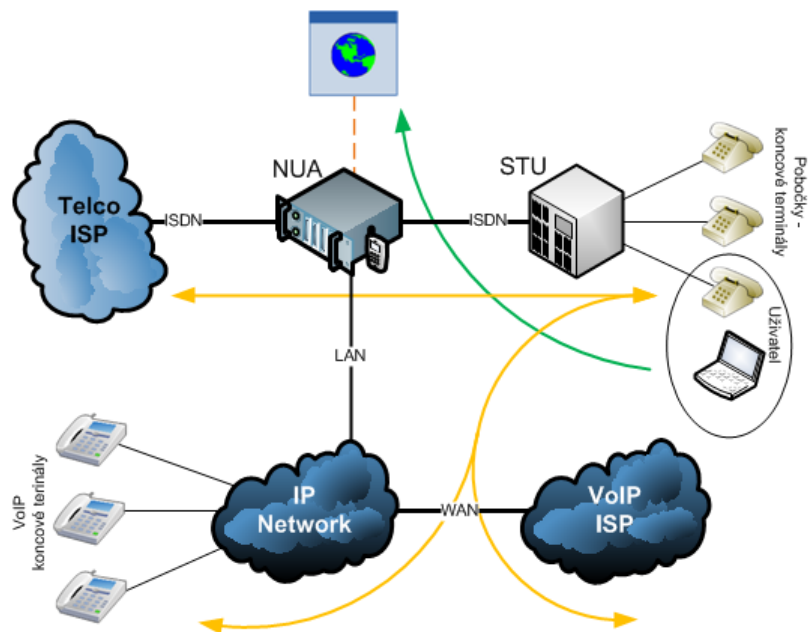
// Nastaveni hesla a jmena pro pripoeni AMI
$man_name = "yourname";
$man_pword = "yourpswd";

//Vyhodnoceni konce retezce pro 1.4 = "END", pro 1.6 a vyssi = "results"
$Result="results";
```

Tab. 4-20 - Soubor *evoma_conf.inc*

4.4 Příklad integrace systému I – analogový uživatel

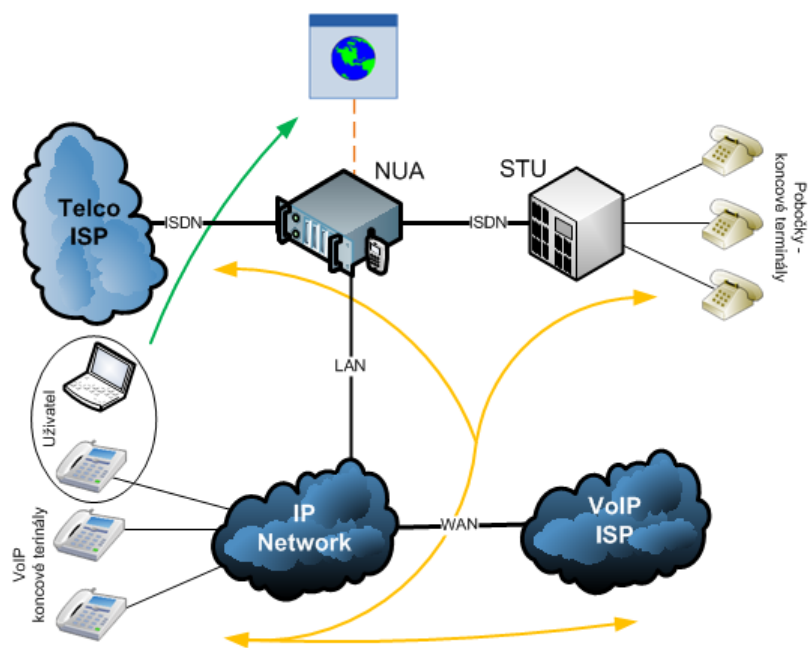
Analogovému uživateli zůstala možnost volat do TDM sítí. Má rozšířenou schopnost volání i na nové terminály systému VoIP a volání do sítí VoIP. Dostupnost při příchozím volání je stejná a rozšířená o VoIP. Díky tomuto systému je možno využívat všechny standardní telefonní služby, jako jsou ID volajícího (umožňuje-li to koncový terminál), přesměrování, přepojení apod. Rozšířili se možnosti o nové služby jako integrovaný telefonní seznam, kliknutím volat, webový telefon, zpětné volání apod. Díky možnostem ústředny Asterisk není problém jakékoli další služby do telefonní ústředny implementovat. Webové služby uživatele jsou propojeny i s STU takovým způsobem, že uživatel provede do webového rozhraní registraci svého telefonu, a NUA se postará o sestavení volání i na jeho koncový terminál. Obr. 4-8 ukazuje možnosti propojení, hlasové služby jsou znázorněny oranžovou barvou.



Obr. 4-8 Integrace systému – analogový uživatel

4.5 Příklad integrace systému II – VoIP uživatel

Rozšířením STU o NUA, vznikl systém, kde lze připojit VoIP koncové terminály. Ty rozšíří celý systém o plnou možnost volání do systému STU. Navíc lze koncové terminály pomocí NUA ovládat. Pomocí modifikace SIP hlavičky lze například automaticky přijmout (vyzvednout) hovor na koncovém terminálu. Tuto službu musí koncový VoIP terminál (VoIP telefon) podporovat, a musí mít ve své konfiguraci tuto službu povolenou.



Obr. 4-9 Integrace systému – VoIP uživatel

Toho využívá web rozhraní, kdy hovor sestavovaný tímto způsobem je při směřování na VoIP terminál automaticky přijat koncovým terminálem volajícího. Ještě před vyzvednutím telefonu se na displeji telefonu objeví ID volaného ve formě telefonního čísla a telefonního jména.

```
exten => _XXX,1,SIPAddHeader(Call-Info: answer-after=1)
```

Tab. 4-21 - Modifikace SIP hlavičky v dialplanu

Modifikace hlavičky v dialplanu Asterisku provádí aplikace *SIPAddHeader*. V tomto případě se nastavuje parametr *Call-Info* na hodnotu *answer-after=1*. Znamená to přijetí hovoru po čase 1s. Kdyby byla hodnota *answer-after=0*, přijal by telefon hovor okamžitě. Volba 1s je vhodná z důvodu zvukového upozornění před vyzvednutím telefonu (zvonění po dobu 1s). Koncový terminál musí mít zapnutou funkci *intercom*.

```
INVITE sip:333@192.168.55.22:5062 SIP/2.0
Via: SIP/2.0/UDP 132.174.103.21:5060;branch=z9hG4bK64411704;rport
Max-Forwards: 70
From: "O2 telefonica" <sip:800123456@132.174.103.21>;tag=as130cc903
To: <sip:333@192.168.55.22:5062>
Contact: <sip:800123456@132.174.103.21:5060>
Call-ID: 07e167c974672cc16d4cc20d0a0dd7b9@132.174.103.21:5060
CSeq: 102 INVITE
User-Agent: Asterisk PBX 1.8.3.3
Date: Sun, 22 Apr 2012 21:19:01 GMT
Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, SUBSCRIBE, NOTIFY, INFO, PUBLISH
Supported: replaces, timer
Call-Info: answer-after=1
Content-Type: application/sdp
Content-Length: 285

v=0
o=root 1276932514 1276932514 IN IP4 132.174.103.21
s=Asterisk PBX 1.8.3.3
c=IN IP4 132.174.103.21
t=0 0
m=audio 14888 RTP/AVP 8 3 0 101
a=rtpmap:8 PCMA/8000
a=rtpmap:3 GSM/8000
a=rtpmap:0 PCMU/8000
```

Tab. 4-22 - Sekvenční diagram protokolu SIP s parametrem Call-Info

Po přijetí hovoru telefonem, a zobrazením ID požadovaného volaného, začne sestavování spojení při zapnutém hlasitém příposlechu telefonního přístroje. Ostatní chování je stejné jako v příkladu analogového řešení.

ZÁVĚR

Cílem práce bylo dosáhnout, pomocí implementace softwarové ústředny Asterisk, integrace nových telefonních služeb do stávající infrastruktury ústředěn TDM. Práce se hlavně zaměřuje na rozšíření služeb o IP telefonii VoIP a služby s tím související, z důvodu ekonomické výhodnosti. Podobné služby lze najít i u ústředěn jiných výrobců, ale buď se jedná o služby uzavřené (nelze je upravit dle požadavků zákazníka), nebo jejich cena je neúměrně vysoká.

Všechny body této práce byly splněny. Teoretická část seznamuje se všemi prvky, sloužícími k propojení ústředěn. Popisuje také rozsáhlé možnosti softwarové ústředny Asterisk. Popisuje strukturu a principy komunikačních linek a signalizací, nutné pro vzájemné navázání spojení mezi účastníky. Není primárně zaměřena na popis principů přenosu medií (hlasových streamů), které samozřejmě k VoIP a TDM patří. Praktická část na příkladu ukazuje, jakým způsobem lze tyto poznatky využít. Jedna z mnoha aplikací, které lze použít, je webový seznam a webový telefon. Je naprogramován v jazyce PHP. Tato aplikace samozřejmě neposkytuje všechny možné volby. Například by měla být doplněna o funkce, jako jsou zavěšení telefonu z webového rozhraní, přihlášení stanice by mělo být zabezpečeno pinem, atd. Také by měl být optimalizován zdrojový programový kód, což z časových důvodů nebylo zatím provedeno.

Práce dokázala rozsáhlé možnosti open source řešení a VoIP. Díky tomu neexistují limity, co by požadovaná telefonní aplikace měla umět. Každopádně je důležité mít na mysli, že díky otevřenosti těchto systémů a především VoIP, které jako přenosové medium používá internet, je potřeba klást veliký důraz na bezpečnost těchto systémů.

ZÁVĚR V ANGLIČTINĚ

The objective was to achieve integration of new telephone services to existing TDM PBX infrastructure through the implementation of the Asterisk PBX software. The work is mainly focused on expansion of services for IP telephony and related VoIP services, and their economic practicality. Similar services can be found in PBXs from other manufacturers, but either it is a rigid contract (cannot be adjusted according to customer requirements), or their cost is too high.

All points of this work have been met. The theoretical part introduces all the elements serving to link PBX. It also describes the extensive opportunities offered by PBX software Asterisk. It describes the structure and principles of communication lines and signaling necessary for the mutual connection between the participants. Its primary focus is not on the description of the principles of transmission media (voice stream), which, of course, are part of VoIP and TDM. The practical part shows on an example how this knowledge can be utilized. One of the many applications that can be used is the web list and web phone. It is written in PHP. Of course, this application does not provide all possible options. For example, it should be accompanied by features such as hanging up the phone from web interface; login station should be secured by a pin, etc. It should also have optimized source code, which due to lack of time has not yet been done.

The work proved extensive open source solutions and VoIP. As a result there are no limits to requirements of the telephone applications. In any case, it is important to keep in mind that thanks to the openness of these systems and VoIP in particular, which as a transmission medium uses the Internet, we need to put great emphasis on the safety of these systems.

SEZNAM POUŽITÉ LITERATURY

- [1] SIMIONOVICH, Nir. Asterisk gateway interface 1.4 and 1.6 programming: design and develop Asterisk-based VoIP telephony platforms and services using PHP and PHPAGI. 1st ed. Birmingham: Packt Publishing, 2009, 200 s. ISBN 978-184-7194-466.
- [2] VOZŇÁK, Miroslav. Voice over IP. 1. vyd. Ostrava: VŠB - Technická univerzita Ostrava, 2008, 176 s. ISBN 978-802-4818-283.
- [3] VOZŇÁK, Miroslav. Spojovací systémy. 1. vyd. Ostrava: Vysoká škola báňská - Technická univerzita Ostrava, 2009, 196 s. ISBN 978-802-4819-617.
- [4] LEIF MADSEN, Jim Van Meggelen. *Asterisk: the definitive guide*. 3rd ed. Sebastopol, CA: O'Reilly Media, Inc, 2011. ISBN 978-059-6517-342.
- [5] WINTERMEYER, Stefan a Stephen BOSCH. Practical Asterisk 1.4 and 1.6. Upper Saddle River: Addison-Wesley, 2009, 793 s. ISBN 978-032-1525-666.
- [6] BRYANT, Russell. DIGIUM, Inc. *Asterisk Project* [online]. Huntsville, AL, 2010 [cit. 2012-04-22]. Dostupné z: <https://wiki.asterisk.org/wiki/display/AST/Home>
- [7] Telefonní ústředna. *Wikipedie* [online]. 2006, 10. 2. 2012 [cit. 2012-04-22]. Dostupné z: http://cs.wikipedia.org/wiki/Telefonn%C3%AD_%C3%BAst%C5%99edna
- [8] Public switched telephone network. *Wikipedia* [online]. 2001, 10 April 2012 [cit. 2012-04-22]. Dostupné z: http://en.wikipedia.org/wiki/Public_switched_telephone_network
- [9] FIŠER, Daniel. ISDN. *Wikipedie* [online]. 18. 7. 2005, 26. 2. 2012 [cit. 2012-04-01]. Dostupné z: <http://cs.wikipedia.org/wiki/ISDN>
- [10] BRI a PRI přípojky ISDN. ATLANTIS TELECOM S.R.O. *Matra Nortel Communications* [online]. © 1997-1999 [cit. 2012-04-22]. Dostupné z: http://www.matra.cz/isdn_pri.htm
- [11] Služby ISDN. ATLANTIS TELECOM S.R.O. *Matra Nortel Communications* [online]. © 1997-1999 [cit. 2012-04-22]. Dostupné z: http://www.matra.cz/isdn_srv.htm
- [12] LUDÍK, Martin. *Domáci VoIP ústředna s připojením do GSM sítí*. Zlín, 2009-06-01. Dostupné z: <http://dspace.k.utb.cz/handle/10563/8854>. Bakalářská práce.

Univerzita Tomáše Bati ve Zlíně, Ústav aplikované informatiky, Informační technologie. Vedoucí práce Ing. Tomáš Dulík.

- [13] RFC3261: Session Initiation Protocol. *IETF Tools* [online]. June 2002 [cit. 2012-04-22]. Dostupné z: <http://tools.ietf.org/html/rfc3261>
- [14] About The Asterisk Project. DIGIUM, Inc. *Asterisk* [online]. © 2012 [cit. 2012-04-22]. Dostupné z: <http://www.asterisk.org/asterisk>
- [15] HyperText Markup Language. MOJŽÍŠEK, Zdeněk. *Wikipedie* [online]. 16.7.2004, 1.2.2012 [cit. 2012-04-22]. Dostupné z: http://cs.wikipedia.org/wiki/HyperText_Markup_Language
- [16] GILMORE, W. *Velká kniha PHP 5 a MySQL: kompendium znalostí pro začátečníky i profesionály*. Nové, 3. vyd. Překlad Jan Pokorný. Brno: Zoner Press, 2011, 736 s. Encyklopedie Zoner Press. ISBN 978-80-7413-163-9.
- [17] PHP. *Wikipedie* [online]. 2.6.2004, 20.4.2012 [cit. 2012-04-22]. Dostupné z: <http://cs.wikipedia.org/wiki/PHP>

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

PSTN	Public Switched Telephone Network
PBX	Private Branch Exchange
TCP/IP	Transmission Control Protocol/Internet Protocol
SIP	Session Initiation Protocol
POTS	Plain Old Telephone Service,
BRI	Basic Rate Interface
PRI	Primary Rate Interface
VoIP	Voice over Internet Protocol
HTTP	Hypertext Transfer Protocol
MGCP	Media Gateway Control Protocol
H323	VoIP protokol H323
ISDN	Integrated Services Digital Network
DSS1	Digital Subscriber Signalling System No. 1
DDI	Direct Dial-In (Direct Inward Dialing (DID))
MSN	Multiple Subscriber Numbering
QSIG	Signalizační protokol Q
OSI	Open Systems Interconnection
PTP	Point to Point (ISDN)
TEI	Terminal Endpoint Identifier
CLIR	Calling Line Identification Restriction
CLIP	Calling line ID Presentation
TP	Terminal portability
VoIP	Voice over IP Protocol
SDP	Session Description Protocol

RTP	Real-time Transport Protocol
ENUM	E164 NUmber Mapping
ACD	Automatic Call Distribution
IVR	Interactive voice response
CLI	Command-Line Interface
AGI	Asterisk Gateway Interface
AMI	Asterisk Manager Interface
NUA	Nová Telefonní Ústředna Asterisk
STU	Stávající telefonní ústředna
AstDB	Asterisk Database
Apache	Softwarový Webový Server
PHP	Skriptovací Programovací Jazyk
ODBC	Open Database Connectivity
GUI	Graphical User Interface
TLS	Transport Layer Security
LCR	Least Cost Routing
TDM	Time-division multiplexing

SEZNAM OBRÁZKŮ

Obr. 2-1 Model OSI - ISDN	15
Obr. 2-2 Průběh výměny zpráv při sestavení spojení	16
Obr. 2-3 Příklad zprávy SETUP u DSS1	18
Obr. 2-4 Typické rozložení informací prvního řádku žádosti / odpovědi	23
Obr. 2-5 Výměna zpráv v dialogu	24
Obr. 2-6 Transakce a dialog SIP protokolu	25
Obr. 2-7 Tok zpráv při síťování mezi SIP, Q.931, Loop-Start	26
Obr. 3-1 Asterisk jako ústředna	28
Obr. 3-2 Asterisk jako brána.....	28
Obr. 3-3 Asterisk voicemail.....	28
Obr. 3-4 Porovnání architektury standartní PBX a Asterisku	29
Obr. 3-5 HW karta Sangoma	30
Obr. 3-6 HW karta Digium	30
Obr. 3-7 Vztah kontextu a dialplanu.....	32
Obr. 3-8 Správce událostí AMI.....	38
Obr. 3-9 Správce akcí AMI.....	38
Obr. 3-10 Jak pracuje AGI.....	39
Obr. 4-1 Zapojení NUA do nového telefonního systému	44
Obr. 4-2 Volba čísla 581123455 nebo 55	45
Obr. 4-3 Úplná integrace STU a webové služby NUA.....	46
Obr. 4-4 Integrovaný telefonní seznam	47
Obr. 4-5 Příklad jednoho řádku telefonního seznamu HTML.....	52
Obr. 4-6 Webový telefon	56
Obr. 4-7 Přiřazení web telefonu ke koncovému terminálu uživatele	56
Obr. 4-8 Integrace systému – analogový uživatel.....	61
Obr. 4-9 Integrace systému – VoIP uživatel.....	61

SEZNAM TABULEK

Tab. 2-1 - Porovnání linek E1, T1, J1	15
Tab. 2-2 - SIP hlavička (červená) a tělo (zelená) zprávy.....	22
Tab. 3-1 - Ukázka větvení dialplanu pomocí lokálního kanálu	31
Tab. 3-2 - Zápis extension	32
Tab. 3-3 - Tři položky <i>exten</i> , odděleny čárkami.....	32
Tab. 3-4 - Příklad zápisu extension	32
Tab. 3-5 - Ukázka odpovědi a zavěšení.....	33
Tab. 3-6 - Ukázka dialplanu jednoho kontextu a konkrétní extensions.....	33
Tab. 3-7 - Znaků vzorů se speciálním významem	34
Tab. 3-8 - Syntaxe řetězců vracejících hodnotu danou parametrem za dvojtečkou	34
Tab. 3-9 - Ukázka práce se vzory a řetězci	34
Tab. 3-10 - Správná syntaxe makra	34
Tab. 3-11 - Ukázka práce s makrem	35
Tab. 3-12 - Uložení dat do databáze	35
Tab. 3-13 - Načtení dat z databáze	36
Tab. 3-14 - Odstranění dat z databáze pomocí funkce	36
Tab. 3-15 - Odstranění dat z databáze pomocí aplikace	36
Tab. 3-16 - Ukázka komunikace AMI	37
Tab. 3-17 - Volání jednoduchého AGI skriptu "Ahoj světe" z dialplanu.....	39
Tab. 3-18 - Syntaxe AGI()	39
Tab. 3-19 - FastAGI()	40
Tab. 3-20 - Async AGI—AMI-Controlled AGI.....	40
Tab. 4-1 - Úprava a použití čísla v národním formátu.....	45
Tab. 4-2 - Závislosti klíčů databáze.....	48
Tab. 4-3 - Programový kód – komunikace ústřednu pomocí soketu	48
Tab. 4-4 Ukázka konfiguračního souboru <i>manager.conf</i>	49
Tab. 4-5 - Obsah proměnné <i>\$dbentries1</i>	49
Tab. 4-6 - Obsah pole proměnné <i>\$array1[18]</i>	49
Tab. 4-7 - Skript pro získání požadovaných hodnot.....	50
Tab. 4-8 - Závislost proměnné a hodnoty	51
Tab. 4-9 - Tisk hodnot do HTML tabulky	51
Tab. 4-10 - HTML formulář pro uložení dat telefonního seznamu	52

Tab. 4-11 - Skript <i>speed_status.php</i> pro uložení hodnoty do databáze	53
Tab. 4-12 - Kód tlačítka <i>edit</i>	54
Tab. 4-13 - Kód tlačítka <i>delete</i>	55
Tab. 4-14 - Skript <i>speed_delete.php</i> pro smazání hodnoty z databáze	55
Tab. 4-15 - Uložení cookies s ID uživatele	57
Tab. 4-16 - Akce Asterisku, sestavení spojení	57
Tab. 4-17 - Volby akce <i>Originate</i>	57
Tab. 4-18 - Proměnné v akci <i>Originate</i>	58
Tab. 4-19 - Dialplan Asterisku pro volání z web rozhraní	58
Tab. 4-20 - Soubor <i>evoma_conf.inc</i>	60
Tab. 4-21 - Modifikace SIP hlavičky v dialplanu	62
Tab. 4-22 - Sekvenční diagram protokolu SIP s parametrem Call-Info	62

SEZNAM PŘÍLOH

PŘÍLOHA P I: SOUBOR EXTENSION.CONF	73
PŘÍLOHA P II: SOUBOR MANAGER.CONF	75
PŘÍLOHA P III: SOUBOR SPEED.PHP	75
PŘÍLOHA P IV: SOUBOR SPEED_STATUS.PHP (jen PHP kód)	80
PŘÍLOHA P V: SOUBOR SPEED_STATUS.PHP (jen PHP kód)	81
PŘÍLOHA P VI: SOUBOR PHONE_DIAL.PHP (jen PHP kód)	82

PŘÍLOHA P I: SOUBOR EXTENSION.CONF

```
[general]
;=====
static = yes
writeprotect = yes
autofallthrough = no

[globals]
;=====
DYNAMIC_FEATURES=>automon

[unauthenticated]
exten => _X.,1,Hangup()

[services]
;=====
;Echotest
exten => 399,1,Answer()
exten => 399,n,NoOp(Echotest)
exten => 399,n,Playback(demo-echotest)
exten => 399,n,Echo()
exten => 399,n,Hangup()

exten => 398,1,Answer
exten => 398,n,Playtones(info)
exten => 398,n,Wait(10)

exten => _8XXX,1,NoOp(DB/speeddial: ${DB(speeddial/${EXTEN})})
exten => _8XXX,n,Goto(level_1,0${DB(speeddial/${EXTEN})},1)
exten => _8XXX,n,Hangup()

[halali]
;=====
exten => s,1,Answer()
exten => s,n,Playtones(info) ;halalí
exten => s,n,Wait(2)
exten => s,n,Playback(invalid) ;co není definováno zahodit
exten => s,n,Playtones(info) ;halalí
exten => s,n,Wait(15)
exten => s,n,Hangup()

[level_1]
;=====
include => level_0
exten => _+420X.,1,Goto(7${EXTEN:4},1)
exten => _420X.,1,Goto(7${EXTEN:3},1)
exten => _0XXXXXXXX,1,Macro(national,${EXTEN:1}) ;narodní číslo s nulou
exten => _XXXXXXXX,1,Macro(national,${EXTEN}) ;narodní číslo bez nuly

[level_0]
;=====
include => services
include => halali

exten => _XXX, hint, SIP/${EXTEN}
```

```

exten => _[0123]X,1,Macro(voipUA,${EXTEN}) ;UA 00 - 39
exten => _[4-9]X,1,Macro(isdnOXO,${EXTEN}) ;ISDN 40 - 99
exten => _39X,1,Goto(services,${EXTEN},1) ;Sluzby
exten => _X.,1,Goto(halali,s,1)

```

```
[local_ami]
```

```
;=====
include => halali
```

```

exten => _XXX,1,SIPAddHeader(Call-Info: answer-after=1)
exten => _XXX,n,Goto(level_0,${EXTEN},1)

```

```
[CallerDB]
```

```
;=====
exten => s,1,GotoIf("${var1}" = "")?n8
;exten => s,n,Set(CALLERID(num)=${var1})
exten => s,n(n8),NoOp(2, CALLERID number: ${CALLERID(num)})
exten => s,n,NoOp(2, CALLERID name: ${CALLERID(name)})
exten => s,n,NoOp(2,astDB number (speedname):
${DB(speedname/${CALLERID(num)})})
exten => s,n,NoOp(2,astDB number (cidname): ${DB(cidname/${CALLERID(num)})})
exten => s,n,NoOp(2,CHANNEL info: ${CHANNEL})
exten => s,n,GotoIf("${LEN(${CALLERID(num)})}!=3"?speedn);delka cisla volajici
exten => s,n(cidn),Set(CALLERID(name)=${DB(cidname/${CALLERID(num)})})
exten => s,n,Goto(n7)
exten => s,n(speedn),Set(CALLERID(name)=${DB(speedname/${CALLERID(num)})})
exten => s,n(n7),NoOp(2,new CALLERID name: ${CALLERID(name)})
exten => s,n,Return()

```

```
[macro-voipUA]
```

```
;=====
exten => s,1,NoOp(3,Dial user: SIP/${ARG1})
same => n,NoOp(3, CALLERID: ${CALLERID(all)})
same => n,Gosub(DevState,s,1)
same => n,Gosub(CallerDB,s,1)
same => n,Dial(SIP/${ARG1},,tT)
same => n,Hangup()

```

```
[macro-isdnOXO]
```

```
;=====
exten => s,1,NoOp(Volba ucastnika: DAHDI/g0/${ARG1})
same => n,Set(GLOBAL(PICKUPMARK)=${ARG1})
same => n,NoOp(${CALLERID(all)})
;same => n,Gosub(DevState,s,1)
same => n,Dial(DAHDI/g0/${ARG1})
same => n,Hangup()

```

```
[macro-national]
```

```
;=====
exten => s,1,NoOp(Calling ${ARG1})
same => n,Dial(DAHDI/g0/0${ARG1})
same => n,Hangup()

```

```
[macro-international]
```

```
;=====
exten => s,1,NoOp(Calling ${ARG1})
same => n,Dial(DAHDI/g0/0${ARG1},,L(900000))
same => n,Hangup()

```

```

[pbx-NT]
;=====
include => level_1

exten => s,1,Answer
exten => s,n,Playtones(stutter)
exten => s,n,Wait(10)

[DevState]
;=====
exten => s,1,NoOp( Dial Status: ${DEVICE_STATE(SIP/${ARG1})});
    same => n,Goto(s-${DEVICE_STATE(SIP/${ARG1})},1)

exten => s-NOT_INUSE,1,Return()
exten => s-BUSY,1,Playback(vm-isonphone)
exten => s-BUSY,n,Hangup()
exten => s-INUSE,1,Playback(vm-isonphone)
exten => s-INUSE,n,Hangup()
exten => s-UNKNOWN,1,Goto(halali,s,1)
exten => s-UNKNOWN,n,Hangup()
exten => s-UNAVAILABLE,1,Playback(vm-isunavail)
exten => s-UNAVAILABLE,n,Hangup()
exten => s-INVALID,1,Playback(invalid)
exten => s-INVALID,n,Hangup()
exten => s-RINGING,1,Busy
exten => s-RINGINGUSE,1,Busy
exten => s-ONHOLD,1,Busy

```

PŘÍLOHA P II: SOUBOR MANAGER.CONF

```

[general]
enabled = yes
port = 5038
bindaddr = 0.0.0.0

[admin]
secret = mypass
read = system,call,log,verbose,command,user,config,command,reporting,dialplan,
originate
write = system,call,log,verbose,command,user,config,command,reporting,dialplan,
originate

```

PŘÍLOHA P III: SOUBOR SPEED.PHP

```

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"><!-- InstanceBegin
template="/Templates/index.dwt" codeOutsideHTMLOIsLocked="false" -->
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<link rel="stylesheet" type="text/css" href="../style.css" />
<!-- InstanceBeginEditable name="doctitle" -->
<title>VoIPMachine</title>
<!-- InstanceEndEditable -->
<!-- InstanceParam name="OptionalRegion1" type="boolean" value="true" -->
<!-- InstanceBeginEditable name="head" -->

```

```

<!-- InstanceEndEditable -->
<style type="text/css">
body {
background-color: #1C478E;
}
</style>
</head>

<body>
<div id="container">
<div id="header">
<h1>VoIP<span class="off">Machine</span></h1>
<h2><span class="menuitem">
<?php
include "../evoma_conf.inc"; // vlozeni specifickych udaju ze souboru
$ExtenCookieValue = $_COOKIE["ExtenCookVar"]; // Nastaveni cookies uzivatele
if (!isset($ExtenCookieValue)) {
print "Nepřihlášen";
}
else{
print "Přihlášen jako: $ExtenCookieValue";
}
?>
</span></h2></div>
<div id="menu">
<ul>
<li class="menuitem"><a href="../index.php">Home</a></li>
<li class="menuitem"><a href="#">Intranet</a></li>
<li class="menuitem"><a href="#">Záznam hovorů</a> </li>
<li class="menuitem"><a href="#"></a> </li>
<li class="menuitem"><a href="#">Help</a></li>
<li class="menuitem"><a href="#"></a> </li>
<li class="menuitem"><a href="../login/login.php">Přihlášení</a></li>
<li class="menuitem"><a href="/test2/login/login_off.php">Odhlášení</a> </li>
</ul>
</div>
<div id="leftmenu">
<div id="leftmenu_top"></div>
<div id="leftmenu_main">
<h3>Základní služby</h3>
<ul>
<li><a href="speed.php">Telefonní seznam</a></li>
<li><a href="/test2/directory/directory.php">Interní uživatelé</a></li>
<li><a href="../phone/phone.php">Web telefon</a></li>
</ul>
<h3>Rozířené služby</h3>
<ul>
<li><a href="/test2/extens/extens.php">SIP Extensions</a></li>
<li><a href="#">Time ranges</a></li>
<li><a href="#">External tools</a></li>
<li><a href="#">Entertainment</a></li>
</ul>
<h3>Odkazy</h3>
<ul>
<li></li>
</ul>
</div>
<div id="leftmenu_bottom"></div>
</div>

```

```

<div id="content">
<div id="content_top"></div>
<div id="content_main"><!-- InstanceBeginEditable name="EditRegion3" -->
<p>&nbsp; </p>
<form action=" ../speed/speed_status.php" method="post" >
<table width="408" cellpadding="2">
<tr>
<td width="184"><strong>Číslo rychlovolby:</strong></td>
<td width="208"><input name="SpeedNumber"
value="<?php echo $_GET["speednumber"];?>" size="30" /></td>
</tr>
<tr>
<td><strong>Telefonní číslo:</strong></td>
<td><input name="PhoneNumber"
value="<?php echo $_GET["phonenumber"];?>" size="30" /></td>
</tr>
<tr>
<td><strong>Jméno stanice:</strong></td>
<td><input name="PhoneName"
value="<?php echo $_GET["phonenumber"];?>" size="30" /></td>
</tr>
<tr>
<td></td>
<td><div align="right"><strong>
<input type="submit" value="Uložit" />
</strong></div>
<div align="right"></div></td>
</tr>
</table>
</form>
<p>
<?php
import_request_variables("GP");

$BackLink = $_SERVER['HTTP_REFERER'];

print "<table width=\"100%\">\n";
print "<tr>\n";
print "<th><b>Rychlovolba</b></th>";
print "<th><b>Číslo</b></th>";
print "<th><b>Jméno</b></th>";
print "<th><b>Volat</b></th>";
print "<th><b>Editace</b></th>";
print "<th><b>Smazat</b></th>";
print "</tr>\n";

//-----Nacteni databaze speeddial-----

$socket = fsockopen("127.0.0.1","5038", $errno, $errstr, $timeout);
//Open AMI socket

fputs($socket, "Action: Login\r\n"); // Akce prihlaseni
fputs($socket, "UserName: $man_name\r\n"); // Vlozeni uzivatelskeho jmena
fputs($socket, "Secret: $man_pword\r\n\r\n"); // Vlozeni uzivatelskeho hesla
fputs($socket, "Action: Command\r\n"); // Akce zadavani prikazu do
Asterisku
fputs($socket, "Command: database show speeddial\r\n\r\n");
// Specificky dotaz na databazi
fputs($socket, "Action: Logoff\r\n\r\n"); // Odhlaseni

```

```

while (!feof($socket))
{
$dbentries1 .= fread($socket, 8192);
    // Cteni vstupu ze soketu a ulozeni do promenne $dbentries1
}

$array1 = preg_split("/\n/", $dbentries1, -1, PREG_SPLIT_NO_EMPTY);
    // rozdeleni promenne $dbentries1 do pole řetězců $array1 dle parametru

for ($i = 1; $i < 7; $i++)
{
array_shift($array1); // Ostraneni prvnych "i" radku nemajici potrebnou
informaci
}

fclose($socket); // Zavreni soketu socket

//-----Nacteni databaze speedname-----

$socket = fsockopen("127.0.0.1", "5038", $errno, $errstr, $timeout);

fputs($socket, "Action: Login\r\n");
fputs($socket, "UserName: $man_name\r\n");
fputs($socket, "Secret: $man_pword\r\n\r\n");

fputs($socket, "Action: Command\r\n");
fputs($socket, "Command: database show speedname\r\n\r\n");
fputs($socket, "Action: Logoff\r\n\r\n");

while (!feof($socket))
{
$dbentries2 .= fread($socket, 8192);
}

$array2 = preg_split("/\n/", $dbentries2, -1, PREG_SPLIT_NO_EMPTY);

for ($i = 1; $i < 7; $i++)
{
array_shift($array2);
}

fclose($socket);

//-----Ziskani zaznamu cisla a vypsani do html tabulky-----

for ($a=$start_a;$a<=$stop_a;$a++)
    // Citac 1000 hodnot (pvni 4 radky odstraneny dle verze asterisku)
{
$speed_no_line=$array1[$a];
    // Nacteni jednotlivich radku z pole od cisla 0 do 999

if (ereg("results.*",$speed_no_line, $status_dial))
{
    //Zastaveni zpracovani pri vyskytu "result" v prohledavanych radcich
$a=$stop_a; // Nastaveni koncove hodnoty pri ukonceni prohledavani
print "Data zpracována.";
    // Ukonceni pri vyskytu "result" v prohledavanych radcich
}
}

```

```

else
{
$cut_line = explode ("/", $speed_no_line);
// Rozdeleni radku na jednotlivé hodnoty přes "/", vystup do pole
ereg("[0-9][0-9][0-9][0-9]", $cut_line[2], $out_no);
// Vyber druhého pole z rozdeleneho radku, jen ctyrmistne cislo
ereg("[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]", $cut_line[2], $out_dial_no);

//-----Ziskani jmena z databaze pomoci cisla telefonu-----

for ($x=0;$x<999;$x++) // Citac pro hledani vyberu jmena k tel číslu
{
$speed_name_line=$array2[$x]; // Nacteni hodnoty z radku pole $array2
$cut_name_line = explode ("/", $speed_name_line);
// Rozdeleni radku na jednotlivé hodnoty přes "/", vystup do pole
$text_name = $cut_name_line[2];
// Ulozeni promene (j) radku z pole dve promenne "$cut_name_line"
$result_name = strstr($text_name, $out_dial_no[0]);
// Hledani text stringu hodnoty "out_dial..." v "text_name",
// prirazeni jmena k cislu
if ($result_name>""){ // Jestlize li je vysledek vetsi jak 0
$result_name_array[$a]=$result_name;
// Uloz vysledek do pole s hodnotou "speeddial cisla"
}
}
ereg(":[A-z].*", $result_name_array[$a], $out_name_1);
// Uprava nalezene hodnoty pred zobrazenim
ereg("[A-z].*", $out_name_1[0], $out_name);

//-----Tisk tabulky -----

if ($a/2 != floor($a/2)) { // Test lichého čísla na obarvení radku
$color="#E6E6E6"; // liche číslo
}
else {
$color="#A5B9FE"; // sude číslo
}
print "<tr>\n";
print "<td bgcolor=\"\$color\"><div align=\"center\"><b>$out_no[0]</b></div></td>";
print "<td bgcolor=\"\$color\"><b>$out_dial_no[0]</b></td>";
print "<td bgcolor=\"\$color\"><b>$out_name[0]</b></td>";

print "<td width=\"60\" bgcolor=\"\$color\"><b><div align=\"center\">
<a href=\"../phone/phone_dial.php?phonenumber=$out_dial_no[0]\">
<img src=\"../img/contact16.png\" alt=\"edit\" title=\"Volat\" />
</a></b></div></td>";

print "<td width=\"60\" bgcolor=\"\$color\"><div align=\"center\">
<a href=\"speed.php?speednumber=$out_no[0]&
phonenumber=$out_dial_no[0]&phonename=$out_name[0]\">
<img src=\"../img/edit16.png\" alt=\"edit\" title=\"Editovat\" />
</a></div></td>";

print "<td width=\"60\" bgcolor=\"\$color\"><div align=\"center\">
<a href=\"speed_delete.php?speednumber=$out_no[0]&
phonenumber=$out_dial_no[0]&phonename=$out_name[0]\"
onclick=\"return confirm('Opravdu smazat záznam $out_no[0]?')\">

```

```

<img src=\"../img/delete16.png\" alt=\"Smazat\" title=\"Smazat\" />
</a></div></td>";

print "</tr>\n";
} //End else
} //End for
print "</table>";
print "<br/>";
?>
</p>
<!-- InstanceEndEditable --></div>
<div id="content_bottom"></div>
<div id="footer">
<h3><a href="http://www.bryantsmith.com">florida web desig</a></h3></div>
</div>
</div>
<div style="text-align: center; font-size: 0.75em;">
<p>Design by Martin Ludík</p>
<p>Version 3.8.4</p>
</div></body>
<!-- InstanceEnd --></html>

```

PŘÍLOHA P IV: SOUBOR SPEED_STATUS.PHP (jen PHP kód)

<?php

```

set_time_limit(5);
import_request_variables("GP");

$BackLink = $_SERVER['HTTP_REFERER'];

$PhoneNumber = $_POST['PhoneNumber']; // Ziskani dat z formulare metodou POST
$PhoneName = $_POST['PhoneName'];
$SpeedNumber = $_POST['SpeedNumber'];

//-----Ulozeni databaze speeddial-----

if ($PhoneNumber <> "" && $PhoneName <> "" && $SpeedNumber <> "")// Test
{
$socket = fsockopen("127.0.0.1","5038", $errno, $errstr, $timeout);

fputs($socket, "Action: Login\r\n");
fputs($socket, "UserName: $man_name\r\n");
fputs($socket, "Secret: $man_pword\r\n\r\n");

fputs($socket, "Action: Command\r\n"); // Akce zadavani prikazu do Asterisku
fputs($socket, "Command: database put speeddial $SpeedNumber $PhoneNumber
\r\n\r\n");
fputs($socket, "Action: Command\r\n");
fputs($socket, "Command: database put speedname $PhoneNumber \"$PhoneName\"
\r\n\r\n");
fputs($socket, "Action: Logoff\r\n\r\n");

while (!feof($socket))
{
$response .= fread($socket, 8192); // Cteni vstupu ze soketu a ulozeni do
promenne $response
}

```

```

fclose($socket);    // Zavreni soketu

if (strpos($response, 'Updated database successfully'))
    // Kontrola uspesne odpovedi od Asterisku
{
print("<p><strong>Vaše data byla úspěšně uložena do
databáze.</strong></p>\r\n");
print("<p><h2><a href=\"\$BackLink\">Zpět</a></h2</p>");
}
else print(" Error....");
}
else          // Nejsou li zadana spravna vstupni data
{
print "<p><strong>Někde je chyba!</strong></p>\n";
print "<p><strong>Prosíme zkontrolujte správné vstupní údaje!</strong></p>";
print("<p><h2><a href=\"\$BackLink\">Zpět</a></h2</p>");
}
}
?>

```

PŘÍLOHA P V: SOUBOR SPEED_STATUS.PHP (jen PHP kód)

<?php

```

set_time_limit(5);
import_request_variables("GP");

$BackLink = $_SERVER['HTTP_REFERER'];

$PhoneNumber = $_GET['phonenumber'];
$PhoneName = $_GET['phonename'];
$SpeedNumber = $_GET['speednumber'];

if ($PhoneNumber <> "" && $PhoneName <> "" && $SpeedNumber <> "")
{
$socket = fsockopen("127.0.0.1", "5038", $errno, $errstr, $timeout);    //
Login
fputs($socket, "Action: Login\r\n");
fputs($socket, "UserName: $man_name\r\n");
fputs($socket, "Secret: $man_pword\r\n\r\n");

fputs($socket, "Action: Command\r\n");
fputs($socket, "Command: database del speeddial $SpeedNumber \r\n\r\n");
// Mazani cisla
fputs($socket, "Action: Command\r\n");
fputs($socket, "Command: database del speedname $PhoneNumber \r\n\r\n");
// Mazani jmena
fputs($socket, "Action: Logoff\r\n\r\n");

while (!feof($socket))
{
$response .= fread($socket, 8192);
}

fclose($socket);

if (strpos($response, 'Database entry removed'))

```

```

{
print("<p><strong>Data byla úspěšně vymazána.</strong></p>\r\n");
print("<p><h2><a href=\"\$BackLink\">Zpět</a></h2</p>");
}
else print("Error....");
}
else
{
print "<p><strong>Někde je chyba!</strong></p>";
print("<p><h2><a href=\"\$BackLink\">Zpět</a></h2</p>");
}
?>

```

PŘÍLOHA P VI: SOUBOR PHONE_DIAL.PHP (jen PHP kód)

<?php

```

set_time_limit(5);
import_request_variables("GP");

$BackLink = $_SERVER['HTTP_REFERER'];

$PhoneNumber = $_GET['onenumber'];
$ExtenCookie = $_COOKIE["ExtenCookVar"];

if ($ExtenCookie == "") {
print "<strong><p>Nejste přihlášení.</p><p>Přihlašte se a zkuste to znovu!</p></strong>";
}
else {
print "<p><strong>Spojuji hovor na číslo: $PhoneNumber.</strong></p>";

$socket = fsockopen("127.0.0.1","5038", $errno, $errstr, $timeout);
fputs($socket, "Action: Login\r\n");
fputs($socket, "UserName: $man_name\r\n");
fputs($socket, "Secret: $man_pword\r\n\r\n");
fputs($socket, "Action: Originate\r\n"); // Akce Asterisku - originate
fputs($socket, "Channel: local/$ExtenCookie@$LocContext\r\n"); // Volající
fputs($socket, "Context: $VarContext\r\n"); // COntext (opravení)
fputs($socket, "Exten: $PhoneNumber\r\n"); // Volané číslo
fputs($socket, "Priority: 1\r\n"); // Priorita v dial planu
fputs($socket, "Timeout: 30000\r\n");
fputs($socket, "Variable: var1=$PhoneNumber\r\n");
// Pomocná proměnná pro správnou identifikaci
fputs($socket, "Callerid: $PhoneNumber\r\n\r\n"); // Identifikace volaného
fputs($socket, "Action: Logoff\r\n\r\n");

while (!feof($socket)) {
$response .= fread($socket, 8192);
}

fclose($socket);
}
print("<p><h2><a href=\"\$BackLink\">Zpět</a></h2</p>");
?>

```