České vysoké učení technické v Praze
Fakulta elektrotechnická
Katedra počítačů

Návrh a implementace protokolu SIPREC do dispečerského řešení „KONOS“

Bakalářská práce

Vypracovala: Sofia Shchepetova
Vedoucí: RNDr. Ladislav Serédi

Praha, 2021

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Shchepetova** Jméno: **Sofia** Osobní číslo: **483759**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra počítačů**

Studijní program: **Otevřená informatika**

Specializace: **Software**

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

Návrh a implementace podpory protokolu SIPREC v prostředí dispečerského řešení KONOS

Název bakalářské práce anglicky:

Design and implementation of SIPREC protocol support in KONOS call center system

Pokyny pro vypracování:

Studujte stávající způsoby nahrávání VoIP hlasové komunikace. Diskutujte doposud používané řešení VoIP nahrávání ReDat od společnosti „Retia“, s proprietárním rozhraním založeným na protokolu H.323. Do dispečerského řešení KONOS společnosti „TTC Marconi“ implementujte podporu SIPREC protokolu - současného standartu pro řízení nahrávání pomocí informací a meta-dat posílaných v rámci SIP zpráv - v jazyce Java. Porovnejte možnosti vámi navrženého a stávajícího řešení z hlediska integrace s dostupnými nahrávacími systémy. Implementovaný software otestujte - porovnejte, analyzujte a diskutujte různé způsoby nahrávání hlasové komunikace, jejich výhody a nevýhody.

Seznam doporučené literatury:

1. K. Rehor, L. Portman, A. Hutton, R. Jain. Use Cases and Requirements for SIP-Based Media Recording (SIPREC). 8.2011, RFC 6341.
2. K. Rehor, L. Portman, A. Hutton, R. Jain. An Architecture for Media Recording Using the Session Initiation Protocol. 5.2014, RFC 7245.
3. R. Ravindranath, P. Ravindran, P. Kyzivat. Session Initiation Protocol (SIP) Recording Metadata. 5.2016, RFC 7865.
4. L. Portman, H. Lum, C. Eckel, A. Johnston, A. Hutton. Session Recording Protocol. 5.2016, RFC 7866.
5. R. Ravindranath, P. Ravindran, P. Kyzivat. Session Initiation Protocol (SIP) Recording Call Flows. 2.2017, RFC 8068.

Jméno a pracoviště vedoucí(ho) bakalářské práce:

RNDr. Ladislav Serédi, kabinet výuky informatiky FEL

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) bakalářské práce:

Datum zadání bakalářské práce: **04.03.2021** Termín odevzdání bakalářské práce: **21.05.2021**

Platnost zadání bakalářské práce: **19.02.2023**

RNDr. Ladislav Serédi podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	prof. Mgr. Petr Páta, Ph.D. podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Studentka bere na vědomí, že je povinna vypracovat bakalářskou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací.
Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v bakalářské práci.

. _____
Datum převzetí zadání

_____
Podpis studentky

Čestné prohlášení

Prohlašuji, že jsem předloženou práci vypracovala samostatně a že jsem uvedla veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze dne

.....
Sofia Shchepetova

Poděkování

Chtěla bych poděkovat své rodině a přátelům za podporu během celého studia. Děkuji také svému vedoucímu RNDr. Ladislavu Serédi za odborné vedení, čas a pomoc, kterou mi v průběhu tvorby bakalářské práce věnoval. V neposlední řadě děkuji oponentovi Janu Neumannovi a kolegům z firmy „TTC Marconi“ za konstruktivní kritiku a cenné nápady na zlepšení práce.

Abstrakt

Bakalářská práce se zaměřuje na zkoumání různých způsobů nahrávání VoIP hlasové komunikace a na implementaci SIPREC protokolu do dispečerského řešení „KONOS“ společnosti „TTC Marconi“. Tento protokol je v současné době standardem pro řízení nahrávání pomocí informací a metadat posílaných v rámci SIP zpráv. V řešení je doposud používáno VoIP nahrávání „ReDat“ od společnosti „Retia“ s proprietárním rozhraním založeným na H.323 protokolu. Cílem teoretické části projektu je analýza protokolu SIPREC, jeho architektury, funkcionality a požadavků na implementaci. Tyto teoretické poznatky jsou dále uplatněny v části praktické, jejímž cílem je do terminálu „KONOS“ implementovat podporu SIPREC protokolu v jazyce Java. Toto umožní integraci s většinou dostupných nahrávacích systémů a zvýší potenciál celého řešení. V závěrečné části práce je navržená implementace otestována a porovnána s implementací stávající. Na základě srovnání jsou rozebrány a výhody a nevýhody jednotlivých řešení.

Klíčová slova: VoIP nahrávání, SIP, SIPREC, H.323, dispečerské řešení, metadata, IP protokol, IP telefonie

Abstract

The bachelor thesis focuses on the examination of different ways of recording VoIP communication and on the implementation of the SIPREC protocol in the dispatching solution „KONOS“ of the company „TTC Marconi“. This protocol is currently the standard for establishing and management recording session using information and metadata sent as part of SIP messages. So far, „ReDat“ server for VoIP recording from the company „Retia“ with a proprietary interface based on protocol H.323 is used in the solution. The goal of the theoretical part of the project is to analyze the SIPREC protocol, its architecture, functionality and implementation requirements. This theoretical knowledge is further applied in the practical part, which aims to implement support for the SIPREC protocol in Java. It will allow integration with most of the available recording systems and increase the potential of the entire solution. In the concluding part of the work, the proposed implementation is tested and compared with the original implementation. Based on the comparison, the advantages and disadvantages of each solution are discussed.

Keywords: VoIP recording, SIP, SIPREC, H.323, dispatching solution, metadata, IP protocol, IP telephony

Seznam obrázků

Obsah

Kapitola 1

Úvod

V dnešní době je dispečer klíčovou osobou, když se jedná o záchranu lidských životů a majetku. Operátoři dispečerských služeb musí rychle a efektivně reagovat na tísňová volání, aby byli schopni udělat pro vyřešení kritické situace maximum v co nejkratším čase. K tomu jim slouží dispečerské terminály.

Jedním z takových terminálů je dispečerský systém „KONOS“ firmy „TTC Marconi“. Umožňuje hlasovou komunikaci mezi různými koncovými uživateli, kteří mohou být připojení přes technologicky rozdílné pevné nebo mobilní sítě. Řízení jejich komunikace probíhá na bázi aplikací IP protokolů. Dispečerské řešení „KONOS“ usnadňuje a zjednodušuje komunikaci, nabízí jednotné rozhraní pro komunikaci do různých hlasových sítí. [15]

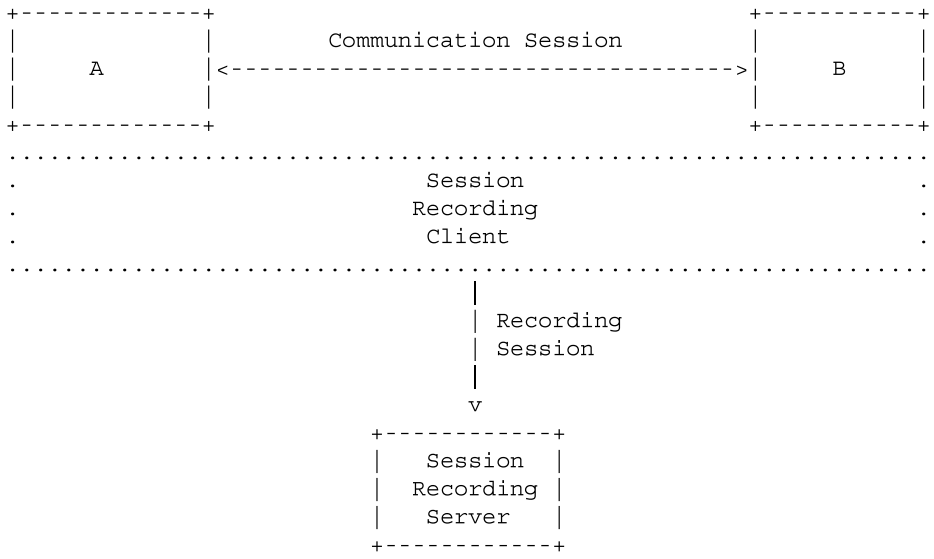
Jednou z důležitých funkcí dispečerských terminálů je nahrávání hovorů, a to především z bezpečnostních důvodů. Pro tento účel existuje několik protokolů, například H.323, který je aktuálně používán v dispečerském terminálu „KONOS“. V dnešní době se ale dává přednost standardizovaným protokolům. Jedním z nich je protokol SIPREC.

Cílem této bakalářské práce je navrhnout a implementovat klientskou část tohoto protokolu do dispečerského terminálu „KONOS“. Nahradí se tím komunikace založená na H.323 protokolu, který ale, kvůli zpětné kompatibilitě, nesmí být kompletně odstraněn z řešení. Práce je rozdělena na dva velké bloky - teoretický (kapitoly 3 až 6) a praktický (kapitoly 7 až 11). V teoretickém bloku jsou popsány protokoly SIP, SDP a SIPREC, případy a požadavky na jejich použití a potřebná architektura. Jsou také uvedeny kroky k úspěšnému nahrávání komunikace v závislosti na roli účastníka. V praktické části je navržená implementace protokolu SIPREC do dispečerského řešení „KONOS“ a jsou otestovány jednotlivé části této implementace.

Kapitola 2

Definice

- **SIP signalizace** - výměna SIP zpráv mezi účastníky relace. Zprávy protokolu SIP jsou dvojího druhu - žádosti, říká se jim též metody, a odpovědi.
- **Uživatelský agent (UA)** - internetový koncový bod.
- **Komunikační relace (Communication Session, CS)** - SIP relace vytvořená mezi dvěma nebo více uživatelskými agenty SIP, která je předmětem záznamu.
- **Server nahrávající relace (Session Recording Server, SRS)** - uživatelský agent SIP, záznamová jednotka nebo specializovaný mediální server, který působí jako přijímač zaznamenaných médií a metadat komunikační relace. SRS je obvykle implementován jako multiportové zařízení, které je schopné přijímat média z více zdrojů současně.
- **Klient nahrávající relace (Session Recording Client, SRC)** - uživatelský agent SIP, který se chová jako zdroj zaznamenaných médií, které posílá na SRS. Je to logická funkce, jejíž požadované schopnosti mohou být implementovány v jednom nebo více fyzických zařízeních. V praxi může být SRC například osobním zařízením (jako je SIP telefon) nebo dispečerským terminálem.
- **Nahrávací relace (Recording Session, RS)** - SIP relace vytvořená mezi SRC a SRS za účelem záznamu komunikační relace. Obrázek 2.1 představuje vztah mezi nahrávací relací a komunikační relací.



Obrázek 2.1: Vztah mezi RS a CS [2]

- **Recording-aware User Agent (RA UA)** - uživatelský agent SIP, který si je vědom rozšíření SIP spojených s komunikační relací. Taková rozšíření mohou být použita k informování UA, že hovor je nahráván, nebo k tomu, aby UA mohl uvést své preference ohledně spuštění, pozastavení, obnovení a ukončení nahrávání.
- **Recording-unaware User Agent (RU UA)** - uživatelský agent SIP, který si není vědom rozšíření SIP spojených s komunikační relací.
- **Metadata záznamu** - metadata popisující CS, která jsou vyžadována SRS. Zahrnují například ID uživatelů, kteří se účastní CS, a stav hovoru. Obvykle jsou tato metadata archivována spolu s replikovanými médií na SRS.
- **Replikovaná media** - kopie médií, která je spojena s CS, byla vytvořena SRC a poslána do SRS. Může obsahovat všechna média spojená s CS (např. audio a video) nebo pouze podmnožinu (např. jenom audio).

Kapitola 3

SIP: Session Initiation Protocol

3.1 Úvod

Existuje několik protokolů, které slouží pro přenos různých typů multimediálních dat relace jako hlas, video nebo textové zprávy v reálném čase. Session Initiation Protocol (**SIP**) spolupracuje s těmito protokoly tak, že umožňuje různým UA rozpoznat se navzájem a dohodnout se na charakteru relace, kterou by chtěli sdílet. Přesněji řečeno, SIP umožňuje vytvoření infrastruktury síťových hostitelů (proxy serverů), kterým mohou UA odesílat registrační zprávy, zprávy `INVITE` a další požadavky. SIP je univerzální nástroj pro vytváření, úpravu a ukončování relací, který funguje nezávisle na základních transportních protokolech a na typu relace, kterou chce uživatel vytvořit. Mediální streamy mohou být přidány do (resp. odstraněny z) existující relace. Jedny z nejdůležitějších takovýchto relací jsou internetové telefonní hovory.

3.2 Přehled funkcí SIP

Podle [1], SIP podporuje pět aspektů zakládání a ukončování multimediální komunikace:

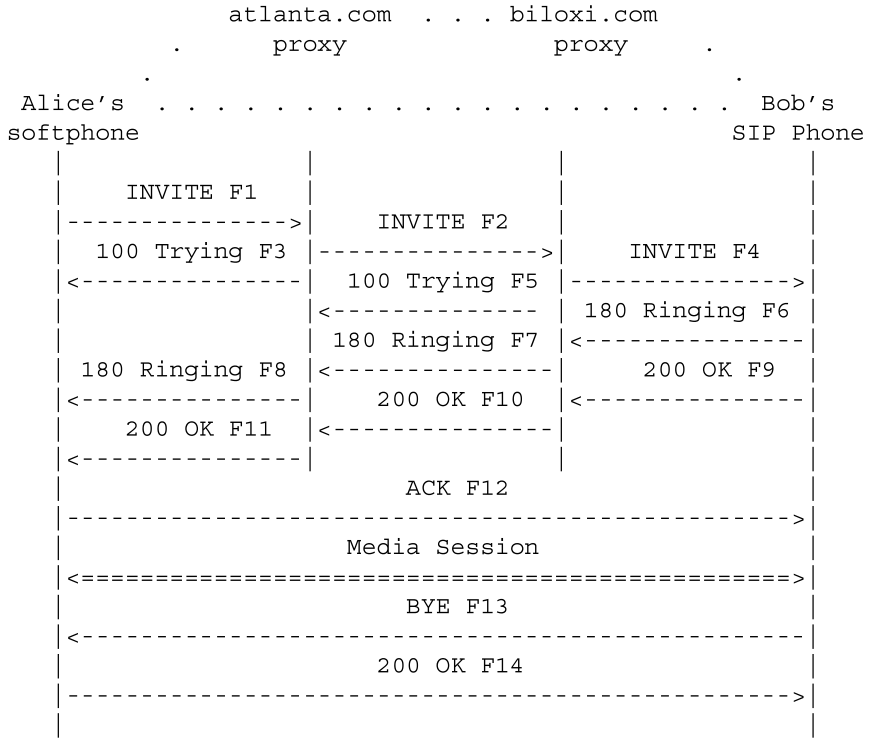
1. **Umístění uživatele**: určení koncového bodu, který má být použit pro komunikaci.
2. **Dostupnost uživatele**: určení ochoty volané strany zapojit se do komunikace.
3. **Schopnosti uživatele**: určení parametrů médií, které mají být použity.
4. **Založení relace**: „vyzvánění“ („ringing“), stanovení parametrů relace na volané i volající straně.
5. **Správa relace**: přenos a ukončení relace, úprava parametrů relace a vyvolání potřebných služeb.

SIP není integrovaný komunikační systém, je to spíše komponenta, kterou lze použít s jinými IETF protokoly (typicky Real-time Transport Protocol (RTP), Session Description Protocol (SDP) apod.) pro vytvoření kompletní multimediální architektury. SIP jako takový neposkytuje služby, ale primitivy, které lze použít k implementaci různých služeb. Například, SIP může najít uživatele a doručit multimediální objekt do jeho aktuálního umístění. Pokud se těchto primitiv používá například k doručení popisu relace napsané v SDP, koncové body se mohou shodnout na parametrech relace už jen na základě SIP protokolu. Pokud se stejných primitiv používá k doručení fotografie volajícího navíc, je možné snadno implementovat příslušnou službu. Jak ukazuje tento příklad, jedno primitivum se obvykle používá k poskytování několika různých služeb. Proto by měl být SIP používán spolu s jinými protokoly, aby uživatelům poskytoval kompletní služby. Základní funkce a operace SIP však nezávisí na žádném z těchto protokolů. [1]

3.3 Přehled vytvoření hovoru

Tato sekce představuje základní funkci SIP: umístění koncového bodu, vyjádření ochoty komunikovat, vyjednávání parametrů pro vytvoření relace a ukončení stanovené relace.

Obrázek 3.1 ukazuje typický příklad výměny SIP zpráv mezi dvěma uživateli, Alicí a Bobem. V tomto příkladu Alice používá na svém počítači SIP aplikaci označovanou jako softphone, aby zavolala Bobovi na jeho SIP telefon přes Internet. Jsou zobrazeny také dva proxy servery SIP, které jednají jménem Alice a Boba pro usnadnění založení relace. Toto typické uspořádání je často nazýváno „SIP lichoběžník“. Každá zpráva je označena písmenem „F“ a pořadovým číslem.



Obrázek 3.1: Typický SIP hovor [1]

Alice volá Bobovi pomocí jeho SIP identity (tzv. SIP URI). Ta se podobá e-mailové adrese a obsahuje uživatelské jméno a jméno hostitele. V tomto případě je to `sip:bob@biloxi.com`, kde `biloxi.com` je doménou poskytovatele SIP služeb Bobovi. Alice má URI `sip:alice@atlanta.com`.

SIP je založen na transakčním modelu připomínajícím protokol HTTP. Každá transakce se skládá z požadavku, který vyvolá nějakou metodu nebo funkci na serveru, a z alespoň jedné odpovědi. V tomto příkladu transakce začíná tím, že softphone Alice zasílá žádost `INVITE`, adresovanou SIP URI Boba. `INVITE` je příklad metody SIP, která specifikuje akci, kterou požaduje koncový bod (Alice) od serveru (Bob). Žádost `INVITE` má svou hlavičku, a ta obsahuje řadu polí, která slouží pro poskytnutí další informace o zprávě.

```
INVITE sip:bob@biloxi.com SIP/2.0
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bK776asdhds
Max-Forwards: 70
To: Bob <sip:bob@biloxi.com>
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710@pc33.atlanta.com
CSeq: 314159 INVITE
Contact: <sip:alice@pc33.atlanta.com>
Content-Type: application/sdp
Content-Length: 142
```

Obrázek 3.2: Zpráva `INVITE` [1]

První řádek textové zprávy obsahuje název metody (`INVITE`). Na následujících řádkách jsou uvedena jednotlivá pole hlavičky. Tento příklad obsahuje minimální požadovanou sadu. Pole hlavičky jsou stručně popsána níže:

- **Via** obsahuje adresu (`pc33.atlanta.com`), na které Alice očekává odpověď. Obsahuje také parametr `branch`, který identifikuje tuto transakci.
- **To** obsahuje zobrazované jméno (Bob) a SIP URI (`sip:bob@biloxi.com`), ke kterému byla žádost původně zaměřena.
- **From** také obsahuje zobrazované jméno (Alice) a SIP URI (`sip:alice@atlanta.com`) koncového bodu. Toto pole má také parametr `tag` obsahující náhodný řetězec (`1928301774`), který se používá pro identifikační účely.
- **Call-ID** obsahuje globálně unikátní identifikátor pro tento hovor, generovaný kombinací náhodného řetězce a hostitelského jména nebo IP adresy softphonu.
- **CSeq** obsahuje celé číslo a název metody. **CSeq** číslo se inkrementuje pro každou novou žádost v dialogu a představuje tradiční pořadové číslo.
- **Contact** obsahuje SIP URI, který představuje přímou cestu ke kontaktu Alice a je obvykle složen z uživatelského jména a plně kvalifikovaného doménového jména. Zatímco pole **Via** říká dalším prvkům, kam poslat odpověď, pole **Contact** říká, kam posílat budoucí požadavky.
- **Max-forwards** slouží k omezení počtu přesměrování, který může žádost urazit na cestě k cíli. Skládá se z celočíselného čísla, které se po každém přesměrování dekrementuje.
- **Content-Type** popisuje typ těla zprávy.
- **Content-Length** obsahuje délku těla zprávy v bajtech.

Podrobnosti o relaci, jako je typ média, kodek nebo vzorkovací frekvence, nejsou popsány pomocí SIP. Tyto informace jsou obvykle zapsané do těla SIP zprávy pomocí protokolu Session Description Protocol (**SDP**).

Jelikož softphone Alice nezná umístění Boba nebo SIP serveru v `biloxi.com` doméně, odešle zprávu `INVITE` SIP serveru, který obsluhuje doménu Alice, a to je `atlanta.com`. Je to typ SIP serveru, známý jako proxy server. Proxy server přijímá žádosti SIP a předává je jménem klienta (koncového bodu). V tomto příkladu proxy server obdrží žádost `INVITE` a odešle odpověď `100 (Trying)` zpět do softphonu Alice. Ta označuje, že žádost `INVITE` byla přijata a že zástupce (proxy server) pracuje jménem klienta na cestě té zprávy do cíle. Odpovědi v SIP používají třímístný kód následovaný popisnou textovou frází. `atlanta.com` proxy server vyhledá proxy server na `biloxi.com` (například provedením určitého typu DNS vyhledávání), získá IP adresu `biloxi.com` proxy serveru a přesměruje na něj žádost `INVITE`. Před tím `atlanta.com` proxy server ještě přidává do hlavičky další pole, které obsahuje jeho vlastní adresu. `biloxi.com` proxy server přijímá žádost `INVITE` a posílá odpověď `100 (Trying)` zpět do `atlanta.com` serveru, aby mu oznámil, že obdržel žádost `INVITE` a teď ji zpracovává. Pak konzultuje databázi, která obsahuje aktuální IP adresu Boba. `biloxi.com` server přidává do hlavičky další pole s jeho vlastní adresou a posílá zprávu na SIP telefon Boba.

Bobův SIP telefon obdrží zprávu `INVITE` a upozorní Boba na příchozí hovor od Alice, aby se mohl rozhodnout, zda chce přijmout hovor. Bobův telefon přitom zazvoní a oznámí to pomocí zprávy `180 (Ringing)`, která je směrována přes oba dva proxy v opačném směru. Každý proxy používá pole hlavičky **Via** k určení, kam poslat odpověď, a odstraní svou vlastní adresu z toho pole. Výsledkem je, že i když použití DNS a vyhledávání umístění služby bylo nutné na začátku cesty zprávy `INVITE`, zpráva `180 (Ringing)` už může být poslána volajícímu bez vyhledávání a zpracování proxy servery.

Když softphone Alice obdrží odpověď `180 (Ringing)`, předá tyto informace Alici například pomocí zvukového vyzváněcího tónu nebo zobrazení zprávy na obrazovce.

V tomto příkladu se Bob rozhodne hovor přijmout. Když zvedne telefon, odešle se odpověď `200 (OK)`, která oznámí protistraně, že hovor byl přijat. Tělo zprávy `200 (OK)` obsahuje SDP popis typu relace, kterou je Bob ochoten vytvořit s Alicí. Výsledkem je dvousměrná výměna zpráv SDP: od Alice k Bobovi a zpět. Tato výměna poskytuje základní vyjednávací schopnosti a je založena na jednoduchém modelu nabídky / odpovědi. Pokud by Bob nechtěl hovor přijmout nebo byl zaneprázdněn, namísto `200 (OK)` se zašle chybová hláška a tím pádem se žádná mediální relace nezaloží.

V uvedeném případě zpráva `200 (OK)` je směrována zpět přes dva proxy servery a je

přijata softphonem Alice, který pak zastaví vyzváněcí tón a tím indikuje, že hovor byl přijat. Nakonec softphone Alice odešle potvrzovací zprávu (`ACK`) na Bobův SIP telefon, aby potvrdil příjem poslední odpovědi `200 (OK)`. V tomto příkladu je `ACK` odeslán ze softphonu Alice do Bobova SIP telefonu přímo, tj. nikoliv přes proxy, protože koncové body v této fázi už vzájemně znají své adresy z obsahu pole **Contact** hlavičky zprávy `INVITE`. Tím se dokončí takzvaný „trojcestný handshake“ `INVITE/200/ACK` používaný pro vytvoření SIP relace. Mediální relace Alice a Boba nyní začala a oni si navzájem posílají mediální pakety ve formátu, na kterém se dohodli při výměně SDP.

Během hovoru se Alice nebo Bob mohou rozhodnout změnit vlastnosti mediální relace. To je dosaženo zasláním zprávy re-`INVITE` obsahující nový popis médií. Tato zpráva odkazuje na existující dialog tak, aby druhá strana věděla, že namísto vytvoření nové relace se mění parametry relace existující. Jestliže druhá strana chce přijmout změnu, pošle `200 (OK)`. Žadatel na to reaguje zasíláním `ACK`. Pokud druhá strana změnu nepřijme, odešle chybovou hlášku, jakým je například `488 (Not Acceptable Here)`, která se také potvrdí `ACK`. Selhání re-`INVITE` však nezpůsobuje selhání stávajícího hovoru - relace pokračuje a používá dříve dohodnuté charakteristiky.

Na konci hovoru Bob zavěsí (odpojí se) jako první a vygeneruje tím pádem zprávu `BYE`, která se opět pošle přímo Alici. Alice potvrzuje přijetí odpovědí `BYE` zprávou `200 (OK)`, která zároveň ukončuje relaci.

Registrace je další běžnou operací v SIP. Je jedním ze způsobů, jak `biloxi.com` server získá informaci o aktuálním umístění Boba. Po inicializaci odesílá Bobův SIP telefon v pravidelných intervalech zprávy `REGISTER` na server v `biloxi.com` doméně známý jako SIP registrátor. Zprávy `REGISTER` spojují Bobův SIP URI se zařízením, do kterého je aktuálně přihlášen. Registrátor zapíše tuto vazbu do databáze, odkud ji pak může přečíst proxy server.

Kapitola 4

Případy použití a požadavky pro nahrávání médií na bázi SIP (SIPREC)

4.1 Úvod

Velmi důležitým požadavkem obchodní komunikace je nahrávání relace. V dispečerských systémech musí být všechna volání zaznamenána z důvodů dodržování bezpečnosti, regulací a předpisů.

Nahrávání se obvykle provádí zasláním kopie médií relace do záznamové jednotky. Tato kapitola uvádí požadavky na rozšíření SIP, která spravují doručení RTP médií do záznamové jednotky, označované jako nahrávání médií na bázi SIP (krátce **SIPREC**). V rámci této práce budou uvažovány pouze případy, kdy uživatelé vědí o tom, že jejich hovor je zaznamenáván, a to záměrně (tzv. aktivní nahrávání).

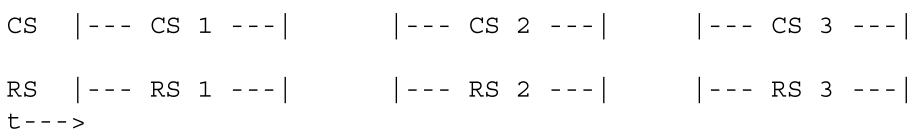
V této kapitole se zkoumají požadavky na použití SIP mezi klientem nahrávajícím relace (SRC) a serverem nahrávajícím relace (SRS) pro řízení záznamů médií, která jsou přenášena v rámci komunikační relace. Zaznamenané relace mohou být libovolné relace RTP médií, včetně hlasu, DTMF, videa a textu. Tato bakalářská práce se zaměřuje převážně na nahrávání hlasových hovorů.

Archivovaný záznam relace se obvykle skládá z mediálního obsahu komunikační relace a jejích metadat. Metadata komunikační relace umožňují později prohledávat a filtrovat nahrávané archivy. Umožňují také přehrávání relace „smysluplným“ způsobem, například se správnou synchronizací mezi médii.

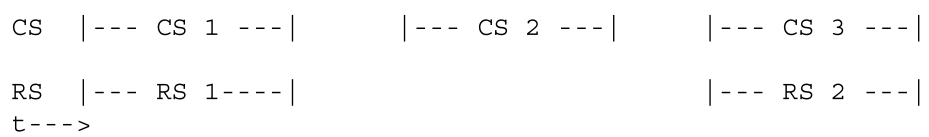
4.2 Případy použití

V této sekci jsou popsány případy použití SIPREC nahrávání, které se vztahují k dispečerskému řešení „KONOS“.

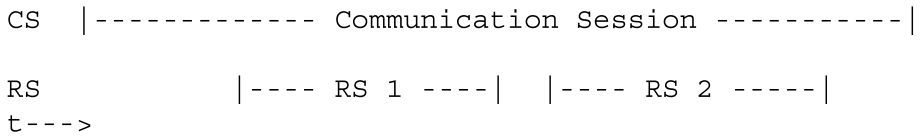
1. **Full-Time Recording** - jedna nahrávací relace pro každou komunikační relaci. V terminálu „KONOS“ se obvykle používá tento způsob nahrávání.



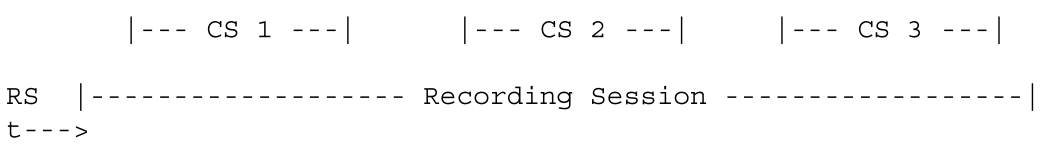
2. **Selective Recording** - nahrávací relace se spouští pouze když je vytvořena komunikační relace, která má být zaznamenána. V následujícím příkladu jsou zaznamenány komunikační relace 1 a 3, nikoliv 2.



3. **Spuštění / zastavení nahrávání během komunikační relace** - nahrávání začne během komunikační relace, a to buď ručně (např. tlačítko na telefonu uživatele) nebo automaticky prostřednictvím aplikace. Nahrávací relace končí buď během komunikační relace nebo po jejím ukončení. Jednu komunikační relaci mohou zaznamenávat jedna nebo více nahrávacích relací.

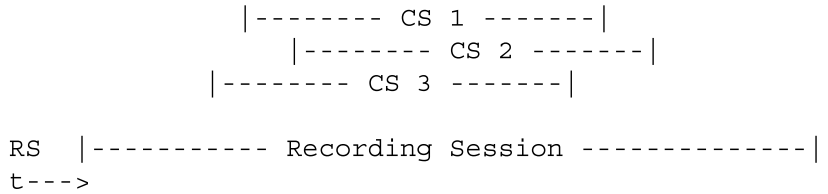


4. **Trvalé nahrávání (Persistent Recording)** - jedna nahrávací relace zachycuje jednu nebo více komunikačních relací.



Obrázek 4.4: Trvalé nahrávání [2]

Nahrávácí relace zaznamenává nepřetržitě bez přerušení. Doba, kdy neprobíhá žádná komunikační relace, musí být reprodukována při přehrávání (např. zaznamenáním ticha). Dispečerské systémy často používají tuto techniku, když se jedná o konference nebo předání existujícího hovoru jinému dispečerovi. Existují i případy, kdy jedna nahrávací relace zaznamenává více souběžných komunikačních relací.



Obrázek 4.5: Nahrávání více souběžných CS [2]

5. **Ovládání záznamu v reálném čase (Real-time Recording Controls)** - v případě aktivního nahrávání mohou ochrana soukromí nebo bezpečnostní důvody vyžadovat vynechání nahrávání určité části konverzace. Příkladem ovládání záznamu v reálném čase je pozastavení / obnovení (pause / resume) nahrávání.

6. **Záznam komplexních volacích scénářů** - pokud je například jeden zaznamenaný hovor spojen s dalším voláním, jež musí být také zaznamenáno.

7. **Vysoká dostupnost a nepřetržitý záznam** – specifické scénáře nasazení představují různé požadavky na dostupnost systému, manipulaci s chybami atd., včetně následujících:

   - SRS musí být vždy k dispozici v době nastavení hovoru.
   - Nesmí dojít ke ztrátě záznamu médií, dokonce i při selhání SRS.
   - V případě selhání záznamu musí být komunikační relace ukončena (nebo musí být poskytnuta vhodná oznámení stranám).

4.3 Požadavky na implementaci SIPREC

Podle [2], mechanismus **musí**:

1. poskytovat prostředky pro použití protokolu SIP pro vytvoření, udržování a ukončení nahrávací relace mezi SRC a SRS;
2. podporovat možnost nahrávat všechny komunikační relace kompletně;
3. podporovat schopnost nahrávání vybraných komunikací dle lokálních předpisů;
4. podporovat schopnost nahrávat vybrané úseky vybraných komunikačních relací;
5. podporovat možnost nahrávat CS bez ztráty médií RS z důvodu přípravy nahrávání nebo časování;
6. podporovat schopnost nahrávat IVR (automatické systémy bez operátora);
7. podporovat nahrávání následujících typů: hlas, DTMF, video, text;
8. podporovat schopnost SRC dodávat sloučené audio streamy z více komunikačních relací (konference) SRS;
9. podporovat schopnost SRC dodávat sloučené audio streamy z různých stran dané komunikační relace SRS;
10. podporovat schopnost dodávat SRS více mediálních streamů pro danou CS;
11. podporovat schopnost pozastavení / obnovení přenosu a sběru RS médií;
12. obsahovat prostředky pro poskytování SRS metadat popisujících nahrávané CS, včetně použitých médií a identifikátorů zúčastněných stran;
13. poskytovat SRS prostředky pro možnost korelace RS médií s médii účastníků CS;
14. být transparentním vzhledem k transportnímu protokolu;
15. podporovat prostředky k zastavení záznamu;
16. poskytovat prostředky pro upozornění uživatelského agenta a dotaz, zda se CS může nahrávat;
17. poskytovat prostředky pro upozornění uživatelského agenta a dotaz, zda se CS může nahrávat, během integrovaných akcí start/stop/pause/resume;
18. podporovat prostředky indikace a její vhodné zobrazení na uživatelském rozhraní, jestli se CS nahrává;

19. poskytovat prostředky, jak metadata postupně zasílat SRS během CS;
20. poskytovat prostředky pro synchronizaci zaznamenaných mediálních streamů a metadat;
21. poskytovat prostředky pro synchronizaci mezi několika zaznamenanými mediálními streamy;
22. poskytovat prostředky pro propojení ovládacích prvků záznamů, jako je start / stop / pause / resume, a záznam přesného času těchto akcí;
23. poskytovat SRS prostředky k autentizaci SRC při zahájení RS;
24. poskytovat SRC prostředky k autentizaci SRS při zahájení RS;
25. poskytovat prostředky pro kontrolu, že integrita metadat odeslaných z SRC do SRS je přesnou reprezentací původních metadat CS;
26. poskytovat prostředky pro kontrolu, že integrita médií odeslaných z SRC do SRS je přesnou reprezentací původních médií CS;
27. poskytovat prostředky pro zajištění důvěrnosti metadat zaslaných ze SRC do SRS;
28. poskytnout prostředky pro zajištění důvěrnosti RS;
29. podporovat schopnost dodávat SRS více mediálních streamů stejného typu najednou.

Podle [2], mechanismus **nesmí**:

1. bránit nasazení s vysokou dostupností (High availability);
2. bránit použití tónů nebo jiného upozornění během nebo na začátku nahrávání CS jako oznámení účastníkům, že hovor je zaznamenáván nebo může být zaznamenán.

Kapitola 5

Architektura pro SIPREC

5.1 Úvod

Tato kapitola popisuje architekturu pro implementaci protokolu SIPREC. Zaměřuje se na vytvoření nahrávací relace mezi SRC a SRS.

Jakmile SRS obdrží replikovaná média a metadata záznamu, obvykle je uloží pro pozdější přehrávání. Činnosti týkající se archivace a přehrávání hovorů nejsou v rámci této bakalářské práce popsány.

Nahrávací relace mezi SRC a SRS využívá běžných pravidel pro vytvoření dialogů založených na zprávách `INVITE`. Pro popis atributů médií, které mají být použity během relace, používá SDP.

5.2 Architektura nahrávání relace

V této sekci je popsána architektura, které je potřebná pro nahrávání komunikační relace.

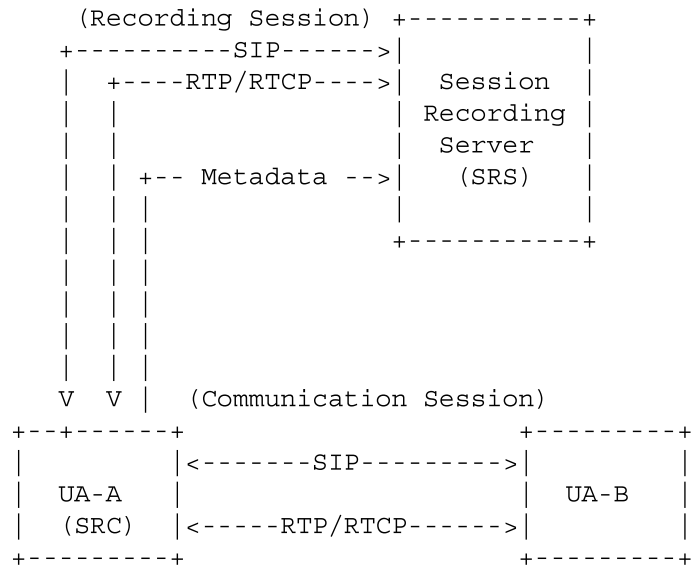
5.2.1 Umístění SRC

Tato část obsahuje příklad umístění SRC, které je používané v dispečerském řešení „KONOS“.

5.2.1.1 Koncový bod v roli SRC

Koncový bod SIP (UA) může působit jako SRC. V takovém případě koncový bod odešle replikovaná média SRS. V případě řešení „KONOS“ koncovým bodem, hrajícím roli SRC, je dispečerský terminál.

Pokud se koncový bod rozhodne zahájit RS, učiní tak zasláním zprávy `INVITE` SRS.



Obrázek 5.1: Koncový bod v roli SRC [3]

5.2.2 Založení nahrávací relace

Nahrávací relace může být zahájena buď SRC nebo SRS. Nahrávání by mělo být zcela nezávislé na komunikační relaci, která je nahrávána, a to jak na úrovni SIP dialogu, tak na úrovni samotné relace. Co se týká sjednávání parametrů médií, měly by být používány běžné funkce SIP a SDP.

V případě dispečerského řešení „KONOS“ je nahrávání vždy inicializováno SRC.

5.2.2.1 Nahrávání iniciované SRC

Když SRC zahájí RS na SRS, měly by podle [3] být provedeny tyto operace:

1. SRS je poskytován URI.
2. Dialog se inicializuje posláním zprávy `INVITE` SRS.
3. Do zprávy `INVITE` je přidána indikace, že je CS založena za účelem záznamu přidruženého média.
4. Do SDP zprávy je vložen atribut „`a=sendonly`“ pro každou mediální linku v případě, že nahrávání replikovaného média nastává okamžitě, nebo atribut „`a=inactive`“ v případě, že klient není připraven média vysílat.
5. Mediální streamy, které jsou určené k nahrávání, jsou replikovány a posílány na nahrávací server.

5.2.2.2 Pozastavení / obnovení nahrávací relace

SRS nebo SRC mohou pozastavit nahrávání změnou atributu směru SDP na „`inactive`“ a obnovit nahrávání změnou atributu zpět na „`recvonly`“ nebo „`sendonly`“.

5.2.2.3 Slučování mediálních streamů

V základní relaci zahrnující pouze audio jsou mezi oběma uživatelskými agenty zapojenými do přenosu médií obvykle dva streamy audio / RTP - jeden v každém směru. Během nahrávání těchto médií oba streamy mohou ale nemusí být smíchány SRC před přenosem na SRS. V případě, když nejsou sloučeny, jsou SRS zasílány dva samostatné streamy a SDP zpráva je musí popisovat. Jinak je SRS odeslán společný mediální stream. V dispečerském řešení „KONOS“ SRC slučování mediálních streamů provádí.

5.2.3 Upozornění nahrávaných uživatelských agentů

Uživatel, zapojený do relace, která má být zaznamenána, obvykle dostává oznámení na začátku relace nebo může obdržet upozorňovací tóny v médiích. SRC poskytuje metody pro oznámení všem SIP uživatelským agentům, pro které replikuje přijatá média za účelem záznamu. Pokud SRC působí jako koncový bod SIP (viz 5.2.1.1), měl by upozornit i lokálního uživatele.

5.2.4 Zabránění nahrávání SIP relace

Během počátečního založení relace nebo během už založené relace by uživatelský agent měl mít možnost oznámit své preference ohledně nahrávání.

5.3 Bezpečnostní úvahy

1. Nahrávací relace je v zásadě standardní SIP dialog a mediální relace, a proto využívá stávajících bezpečnostních mechanismů SIP protokolu pro zajištění nahrávání relace a přenosu metadat.

2. Tato architektura se používá pouze v případech, kdy uživatelé jsou upozorňováni na to, že je jejich hovor nahráván.

3. Je zodpovědnost SRS chránit přijatá replikovaná média a zaznamenávaná metadata. Uložený obsah musí být chráněn šifrou alespoň tak silnou (nebo silnější) než původní obsah.

4. Ochrana RS by neměla být slabší než ochrana CS.

5. Je nezbytné, aby SRC ověřil SRS, protože klient musí mít jistotu, že nahrává CS na správný nahrávací server. Je méně důležité, aby SRS ověřoval SRC, ale implementace musí mít možnost provádět vzájemnou autentizaci.

6. V některých prostředích je vyžadováno nedešifrovat a znovu nešifrovat média. To znamená, že musí být vyjednán a používán stejný šifrovací klíč médií v rámci CS a RS. Pokud jsou z nějakého důvodu média dešifrována na CS a jsou znovu šifrována na RS, musí být použit nový klíč. [3]

Kapitola 6

Session Recording Protocol (SIPREC)

6.1 Úvod

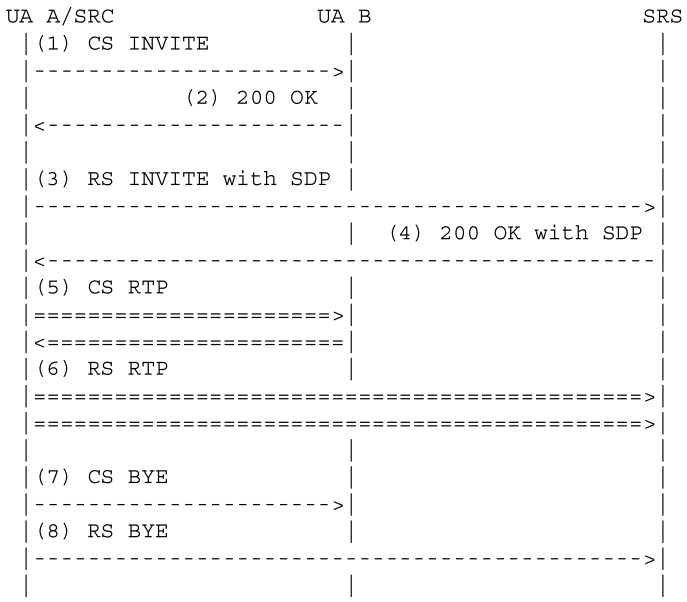
Tato kapitola specifikuje mechanismus nahrávání komunikační relace (CS) doručením médií a metadat CS do záznamové jednotky v reálném čase. V souladu s architekturou popsanou v předchozí kapitole 5, SIPREC používá protokoly SIP, SDP a RTP k vytvoření nahrávací relace (RS) mezi nahrávacím klientem (SRC) a nahrávacím serverem (SRS) na záznamové jednotce. SIP se také používá k doručování metadat do záznamové jednotky v rámci RS.

6.2 Sestavení nahrávací relace

V této sekci je uveden přehled činností, spojených s nahráváním komunikační relace založeném na protokolu SIPREC.

6.2.1 Doručování zaznamenaných médií

Příklad toku hovoru na obrázku 6.1 ukazuje uživatelského agenta (UA) A, působícího jako SRC vytvářející RS směrem k SRS. SRC si může, nezávisle na CS, vybrat, kdy založí RS, v příkladu to dělá po založení CS.



Obrázek 6.1: UA v roli SRC [5]

Výsledkem nahrávací relace na předchozím obrázku jsou úspěšně doručené RTP pakety od SRC na SRS, což znamená úspěšně nahraný hovor. Pro přehlednost, část zpráv `200 OK` a `ACK` nejsou zobrazeny.

SRC může použít jednu RS pro záznam více CS. Pokaždé, když chce nahrát nový hovor, aktualizuje RS novou SDP nabídkou pro přidání nových zaznamenaných streamů k RS a aktualizaci metadat.

6.2.2 Doručování metadat záznamu

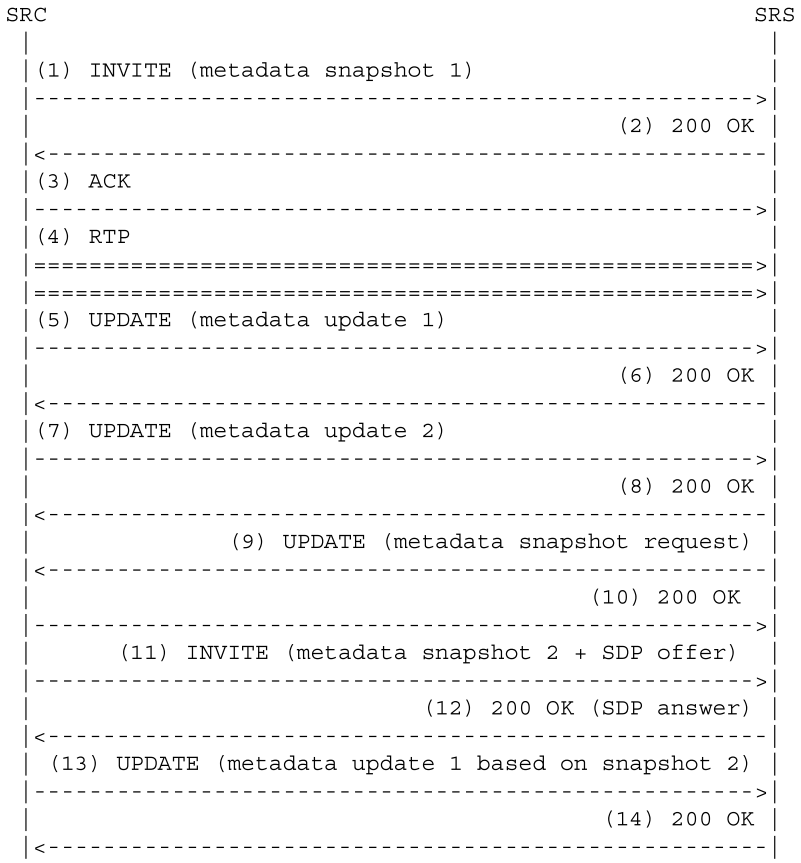
SRC je zodpovědný za doručování metadat na SRS. Metadata jsou posílána v těle zpráv `INVITE`, `UPDATE` a re-`INVITE`:

- SRC může poskytnout počáteční snapshot (snímek) metadat o zaznamenaných mediálních streamech v obsahu počáteční zprávy `INVITE` v RS.
- Následné aktualizace metadat mohou být reprezentovány jako stream událostí ve zprávách re-`INVITE` nebo `UPDATE` zasílaných SRC. [4]

Některá metadata, například atributy zaznamenaného mediálního streamu, jsou umístěna v SDP RS.

SRS má možnost požádat SRC o aktualizaci snapshotu metadat, například, když SRS ztratí aktuální stav kvůli vnitřnímu selhání. SRS může volitelně připojit informace o

důvodu spolu s požadavkem na snapshot. To umožňuje SRC i SRS synchronizovat stavy, takže další aktualizace budou založeny na nejnovějším snapshotu metadat. Podobně jako obsah metadat, je požadavek na snapshot přenášen jako tělo zpráv `UPDATE` nebo `INVITE` odeslaných SRS.



Obrázek 6.2: Doručování metadat via SIP UPDATE [5]

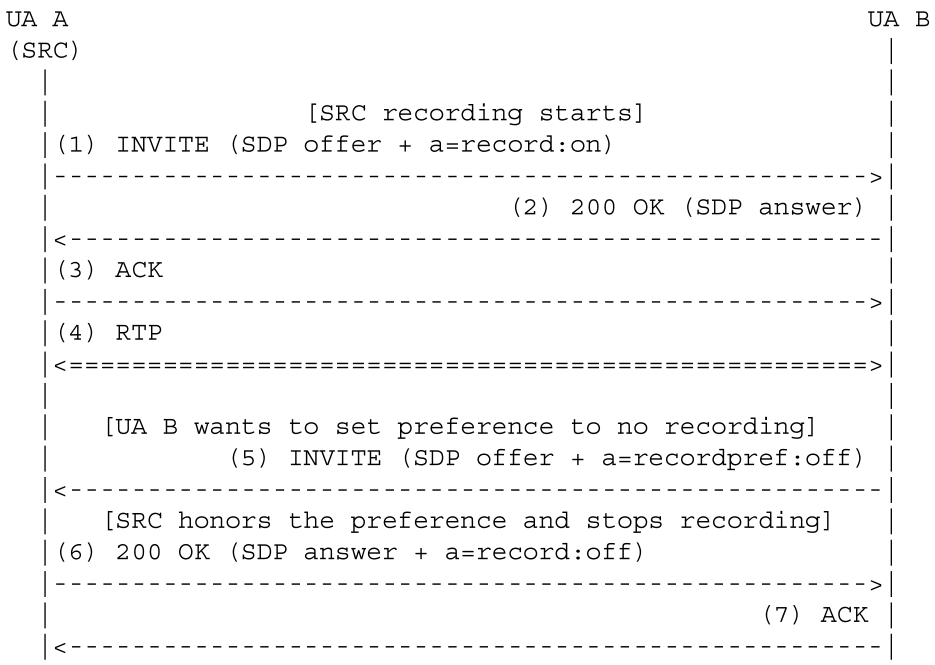
6.2.3 Příjem indikace nahrávání a poskytování preferencí

SRC je zodpovědný za poskytování indikace nahrávání účastníkům CS.

RA UA podporuje příjem indikací nahrávání prostřednictvím atributu SDP „`a=record`“ a je schopen zadat preference nahrávání v CS prostřednictvím atributu SDP „`a=recordpref`“. Pomocí prvního atributu SRC uvádí, že nahrávání probíhá. Druhý atribut slouží pro to, aby UA oznámil své preference nahrávání. UA, který nechce být zaznamenán, může být upozorněn, že dochází k nahrávání. Pokud k tomu dojde, jediným mechanismem UA, jak se vyhnout zaznamenávání, je ukončit jeho účast v relaci.

Pro ilustraci použití atributů slouží následující příklad. Pokud UA A zahajuje hovor na UA B a je zároveň SRC, který provádí nahrávání, pak UA A poskytuje indikaci záznamu v SDP nabídce pomocí atributu „`a=record:on`“. Když UA B obdrží SDP nabídku,

uvidí, že na protistraně této relace probíhá nahrávání. Protože UA B není SRC a na začátku hovoru neposkytuje žádnou preferenci záznamu, SDP odpověď (2) neobsahuje atribut „`a=record`“ ani „`a=recordpref`“. Po založení hovoru se UA B však rozhodne změnit preference tak, aby nahrávání skončilo. Proto odešle zprávu re-`INVITE` s atributem „`a=recordpref`“ nastaveným na „`off`“. Je na SRC, zda bude počítat s preferencemi. V uvedeném příkladu se SRC rozhodne zastavit nahrávání a aktualizuje atribut „`a=record`“ v odpovědi SDP. [5]



Obrázek 6.3: Indikace nahrávání a poskytování preferencí [5]

6.3 SIP manipulace

Tato sekce popisuje kroky, které provádějí účastníci nahrávací relace v rámci protokolu SIP pro úspěšné nahrávání hovoru v závislosti na jejich roli.

6.3.1 Kroky potřebné pro SRC

6.3.1.1 Založení nahrávací relace

RS je normální SIP relace se specifickými rozšířeními. SRC zakládá RS zasláním SIP zprávy `INVITE` SRS. SRC a SRS jsou identifikovány v hlavičkách **From** a **To**.

SRC musí připojit tag „`+sip.src`“ do **Contact** URI pro všechny nahrávací relace. Tento tag je určen pro oznámení, že dialog je vytvořen pro účely nahrávání, a kromě zprávy

`INVITE` může být součástí i zprávy `REGISTER` pro oznámení registrátorovi, že se jedná o SRC.

Vzhledem k tomu, že SIPREC rozšíření jsou nepovinná, je zaveden další tag „`siprec`“, který mohou přijmout jen SRC a SRS. Při založení RS musí SRC přidat tag „`siprec`“ do **Require** hlavičky pro to, aby uživatelské agenty, nepodporující SIPREC rozšíření, jednoduše odmítly požadavek `INVITE` s odpovědí `420 (Bad Extension)`.

Když SRC obdrží zprávu `INVITE`, musí založit RS pouze v případě, když jsou ve zprávě zahrnuty oba dva tagy „`+sip.srs`“ a „`siprec`“. [5]

6.3.1.2 Rozšíření SIP pro indikaci nahrávání a preference

Pro CS musí SRC poskytnout indikace nahrávání všem účastníkům CS. UA v CS může naznačovat, že si je vědom nahrávání poskytnutím tagu „`record-aware`“, a SRC musí poskytnout indikaci nahrávání v novém SDP atributu „`a=record`“ (viz 6.2.3). V případě nepřítomnosti tagu „`record-aware`“ musí SRC poskytnout indikaci nahrávání jinými prostředky, jako například přehrání upozorňovacího tónu nebo výzvy účastníků k souhlasu.

6.3.2 Kroky potřebné pro SRS

SRS zakládá RS zasláním SIP zprávy `INVITE` SRC. SRS a SRC jsou identifikovány v hlavičkách **From** a **To**.

Když SRS obdrží zprávu `INVITE`, musí založit RS pouze v případě, když jsou ve zprávě zahrnuty oba dva tagy „`+sip.srs`“ a „`siprec`“.

SRS musí připojit tag „`+sip.srs`“ do **Contact** URI pro všechny nahrávací relace. Kromě toho, když SRC odešle `REGISTER` zprávu registrátorovi, ta by měla obsahovat tag „`+sip.srs`“ pro označení, že se jedná o SRS. [5]

6.3.3 Kroky potřebné pro RA uživatelské agenty

RA UA je účastníkem CS, který podporuje rozšíření SIP a SDP pro příjem indikací nahrávání a pro vyžádání preferencí nahrávání. Pro koncového uživatele musí poskytnout indikaci nahrávání prostřednictvím vhodného uživatelského rozhraní, včetně uvedení, zda nahrávání je zapnuto, vypnuto nebo pozastaveno, a to pro každé médium. Příslušná uživatelská rozhraní mohou zahrnovat oznámení, že používané zařízení podléhá záznamu.

6.4 SDP manipulace

Tato sekce popisuje kroky, které provádějí účastníky nahrávací relace v rámci protokolu SDP pro úspěšné nahrávání hovoru v závislosti na jejich roli.

6.4.1 Kroky potřebné pro SRC

SRC a SRS sledují SDP model nabídky / odpovědi.

6.4.1.1 Manipulace s SDP v RS

Vzhledem k tomu, že SRC neočekává příjem médií od SRS, obvykle nastavuje každý mediální stream SDP nabídky pouze pro odesílání médií nastavením atributu „`a=sendonly`“.

SRC posílá zaznamenané streamy účastníků SRS a SRC musí poskytnout na každém mediálním streamu atribut „`label`“ („`a=label`“). Tento atribut identifikuje každý zaznamenaný mediální stream a název, který obsahuje, je mapován na odkaz na tento stream v metadatech. [5]

Obrázek 6.4 ukazuje příklad SDP nabídky od SRC s nahrávanými audio i video streamy.

```
v=0
o=SRC 2890844526 2890844526 IN IP4 198.51.100.1
s=-
c=IN IP4 198.51.100.1
t=0 0
m=audio 12240 RTP/AVP 0 4 8
a=sendonly
a=label:1
m=video 22456 RTP/AVP 98
a=rtpmap:98 H264/90000
<allOneLine>
a=fmtp:98 profile-level-id=42A01E;
          sprop-parameter-sets=Z0IACpZTBYmI,aMljiA==
</allOneLine>
a=sendonly
a=label:2
m=audio 12242 RTP/AVP 0 4 8
a=sendonly
a=label:3
m=video 22458 RTP/AVP 98
a=rtpmap:98 H264/90000
<allOneLine>
a=fmtp:98 profile-level-id=42A01E;
          sprop-parameter-sets=Z0IACpZTBYmI,aMljiA==
</allOneLine>
a=sendonly
a=label:4
```

Obrázek 6.4: SDP nabídka od SRC s audio i video streamy [5]

6.4.1.1.1 Manipulace s aktualizacemi mediálního streamu

Během celé RS může SRC do ní přidávat a z ní odstraňovat zaznamenané streamy:

- Pro odstranění nahraného streamu z RS, odešle SRC novou SDP nabídku, kde je port mediálního streamu, který má být odstraněn, nastaven na nulu.
- Pro přidání nahraného streamu do RS, odešle SRC novou SDP nabídku s popisem nově přidaného mediálního streamu nebo opětovným použitím starého mediálního streamu, který byl dříve odstraněn.

SRC může dočasně přerušit nahrávání streamů. V tomto případě pošle novou SDP nabídku a nastaví atribut na „`a=inactive`“ pro každý zaznamenaný stream, který chce pozastavit. Pro obnovení nahrávání SRC odešle novou SDP nabídku a nastaví příslušný mediální stream na „`a=sendonly`“.

6.4.1.2 Indikace nahrávání v CS

SDP atribut „`record`“ se objevuje na úrovni médií nebo na úrovni relace buď v SDP nabídce nebo odpovědi. Pokud je atribut aplikován na úrovni relace, vztahuje se na všechny mediální streamy v SDP. Pokud je atribut aplikován na úrovni médií, indikace se vztahuje pouze na jeden mediální stream a má přednost před atributem nastaveným na úrovni relace.

Kdykoli se indikace nahrávání musí změnit, například při ukončení nahrávání, musí SRC poslat zprávu re-`INVITE` nebo `UPDATE` pro aktualizaci atributu SDP „`a=record`“.

Atribut „`record`“ může být nastaven na „`on`“, „`off`“ nebo „`paused`“.

6.4.1.3 Preference nahrávání v CS

Když SRC obdrží v SDP nabídce nebo odpovědi „`a=recordpref`“, budou se preference nahrávání řídit podle místního nastavení v SRC. Pokud SRC změní stav nahrávání, musí to oznámit v atributu „`a=record`“ v SDP odpovědi nebo nabídce.

6.4.2 Kroky potřebné pro SRS

Typicky SRS pouze přijímá RTP streamy z SRC, proto v SDP zprávách nastavuje každý mediální stream na „`a=recvonly`“. Pokud SRS není připraven přijímat zaznamenaný stream, nastaví jej jako neaktivní v SDP nabídce nebo odpovědi nastavením atributu „`a=inactive`“.

Obrázek 6.5 ukazuje příklad SDP odpovědi od SRS na SDP nabídku SRC z předchozího obrázku 6.4.

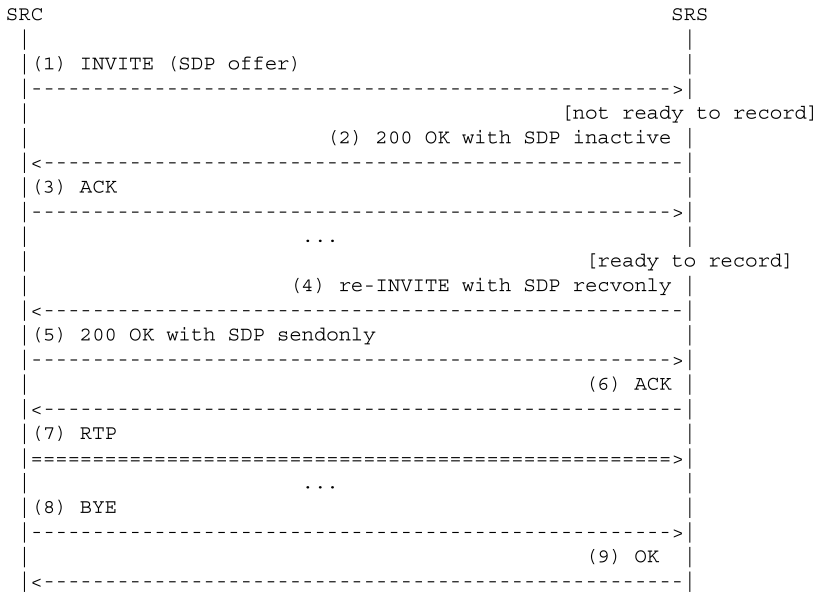
```
v=0
o=SRS 0 0 IN IP4 198.51.100.20
s=-
c=IN IP4 198.51.100.20
t=0 0
m=audio 10000 RTP/AVP 0
a=recvonly
a=label:1
m=video 10002 RTP/AVP 98
a=rtpmap:98 H264/90000
<allOneLine>
a=fmtp:98 profile-level-id=42A01E;
          sprop-parameter-sets=Z0IACpZTBYmI,aMljiA==
</allOneLine>
a=recvonly
a=label:2
m=audio 10004 RTP/AVP 0
a=recvonly
a=label:3
m=video 10006 RTP/AVP 98
a=rtpmap:98 H264/90000
<allOneLine>
a=fmtp:98 profile-level-id=42A01E;
          sprop-parameter-sets=Z0IACpZTBYmI,aMljiA==
</allOneLine>
a=recvonly
a=label:4
```

Obrázek 6.5: SDP odpověď od SRS s audio i video streamy [5]

Během celé RS z ní může SRS odstraňovat zaznamenané streamy. Pro odstranění nahraného streamu z RS, odešle SRS novou SDP nabídku, kde je port mediálního streamu, který má být odstraněn, nastaven na nulu.

SRS nesmí přidávat zaznamenané streamy do RS, když odešle novou nabídku SDP. Podobně, když SRS spustí RS, musí iniciovat `INVITE` bez nabídky SDP, aby SRC vygeneroval SDP nabídku se streamy, které mají být zaznamenány. [5]

Obrázek 6.6 ukazuje příklad, kdy SRS není zpočátku připraven přijímat zaznamenané streamy, ale později, když je připraven k nahrávání, aktualizuje RS.



Obrázek 6.6: SDP odpověď od SRS s a=inactive [5]

6.4.3 Kroky potřebné pro RA uživatelské agenty

6.4.3.1 Indikace nahrávání

Když RA UA obdrží SDP nabídku nebo odpověď, která obsahuje atribut „`a=record`“, poskytuje UA koncovému uživateli indikaci, zda je nahrávání zapnuté, vypnuté nebo pozastavené pro každé médium na základě naposledy přijatého SDP atributu „`a=record`“ pro toto médium.

Když CS prochází více UA, musí každý UA zapojený do CS, který si je vědom toho, že je CS zaznamenávána, poskytnout indikaci nahrávání prostřednictvím atributu „`a=record`“ všem ostatním stranám v CS.

6.4.3.2 Preference nahrávání

Účastník CS může nastavit preference nahrávání v CS, která má být zaznamenána nebo nezaznamenána při založení relace nebo během relace. Je zaveden další SDP atribut „`recordpref`“ a účastník CS může nastavit tento atribut preference nahrávání v jakékoli SDP nabídce nebo odpovědi v době založení relace nebo během relace. SRC není povinen řídit se preferencí nahrávání od účastníka na základě místních nastavení a účastník je informován o záznamu prostřednictvím SDP atributu „`a=record`“.

SDP atribut „`a=recordpref`“ se může objevit na úrovni médií nebo relace a může se objevit v SDP nabídce nebo odpovědi. Pokud je atribut aplikován na úrovni relace, vztahuje se na všechny mediální streamy v SDP. Pokud je atribut aplikován na úrovni médií,

vztahuje se pouze na jeden mediální stream a má přednost před atributem nastaveným na úrovni relace. UA může změnit preference nahrávání změnou atributu „`a=recordpref`“ v následné SDP nabídce nebo odpovědi. Absence atributu „`a=recordpref`“ znamená, že UA nemá preference nahrávání.

6.5 Metadata

Některé atributy metadat jsou obsaženy v SDP a jiné jsou obsaženy v dalším typu obsahu „`application/rs-metadata`“. Pro účely přenosu metadat je definován další typ „`disposition-type`“ hlavičky **Content-Disposition**. Hodnota je „`recording-session`“, což znamená, že „`application/rs-metadata`“ obsahuje metadata, která mají být zpracována v SRS. [4]

6.5.1 SRC manipulace

Podle [4], SRC musí poslat metadata SRS v RS a měl by je posílat, jakmile jsou k dispozici nebo kdykoli, když se změní. Případy, kdy může SRC počkat před odesláním metadat, zahrnují:

- Čekání na dokončení předchozí výměny metadat.
- Omezení rychlosti signalizace v RS.
- Odesílání metadat, když dojde ke klíčovým událostem. Netýká se to každou událost, která má vliv na metadata.

Metadata odeslaná SRC tvoří bud úplný snapshot nebo částečnou aktualizaci. Úplný snapshot popisuje všechna metadata spojená s RS. SRC může kdykoli odeslat úplný snapshot metadat, ale částečnou aktualizaci může odeslat pouze v případě, že byl dříve odeslán úplný snapshot.

SRC může posílat metadata v zprávě `INVITE`, zprávě `UPDATE` nebo odpovědi `200 OK` na `INVITE` zprávu od SRS. Pokud metadata obsahují odkaz na jakékoli SDP labely (popisky), musí požadavek obsahující metadata obsahovat také nabídku SDP, která tyto labely definuje.

Pokud SIP zpráva obsahuje jak SDP nabídku, tak i metadata, musí mít tělo požadavku typ obsahu „`multipart/mixed`“, přičemž jedna podřízená část těla obsahuje nabídku SDP a druhá obsahuje metadata.

SRC by měl přidat úplný snapshot metadat do počáteční zprávy `INVITE`, která založí RS. Pokud metadata ještě nejsou k dispozici (např. RS je založena bez založené CS), měl by SRC odeslat úplný snapshot metadat, jakmile budou metadata dostupná.

Pokud SRC obdrží požadavek na snapshot od SRS, musí jej okamžitě odeslat.

Obrázek 6.7 ilustruje příklad úplného snapshotu metadat odeslaného SRC v počáteční zprávě `INVITE`:

```
INVITE sip:recorder@example.com SIP/2.0
Via: SIP/2.0/TCP src.example.com;branch=z9hG4bKdf6b622b648d9
From: <sip:2000@example.com>;tag=35e195d2-947d-4585-946f-09839247
To: <sip:recorder@example.com>
Call-ID: d253c800-b0d1ea39-4a7dd-3f0e20a
CSeq: 101 INVITE
Max-Forwards: 70
Require: siprec
Accept: application/sdp, application/rs-metadata
Contact: <sip:2000@src.example.com>;+sip.src
Content-Type: multipart/mixed;boundary=foobar
Content-Length: [length]

--foobar
Content-Type: application/sdp

v=0
o=SRS 2890844526 2890844526 IN IP4 198.51.100.1
s=-
c=IN IP4 198.51.100.1
t=0 0
m=audio 12240 RTP/AVP 0 4 8
a=sendonly
a=label:1

--foobar
Content-Type: application/rs-metadata
Content-Disposition: recording-session

[metadata content]
```

Obrázek 6.7: Úplný snapshot metadat v INVITE zprávě [5]

6.5.2 SRS manipulace

SRS přijímá aktualizace metadat ze SRC ve zprávách `INVITE` a `UPDATE`. Vzhledem k tomu, že SRC může odesílat částečné aktualizace na základě předchozí aktualizace, SRS musí mít historii aktualizací ze SRC.

V případě interního selhání nemusí SRS rozpoznat částečnou aktualizaci ze SRC. SRS se může zotavit z interního selhání tím, že požádá o úplný snapshot metadat ze SRC. Některé chyby, jako například chyby syntaxe nebo sémantické chyby v metadatech, jsou pravděpodobně způsobeny chybou na straně SRC, a je také pravděpodobné, že ke stejné

chybě dojde znovu při požadování úplného snapshotu metadat. Aby se zabránilo opakování stejné chyby, SRS může jednoduše ukončit RS, když je v metadatech zjištěna chyba syntaxe nebo sémantická chyba.

SRS může explicitně požádat o úplný snapshot metadat zasláním zprávy `UPDATE`. Tato žádost musí obsahovat tělo s typem „`recording-session`“ hlavičky **Content-Disposition** a nesmí obsahovat tělo SDP. SRS nesmí požadovat úplný snapshot metadat v odpovědi na `UPDATE` nebo v jakékoli jiné SIP transakci. Obrázek 6.8 ukazuje příklad požadavku.

```
UPDATE sip:2000@src.example.com SIP/2.0
Via: SIP/2.0/UDP srs.example.com;branch=z9hG4bKdf6b622b648d9
To: <sip:2000@example.com>;tag=35e195d2-947d-4585-946f-098392474
From: <sip:recorder@example.com>;tag=1234567890
Call-ID: d253c800-b0d1ea39-4a7dd-3f0e20a
CSeq: 1 UPDATE
Max-Forwards: 70
Require: siprec
Contact: <sip:recorder@srs.example.com>;+sip.srs
Accept: application/sdp, application/rs-metadata
Content-Disposition: recording-session
Content-Type: application/rs-metadata
Content-Length: [length]

<?xml version="1.0" encoding="UTF-8"?>
  <requestsnapshot xmlns='urn:ietf:params:xml:ns:recording:1'>
    <requestreason xml:lang="it">SRS internal error</requestreason>
  </requestsnapshot>
```

Obrázek 6.8: Požadavek o metadata [5]

Kapitola 7

Použité nástroje a knihovny

V této kapitole jsou popsané nástroje a knihovny, které budou použity v praktické části práce.

7.1 PJSIP

PJSIP je open-source knihovna v jazyce C určená pro multimediální komunikaci, která implementuje standardní protokoly včetně SIP, SDP a RTP. Dispečerský systém „KONOS“ používá tuto knihovnu pro komunikační účely. [7]

7.2 Wireshark

Wireshark je aplikace pro analýzu provozu v počítačových sítích, která se používá pro zjišťování a odstraňování problémů, vývoj komunikačních protokolů a softwaru a studium síťové komunikace. Obsahuje stovky analyzátorů komunikačních protokolů a formátů, grafické uživatelské rozhraní a mnoho možností uspořádání a filtrování zobrazených informací. Umožňuje zachycovat výměnu zpráv a zkoumat jejich obsah. [8]

7.3 SIPp

SIPp je open-source testovací nástroj a generátor provozu pro protokol SIP. Obsahuje několik základních scénářů uživatelských agentů (klient UAC a server UAS) a zahajuje a řídí hovory pomocí SIP metod jako `INVITE`. Může také číst XML soubory uživatele obsahující scénář, který popisuje tok hovoru, a podle daného scénáře následně hovor nasimulovat. SIPp umí také přenos RTP audio médií. [9]

7.4 Phindows

Phindows je nástroj, který dovoluje používat platformy Microsoft Windows pro připojení a interakci s grafickými aplikacemi Photon běžícími na vzdáleném počítači Neutrino. V rámci této práce je používán pro sledování stavu kanálů nahrávací jednotky „ReDat“. [10]

7.5 Drachtio Siprec Recording Server

Drachtio Siprec Recording Server je open-source implementace SIPREC nahrávacího serveru (SRS) v jazyce JavaScript na základě „**drachtio**“ SIP serveru a proxy pro RTP provoz „**rtpengine**“. V rámci této práce je tento software používán pro simulaci části SIPREC protokolu na straně serveru. [11]

7.6 Qfsm

Qfsm je grafický editor napsaný v jazyce C++ pro vytváření konečných stavových automatů pomocí grafických nástrojů frameworku Qt. V rámci této práce je používán pro vytváření konečného stavového automatu pro obsluhu SIPREC hovorů. [12]

7.7 Enterprise Architect

Enterprise Architect je nástroj pro komplexní modelování softwaru. V rámci této práce je používán pro vytváření diagramů tříd, balíčků a nasazení pro modelování stávající a navržené implementace nahrávacích protokolů.

7.8 Twinkle

Twinkle je open-source aplikace pro hlasovou komunikaci přes VoIP protokol (takzvaný softphone - softwarový telefon). Je určen pro operační systémy Linux a používá sadu nástrojů frameworku Qt pro své grafické uživatelské rozhraní. Pro signalizaci hovorů používá SIP, mediální streamy jsou přenášeny prostřednictvím RTP.

7.9 Asterisk

Asterisk je open source aplikace, která implementuje telefonní ústřednu (PBX). V dispečerském řešení „KONOS“ se tato aplikace používá pro řízení SIP účtů, vyžadujících registraci. [17]

7.10 Hibernate

Hibernate je framework napsaný v jazyce Java, který umožňuje tzv. objektově relační mapování (ORM). V dispečerském řešení „KONOS“ se tento framework používá pro vytváření databázových tabulek z Java objektů. [16]

Kapitola 8

Stávající nahrávací řešení

8.1 Úvod

V současné době společnost „TTC Marconi“ ve svém dispečerském řešení „KONOS“ používá VoIP nahrávání „ReDat“ od společnosti „Retia“ s proprietárním rozhraním založeným na H.323 protokolu. Cílem této kapitoly je ukázat, jak toto nahrávání probíhá a jak vypadá komunikace mezi terminálem a záznamovou jednotkou.

8.2 Příklad komunikace

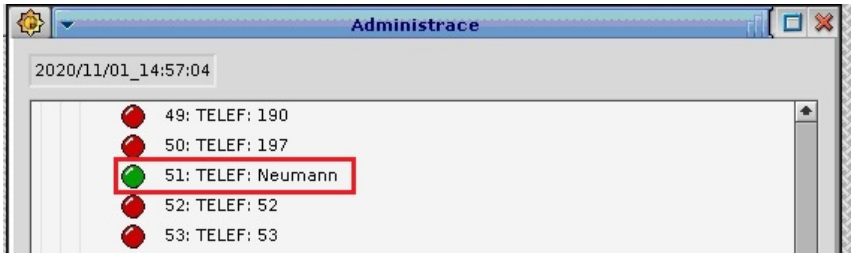
Příkladem bude jednoduchý scénář, podle kterého terminál volá na číslo s automatickým odpovídačem z testovací infrastruktury „TTC Marconi“ a následně mu posílá RTP pakety.

Přes softwarové rozhraní Phindows 7.4 je možné se přihlásit do záznamové jednotky a pozorovat její činnost. V konfiguračním souboru aplikace, v rámci které běží terminál, je definována IP adresa záznamové jednotky a kanál, na který bude hovor nahráván. V uvedeném příkladu je to kanál číslo 51. Když žádný hovor neprobíhá a tím pádem se nic nezaznamenává, kanál je označen červenou barvou.

Obrázek 8.1: Kanál nenahrává hovor

Nahrávání se zapíná automaticky při libovolném hovoru (viz 4.1). Probíhající nahrávání je

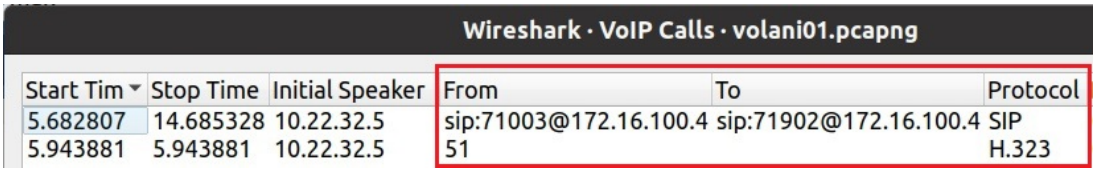
indikováno označením kanálu zelenou barvou. To se stane například i tehdy, když terminál zavolá na zmíněné číslo s automatickým odpovídačem.



Obrázek 8.2: Kanál nahrává hovor

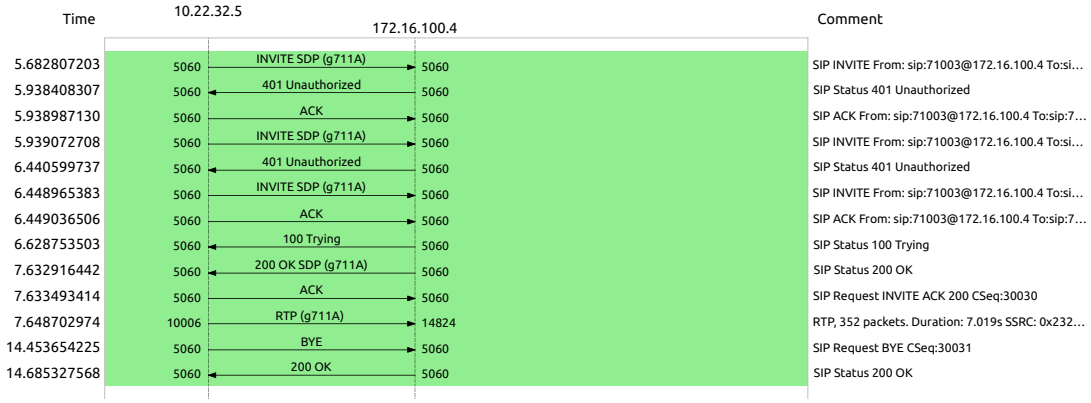
Existuje možnost zachytit pakety, které si terminál a záznamová jednotka navzájem posílají, a sledovat, jak vypadá tato komunikace. Jedním z nástrojů, který to umožňuje, je Wireshark 7.2.

Po zachycení komunikace mezi terminálem a záznamovou jednotkou je třeba zvolit záložku "Telefonie"a pak "VoIP hovory". Tam v tomto případě jsou zachycené dva hovory - SIP hovor mezi terminálem s číslem 71003 a telefonem s číslem 71902 (komunikační relace) a H.323 hovor mezi terminálem a záznamovou jednotkou (nahrávací relace).



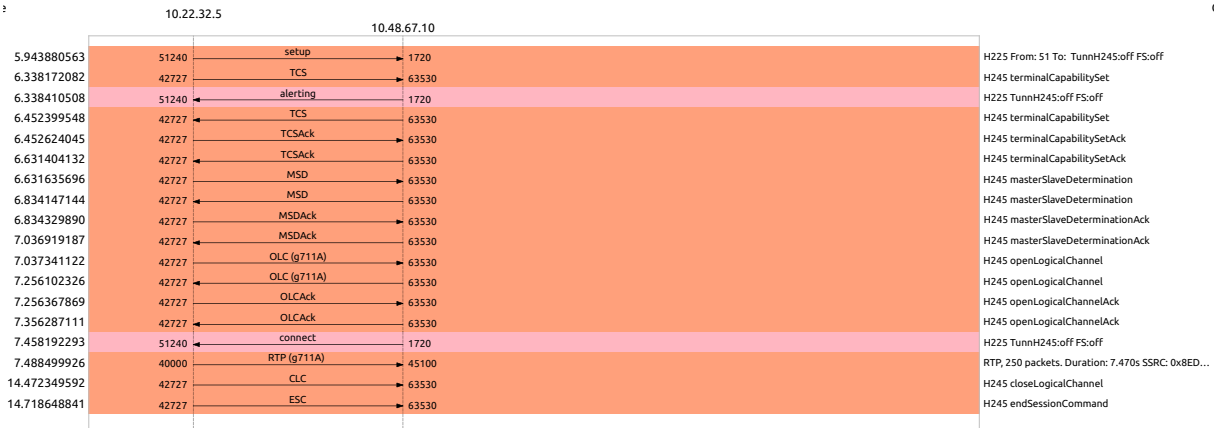
Obrázek 8.3: Wireshark VoIP telefonie

V obou případech byl hovor inicializován terminálem, který má IP adresu 10.22.32.5. Ten nejdříve zavolá na číslo 71902 s IP adresou 172.16.100.4, a zahájí standardní SIP hovor. RTP pakety jsou posílané jednosměrně, protože číslo s automatickým odpovídačem žádné audio posílat neumí.



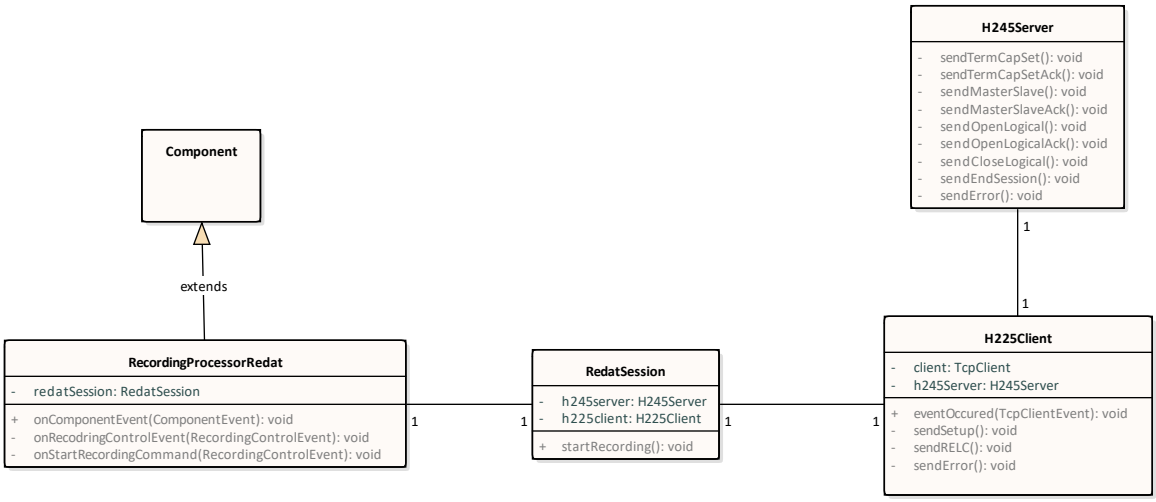
Obrázek 8.4: SIP hovor mezi terminálem a protistranou

Ve stejný okamžik terminál pomocí H.323 protokolu kontaktuje záznamovou jednotku na IP adrese 10.48.67.10. Zahajuje tím nahrávací relaci pro posílání kopií RTP paketů z probíhající komunikace. RTP pakety jsou zase posílány jednosměrně, protože záznamová jednotka je jenom přijímá a ukládá na server.



Obrázek 8.5: H.323 hovor mezi terminálem a záznamovou jednotkou

8.3 Diagram tříd zodpovědných za uvedenou komunikaci



Obrázek 8.6: Diagram tříd zodpovědných za uvedenou komunikaci

8.3.1 RecordingProcessorRedat

1. Pokud nastala událost typu `RecordingControlEvent`, vyvolá se metoda `onRecordingControlEvent()`.

2. Událost může mít různé stavy, pro stručnost je uveden jen příklad, když si klient přeje začít nahrávání - tehdy má událost stav `START_RECORDING` a vyvolá se metoda `onStartRecordingCommand()`.

3. Vyvolá se `redatSession.startRecording()`.

8.3.2 RedatSession

V této třídě se inicializují H225 klient a H245 server.

1. Na začátku nahrávací relace je terminál H225 klientem, který navazuje TCP spojení s nahrávací jednotkou „ReDat“.

2. Klient posílá `SETUP` zprávu, která obsahuje mimo jiné i číslo portu, na kterém chce terminál komunikovat - tím se terminál stává serverem.

3. Terminál v roli serveru posílá H245 zprávu `terminalCapabilitySet` - tato a několik dalších zpráv (viz 8.5) jsou analogií SDP zprávy, která je používána v SIP protokolu.

4. Po výměně zpráv, určujících parametry relace, se zahájí posílání RTP paketů (probíhá nahrávání).

5. Když se ukončí komunikační relace (hovor probíhající na terminálu), terminál v roli H245 serveru ukončuje nahrávací relaci.

8.3.3 H225Client

Tato třída je zodpovědná za vytváření H225 zpráv (například v metodě `sendSetup()` se vytváří `SETUP` zpráva).

8.3.4 H245Server

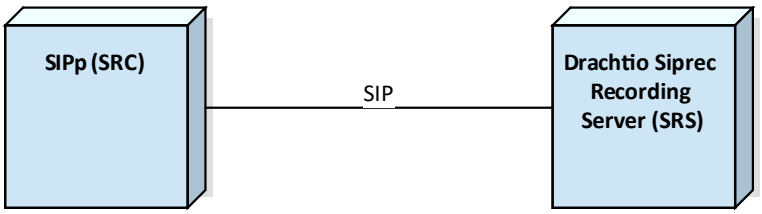
Tato třída je zodpovědná za vytváření H245 zpráv (například v metodě `sendTermCapSet()` se vytváří `terminalCapabilitySet` zpráva). Spolu s H225 klientem tvoří H.323 protokol [13].

Kapitola 9

Testování SIP signalizace se SIPREC rozšířeními

9.1 Úvod

Cílem této bakalářské práce je nahradit H.323 komunikaci na straně terminálu, představujícího v rámci SIPREC protokolu SRC (nahrávacího klienta). Před implementací SIPREC do aplikace by bylo vhodné otestovat základní scénář SIP signalizace se SIPREC rozšířeními pomocí nějakých externích nástrojů. Je to bezpečnější a lepší pro ladění. Pro tyto účely je možné použít program SIPp [7.3], který bude zastupovat SRC, a Drachtio Siprec Recording Server [7.5], který bude zastupovat SRS.



Obrázek 9.1: Architektura pro testování komunikace mezi SIPp a Drachtio

Cílem je vyzkoušet posílání SIP zpráv se SIPREC rozšířeními z počítače přes SIPp na Drachtio Siprec Recording Server, tj. nasimulovat základní komunikaci mezi SRC, který si přeje založit nahrávací relaci, a SRS. Až se to povede a SRS bude správně rozpoznávat přijaté zprávy a tím vytvářet nahrávací relaci založenou na protokolu SIPREC, bude možné implementovat tento scénář přímo do dispečerského řešení „KONOS“ v jazyce Java.

Pro tyto účely byl vytvořen XML scénář (viz A), podle kterého klient začíná komunikaci prostřednictvím následující zprávy `INVITE`:

```
INVITE sip:drachtio:[remote_port] SIP/2.0
Via: SIP/2.0/[transport] [local_ip]:[local_port];branch=[branch]
From: sipp <sip:sipp@[local_ip]:[local_port]>;
tag=[pid]SIPpTag09[call_number]
To: sip:drachtio:[remote_port]
Call-ID: [call_id]
CSeq: 1 INVITE
Contact: <sip:[local_ip]:[local_port]>;+sip.src
Max-Forwards: 70
Require: siprec
Content-Type: multipart/mixed; boundary=690DEB64
Content-Length: [len]

--690DEB64
Content-Type: application/sdp

v=0
o=user1 53655765 2353687637 IN IP[local_ip_type] [local_ip]
s=-
c=IN IP[local_ip_type] [local_ip]
t=0 0
m=audio [auto_media_port] RTP/AVP 8
c=IN IP[local_ip_type] [local_ip]
a=rtpmap:8 PCMA/8000
a=sendonly
a=label:1
m=audio 60000 RTP/AVP 8
c=IN IP[local_ip_type] [local_ip]
a=rtpmap:8 PCMA/8000
a=sendonly
a=label:2

--690DEB64
Content-Type: application/rs-metadata
Content-Disposition: recording-session

<?xml version="1.0" encoding="UTF-8"?>
  <recording xmlns='urn:ietf:params:xml:ns:recording:1'>
```

```
  <datamode>complete</datamode>
  <session session_id="XoM2pKdFR8C3eX33qJKcww==">
   <sipSessionID>hvij5wK4Wx</sipSessionID>
  </session>
  <participant participant_id="omFyWlFETEyFz/KXNLe4iQ==">
  <nameID aor="sip:6001@localhost"/>
  </participant>
  <participant participant_id="fLbZzLMgRXGp+F5w0hOt6w==">
   <nameID aor="sip:101@localhost"/>
  </participant>
  <stream stream_id="1LiAtnDVR9m469/0L4bwCA=="
          session_id="XoM2pKdFR8C3eX33qJKcww==">
  <label>1</label>
  </stream>
  <stream stream_id="WJgj5uxeRH+nwUah02dWqg=="
          session_id="XoM2pKdFR8C3eX33qJKcww==">
  <label>2</label>
  </stream>
 </recording>

--690DEB64--
```

Podle tohoto scénáře jsou do počáteční zprávy `INVITE`, kterou SRC odesílá SRS, přidány následující důležité SIPREC rozšíření:

- do pole hlavičky **Contact** je přiložen tag „`+sip.src`“ pro potvrzení, že dialog je vytvořen pro účely nahrávání;
- do pole hlavičky **Require** je přiložen tag „`siprec`“ pro to, aby uživatelské agenty, nepodporující SIPREC rozšíření, jednoduše odmítly požadavek `INVITE` s odpovědí `420 (Bad Extension)`;
- do SDP zprávy (`Content-Type: application/sdp`) je vložen atribut „`a=sendonly`“ pro okamžité nahrávání replikovaného média;
- jsou poskytnuty atributy „`a=label:1`“ a „`a=label:2`“, které identifikují zaznamenané mediální streamy a názvy, které obsahují, jsou mapovány na odkazy na tyto streamy v metadatech;
- hodnota pole hlavičky **Content-Disposition** v sekci „`application/rs-metadata+xml`“ je „`recording-session`“, což znamená, že sekce obsahuje metadata, která mají být zpracována v SRS.

Pokud všechny výše uvedené atributy jsou nastaveny správně, SRS, který tuto zprávu obdrží, by měl potvrdit založení nahrávací relace zprávou `200 OK`. SRC na ni odpoví zprávou `ACK`, pošle RTP stream (přehraje přes SIPp připravený PCAP soubor) a pak ukončí nahrávací relaci zprávou `BYE`.

9.2 Testování

Klientský scénář „`uac_siprec_pcap.xml`“, obsahující zprávu `INVITE`, uvedenou v předchozí sekci, byl spuštěn v programu SIPp (viz obrázek 9.2). Ten ji poslal na adresu běžícího nahrávacího serveru Drachtio 172.16.30.161:5060 (viz obrázek 9.3).

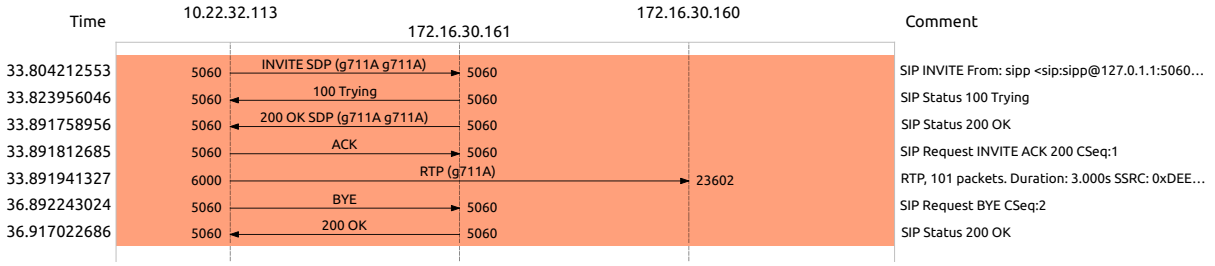
```
shchesof@shchesof-HP-ProBook-450-G7:~/Dokumenty/SIPREC/sipp-3.3$ sudo ./sipp -m 1 -sf uac_siprec_pcap.xml 172.16.30.161:5060
In pcap pcap/g711a.pcap, npkts 236
max pkt length 260
base port 2006
------------------------------ Scenario Screen -------- [1-9]: Change Screen --
  Call-rate(length)   Port   Total-time  Total-calls  Remote-host
  10.0(0 ms)/1.000s   5060       3.29 s            1  172.16.30.161:5060(UDP)

  Call limit reached (-m 1), 0.290 s period  1 ms scheduler resolution
  0 calls (limit 90)                  Peak was 1 calls, after 0 s
  0 Running, 3 Paused, 2 Woken up
  0 dead call msg (discarded)         0 out-of-call msg (discarded)
  1 open sockets
  102 Total RTP pckts sent            8.200 last period RTP rate (kB/s)

                             Messages  Retrans   Timeout   Unexpected-Msg
     INVITE ---------->         1         0         0
        100 <----------         1         0         0         0
        180 <----------         0         0         0         0
        200 <----------  E-RTD1 1         0         0         0

        ACK ---------->         1         0
            [ NOP ]
      Pause [   3000ms]         1                             0
        BYE ---------->         1         0         0
        200 <----------         1         0         0         0

------------------------------ Test Terminated --------------------------------
```

Obrázek 9.2: Úspěšná SIPREC komunikace ze strany SIPp klienta

```
shchesof@shchesof-HP-ProBook-450-G7:~/Dokumenty/SIPREC/drachtio-siprec-recording-server$ node app
{"level":30,"time":1614176504809,"msg":"attempting inbound connection","pid":134177,"hostname":"shchesof-HP-ProBook-450-G7","host":"172.16.30.161","port":9022,"secret":"cymru","v":1}
{"level":30,"time":1614176504812,"msg":"using rtpengine as the recorder","pid":134177,"hostname":"shchesof-HP-ProBook-450-G7","remote":{"host":"172.16.30.160","port":22222},"localPort":22223,"v":1}
{"level":30,"time":1614176504878,"msg":"inbound connection to drachtio listening on tcp/127.0.0.1:5060,udp/127.0.0.1:5060,udp/172.16.30.161:5060,tcp/172.16.30.161:5060","pid":134177,"hostname":"shchesof-HP-ProBook-450-G7","v":1}
{"level":30,"time":1614176509248,"msg":"received SIPREC invite: sip:drachtio:5060","pid":134177,"hostname":"shchesof-HP-ProBook-450-G7","callid":"1-134207@127.0.1.1","v":1}
{"level":30,"time":1614176509310,"msg":"call connected successfully, using rtpengine at {\"host\":\"172.16.30.160\",\"port\":22222}","pid":134177,"hostname":"shchesof-HP-ProBook-450-G7","callid":"1-134207@127.0.1.1","v":1}
{"level":30,"time":1614176512373,"msg":"call ended","pid":134177,"hostname":"shchesof-HP-ProBook-450-G7","callid":"1-134207@127.0.1.1","v":1}
```

Obrázek 9.3: Úspěšná SIPREC komunikace ze strany Drachtio serveru

Jejich komunikace byla zachycena pomocí Wiresharku a výsledek je vidět na obrázku 9.4. 10.22.32.113 je adresa SRC (t.j. SIPp, běžícího lokálně na počítači). 172.16.30.161 je adresa běžícího nahrávacího serveru Drachtio. 172.16.30.160 je adresa běžícího serveru „rtpengine“ (viz 7.5), který je zodpovědný za samotné nahrávání RTP paketů.



Obrázek 9.4: Nahrávací relace mezi SIPp a Drachtio

Podle [6] a [14] je to úspěšný SIPREC hovor. To znamená, že zpráva `INVITE`, která je posílána SRC, obsahuje správné údaje pro to, aby SRS rozpoznal probíhající relaci jako nahrávací a na základě toho nahrávání spustil. Následujícím cílem je implementovat generování a posílání takovéto zprávy v jazyce Java do dispečerského terminálu „KONOS“.

Kapitola 10

Návrh a implementace

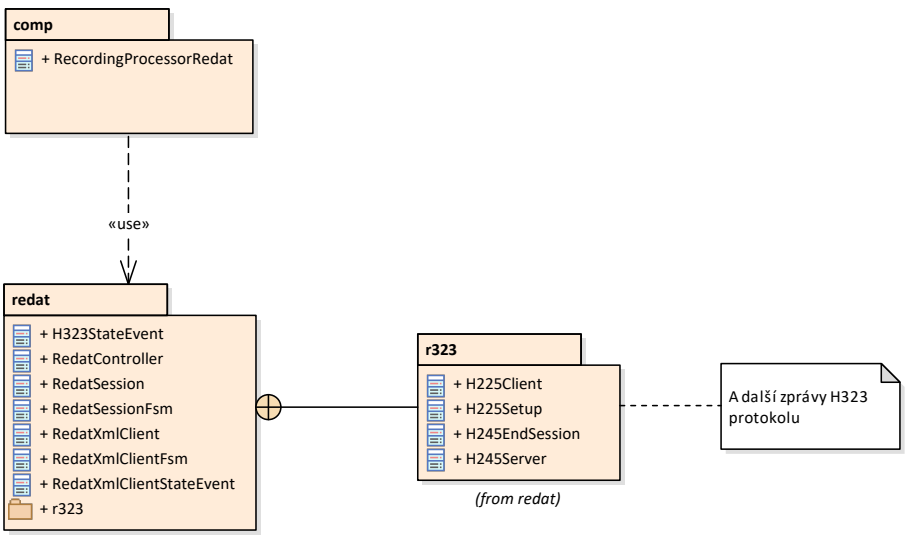
10.1 Úvod

V této kapitole bude popsán návrh a implementace podpory SIPREC protokolu do stávajícího řešení „KONOS“. Bude také uveden způsob a výsledky testování některých částí tohoto protokolu.

10.2 Architektura

Pro implementaci protokolu SIPREC bylo nutné změnit stávající architekturu řešení (viz obrázek 10.1), která počítá s existencí jen H.323 protokolu pro nahrávání hovorů.

10.2.1 Stávající architektura nahrávacího modulu



Obrázek 10.1: Stávající architektura nahrávacího modulu

10.2.1.1 comp

Modul `comp` obsahuje hlavní komponenty řešení, jednou z nichž je třída `RecordingProcessorRedat`. Tato třída používá modul `redat` za účelem správy nahrávání.

10.2.1.2 redat

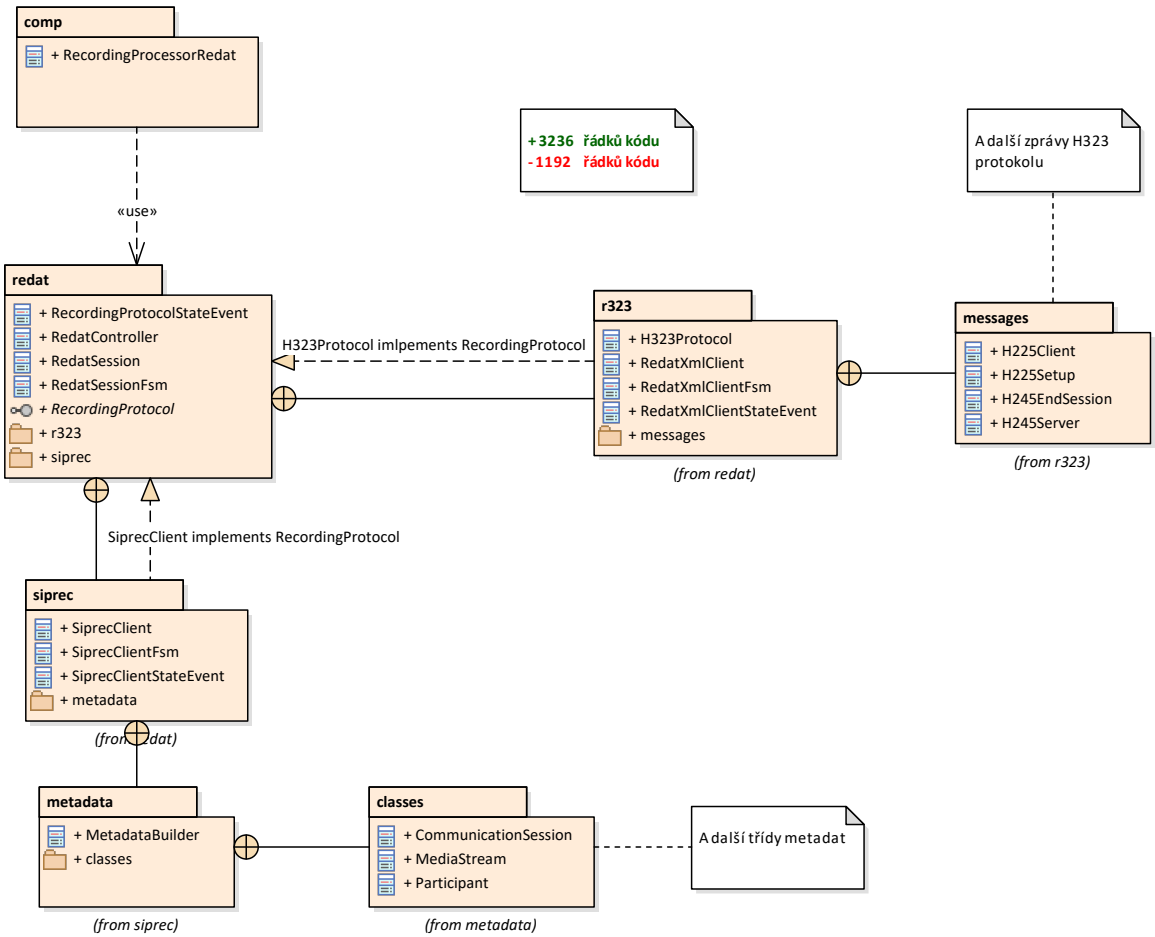
Modul `redat` je hlavním předmětem praktické části této bakalářské práce a většina změn se bude týkat právě tohoto modulu. Podle stávající architektury obsahuje veškeré třídy, které slouží k nahrávání hovorů (např. `RedatController` a `RedatSession`), událost `H323StateEvent`, obsahující informace o nahrávání a další balíček `r323`. Obsahuje také třídy, týkající se XML klienta `RedatXmlClient`, který je zodpovědný za navázání TCP spojení mezi terminálem a záznamovou jednotkou. Toto spojení je potřeba pro protokol H.323 (viz 8.3.2). `RedatXmlClientFsm` je stavový automat, zodpovědný za obsluhu přechodů mezi stavy klienta (viz 10.3). Události `RedatXmlClientStateEvent` jsou tímto stavovým automatem zpracované.

10.2.1.3 r323

Balíček `r323` obsahuje třídy, týkající se samotného H.323 protokolu. Třídy `H225Client` a `H225Server` spolu tento protokol tvoří (viz 8.3), třídy `H225Setup`, `H245EndSession` a několik dalších představují příslušné zprávy H.323 protokolu.

10.2.2 Navržená architektura nahrávacího modulu

Navržená architektura by měla být univerzálnější a dokázat flexibilně pracovat s několika nahrávacími protokoly.



Obrázek 10.2: Navržená architektura nahrávacího modulu

10.2.2.1 redat

V modulu `redat` jsou navrženy následující změny:

1. Zůstaly tu jenom třídy, které jsou společné pro nahrávání podle obou protokolů. Třídy, specifické pro daný protokol, jsou umístěny do příslušných balíčků `r323` a `siprec`.

2. Třída `H323StateEvent` byla přejmenována na `RecordingProtocolStateEvent`, protože zpracování nahrávacích událostí by nemělo záviset na konkrétně zvoleném nahrávacím protokolu.

3. Přibylo rozhraní `RecordingProtocol`, které by měl implementovat každý nahrávací protokol v projektu (viz obrázek 10.3). Rozhodnutí o tom, jaký je to přesně protokol (H.323 nebo SIPREC) by měla mít na starosti třída `RedatController` podle informací z konfigurace. Ostatní třídy by měly pracovat s protokoly jen přes navržený interface.

10.2.2.2 r323

V balíčku `r323` jsou navrženy následující změny:

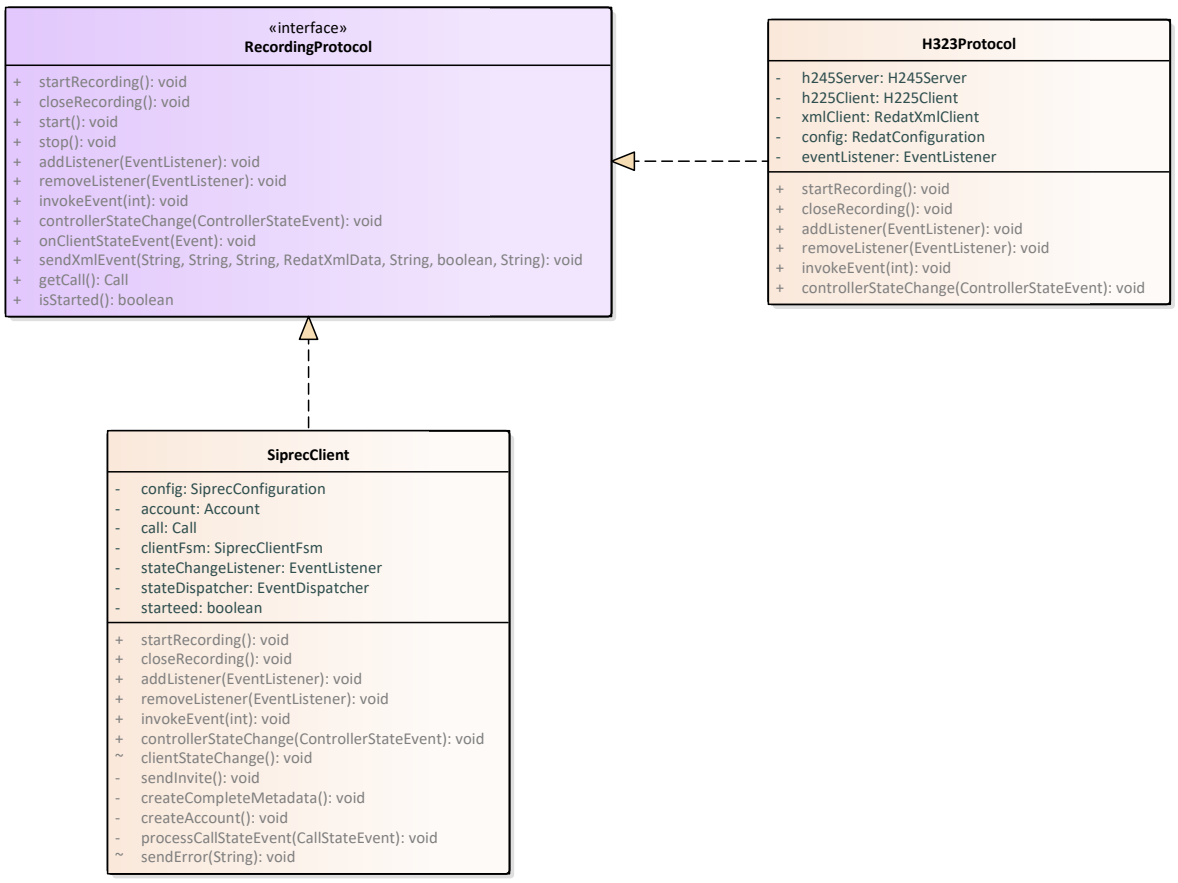
1. Je přidána třída `H323Protocol`, která implementuje rozhraní `RecordingProtocol` a je „sjednocením“ tříd `H245Server` a `H225Client` (viz obrázek 10.3). Metody serveru a klienta jsou teď volány jen touto třídou, nikoliv přímo z `RedatController` nebo `RedatSession`.

2. Třídy, které podle stávající architektury jsou v balíčku `r323` a jsou použity pro samotnou H.323 komunikaci, jsou přemístěny do balíčku `messages`.

10.2.2.3 siprec

Přibyl kompletně nový balíček `siprec`, ve kterém jsou třídy pro správu nahrávání podle SIPREC protokolu:

1. Třída `SiprecClient` je hlavní třídou SIPREC protokolu. Stejně jako třída `H323Protocol`, implementuje rozhraní `RecordingProtocol`.



Obrázek 10.3: RecordingProtocol, H323Protocol a SiprecClient

2. `SiprecClientFsm` je stavový automat, zodpovědný za obsluhu přechodů mezi stavy klienta (více viz 10.3). Tímto stavovým automatem jsou zpracovávány události `SiprecClientStateEvent`.

10.2.2.4 metadata

`metadata` je balíček, obsahující třídy pro vytváření SIPREC metadat podle [4]. Balíček `classes` obsahuje jednotlivé XML třídy. Více viz 10.8.

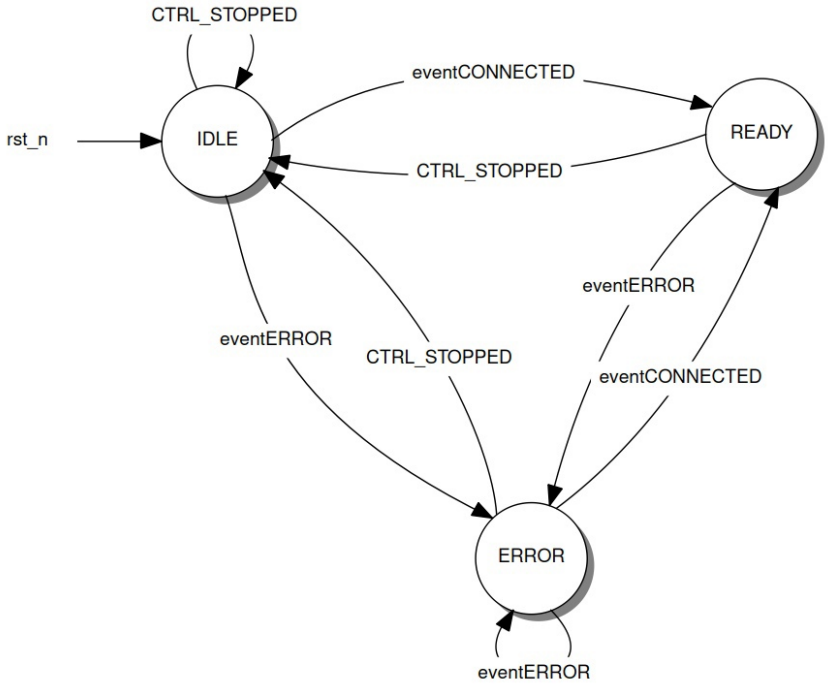
10.3 FSM

Konečný stavový automat (FSM) v programování je možné použít pro objekt, který vždy existuje právě v jednom z několika stavů a přechází mezi nimi podle pevně definovaných pravidel.

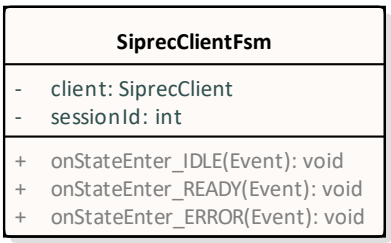
V dispečerském řešení „KONOS“ jsou konečné stavové automaty běžně používané pro různé objekty, například pro `RedatSession` existuje stavový automat `RedatSessionFsm`, pro `RedatXmlClient` - `RedatXmlClientFsm`, atd. Pro vytváření (kreslení) takových automatů existuje speciální nástroj Qfsm [7.6], který z nakresleného diagramu generuje výsledný XML soubor. Ten se pak pomocí interních nástrojů řešení „KONOS“ transformuje do javovských tříd.

Myšlenka stavového automatu pro objekt `SiprecClient` zní takto: je to SIP klient, který před začátkem nahrávání navazuje SIP spojení mezi sebou a nahrávacím serverem. Tento klient se bez ohledu na nahrávání nachází v jednom ze třech stavů:

- `IDLE` - klient je vytvořen, ale SIP spojení mezi ním a záznamovou jednotkou není navázáno. V tomto stavu má klient na starosti ukončení nahrávací relace, pokud existuje.
- `READY` - byla vytvořena SIP nahrávací relace mezi klientem a nahrávacím serverem a klient je připraven nahrávat komunikační relaci. V tomto stavu klient má poslat svým posluchačům `RecordingProtocolStateEvent`, který obsahuje následující informace: SIP hovor se nachází ve stavu `CONNECTED` a klient je připraven posílat RTP pakety příslušné komunikační relace na nahrávací server.
- `ERROR` - stala se chyba SIP spojení mezi nahrávacím klientem a serverem. V tomto stavu má klient informovat `RedatController` o chybě a ten by se s tím měl nějak vypořádat (odpovídající procesy již existují a nejsou předmětem této bakalářské práce).



Obrázek 10.4: SiprecClientFsm v aplikaci Qfsm

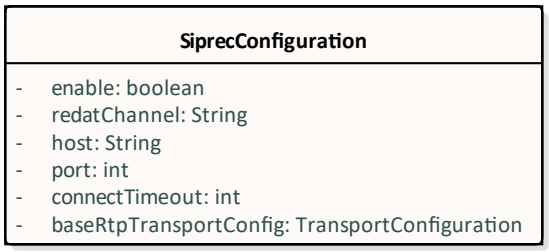


Obrázek 10.5: Třída SiprecClientFsm

Událost `CTRL_STOPPED` klient může obdržet od `RedatController`. Události `eventERROR` a `eventCONNECTED` jsou událostmi typu `CallStateEvent`, které klient může obdržet od probíhajícího hovoru. Posílání a přijímání událostí probíhá pomocí atributů třídy `stateDispatcher` a `stateChangedListener` (viz obrázek 10.3).

10.4 Konfigurace

Stejně jako existuje databázová tabulka `RedatConfiguration` (pro tyto účely se používá framework Hibernate 7.10), obsahující veškeré údaje potřebné pro nahrávání podle H.323 protokolu, byla vytvořena tabulka `SiprecConfiguration`:



Obrázek 10.6: Třída SiprecConfiguration

10.4.1 enable

Tato položka definuje, jestli protokol bude používán pro nahrávání.

10.4.2 redatChannel

Tato položka definuje, na který kanál záznamové jednotky budou hovory nahrávány.

10.4.3 host

Tato položka definuje adresu (typicky IP) záznamové jednotky „ReDat“.

10.4.4 port

Tato položka definuje port záznamové jednotky „ReDat“ pro SIP signalizaci (typicky je to port 5060).

10.4.5 connectTimeout

Tato položka definuje timeout pro navázání SIP spojení mezi nahrávacím klientem a serverem v milisekundách.

10.4.6 baseRtpTransportConfig

Tato položka definuje IP adresu a port záznamové jednotky, na které nahrávací klient bude posílat RTP streamy.

10.5 SIP

Tato sekce se věnuje krokům třídy `SiprecClient`, týkajícím se navázání SIP spojení a SIP signalizace. Všechny popsané metody a atributy jsou zobrazeny na obrázku 10.3.

10.5.1 Vytvoření SIPREC účtu

První věc, která se provede při vytvoření instance třídy `SiprecClient`, je vytvoření speciálního SIPREC účtu. Komunikace v dispečerském řešení „KONOS“ se řeší právě pomocí těchto účtů - vytvářejí hovory, posílají události, nesou v sobě důležité informace. V konstruktoru se volá privátní metoda `createAccount()`, ve které se vytváří SIPREC účet a provádějí se následující důležité kroky:

- Na instanci účtu se volá metoda `addCallStateListener()`, které se jako parameter předává `stateChangedListener` klienta pro zachycování událostí typu `CallStateEvent`, důležité pro `SipecClientFsm` (viz 10.3).
- Do **Contact** URI účtu se přidává tag „`;+sip.src`“, protože podle protokolu SIPREC představuje SRC (viz 6.3.1.1).

10.5.2 Založení nahrávací relace

Když si `RedatSession` přeje začít nahrávání, volá na instanci třídy `RecordingProtocol` metodu `startRecording()`. `SiprecClient` v této metodě volá privátní metodu `sendInvite()`. V ní vytvořený účet založí hovor prostřednictvím posílání zprávy `INVITE`, což je implementováno v knihovně PJSIP [7.1]. Ještě před založením hovoru se vytváří instance třídy `AdditionalSipMessageData`, která pomocí interních nástrojů řešení dovoluje provést následující důležité kroky:

- Do hlavičky zprávy `INVITE` se přidává pole „`Require`“ s hodnotou „`siprec`“ (viz 6.3.1.1).
- Voláním další privátní metody `createCompleteMetadata()` se vytvoří počáteční snapshot metadat hovoru (viz 10.2.2.4).

Nakonec se založí hovor, t.j. účet pošle knihovně PJSIP příslušný požadavek s vloženou instancí třídy `AdditionalSipMessageData`, obsahující potřebnou hlavičku a metadata.

10.5.3 Ukončení nahrávací relace

Když si `RedatSession` přeje ukončit nahrávání, volá na instanci třídy `RecordingProtocol` metodu `closeRecording()`. `SiprecClient` v této metodě ukončuje hovor posíláním zprávy `BYE` na nahrávací server (opět přes knihovnu PJSIP).

10.6 SDP

Tato sekce se věnuje krokům třídy `SiprecClient`, týkajícím se SDP části zprávy `INVITE`, které by měly být provedeny podle 6.4.1.1. Všechny popsané metody a atributy jsou zobrazeny na obrázku 10.3.

- Pokud nedostane žádná média od protistrany, knihovna PJSIP automaticky nastaví každý mediální stream v SDP nabídce pouze pro odesílání nastavením atributu „`a=sendonly`“.
- Knihovna PJSIP poskytuje projektu v jazyce Java metodu `setParametres()`, která musí být aplikována na instanci třídy `Call`. Mezi parametry této metody patří i `label`, jehož hodnota se přiřadí příslušnému atributu v SDP nabídce. V třídě `SiprecClient` se tato metoda volá v privátní metodě `sendInvite()`. Jelikož dispečerské řešení „KONOS“ pro nahrávání hovoru používá sloučený RTP stream 5.2.2.3, nastavuje se tento label pro celý hovor jen jednou.

Výsledkem kroků v této a předchozí sekci je zpráva `INVITE` z kapitoly 9, pomocí které se zahajuje SIPREC komunikace mezi terminálem a záznamovou jendotkou „ReDat“. Žádné další modifikace zprávy pro standardní scénář nejsou potřeba.

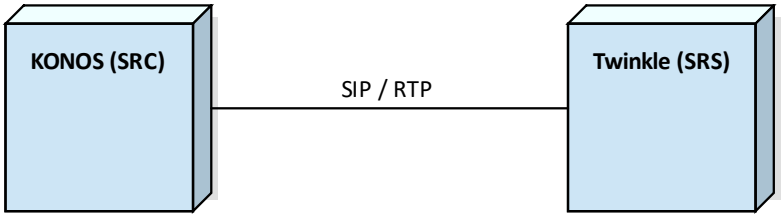
10.7 RTP

Po úspěšné výměně SIP zpráv mezi nahrávacím klientem a serverem by mělo začít posílání RTP paketů, tj. skutečné nahrávání. V dispečerském řešení „KONOS“ se to řeší slučováním mediálních streamů (viz 5.2.2.3).

Už bylo zmíněno, že `RedatSession` také má svůj FSM. Když se nachází ve stavu `STARTING` a obdrží od objektu `RecordingProtocol` událost signalizující, že je připraven provádět nahrávání, přechází do stavu `RECORDING`. Během tohoto přechodu začíná posílání RTP paketů třídou `RecordingProcessorRedat`. Pokud navržený `SiprecClient` a jeho FSM byly implementovány správně, `RedatSession` by se měla dostat do stavu `RECORDING` stejně jako při použití H.323 protokolu.

10.7.1 Testování

Před nahráváním na skutečný „ReDat“ server by bylo vhodné zkontrolovat, jestli se RTP pakety v případě použití SIPREC protokolu posílají správně. Pro tento účel lze použít aplikaci Twinkle [7.8] a architektura pro testování vypadá následovně:

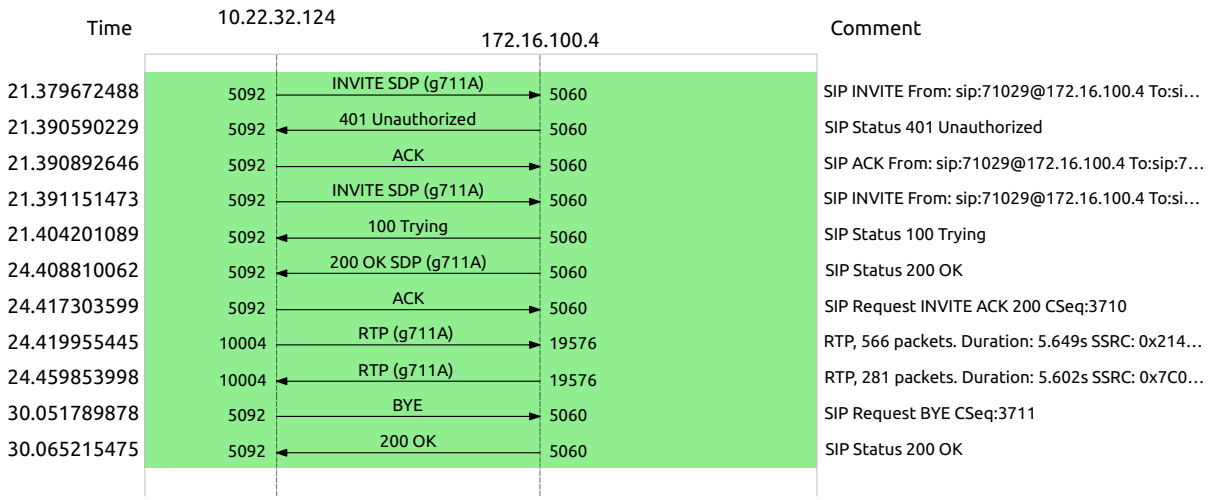


Obrázek 10.7: Architektura pro testování RTP komunikace mezi KONOS a Twinkle

Myšlenka tohoto testu zní takto: z dispečerského terminálu „KONOS“ proběhne volání na číslo s automatickým odpovídačem z testovací infrastruktury firmy „TTC Marconi“, které v porovnání s příkladem z kapitoly 8 navíc přehrává hudbu, tj. RTP pakety chodí obousměrně. Současně s tímto hovorem objekt `SiprecClient` sestaví další SIP hovor na nahrávací server, což v tomto případě bude aplikace Twinkle. Ten SIPREC samozřejmě neumí, takže příslušné rozšíření jednoduše zahodí, ale hovor přijme, a měl by dostat RTP stream, který je sloučen z RTP streamů, posílaných terminálem na číslo přehrávající hudbu a opačným směrem. Pokud Wiresharkem bude zachyceno, že aplikace Twinkle RTP stream dostala a navíc přes ni bude slyšet hovor ze strany terminálu a hudbu ze strany čísla s automatickým odpovídačem, pak přenos RTP podle SIPREC protokolu je implementován správně.

Výsledky popsané komunikace jsou zobrazeny a popsány v následujících sekcích.

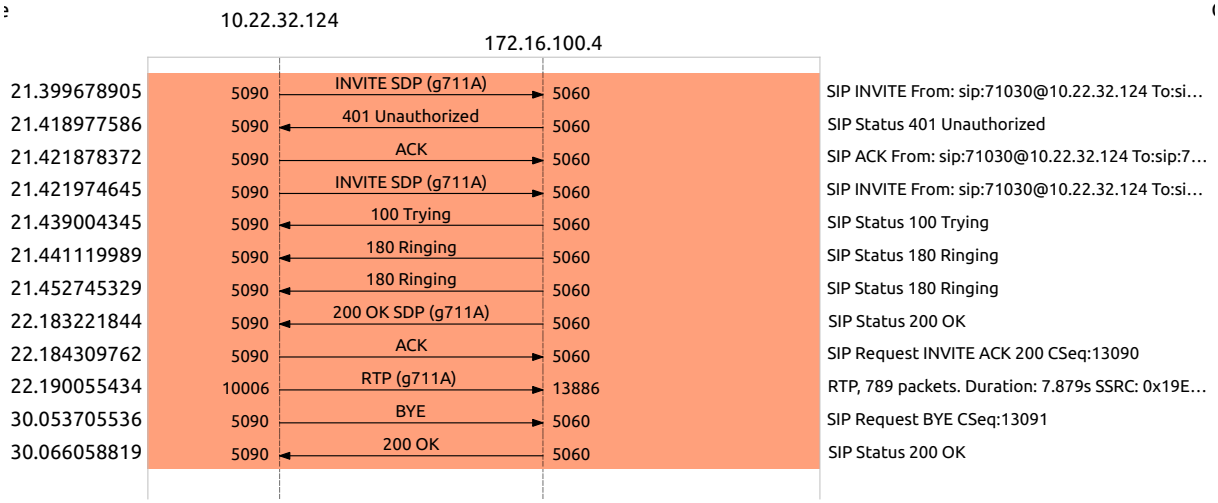
10.7.1.1 SIP hovor mezi terminálem a číslem s automatickým odpovídačem



Obrázek 10.8: Komunikační relace mezi terminálem a číslem přehrávajícím hudbu

Je to typický SIP hovor, ve kterém RTP chodí obousměrně: jedna strana mluví do terminálu a protistrana - číslo s automatickým odpovídačem - přehrává hudbu. Tento hovor bude předmětem nahrávání na softphone Twinkle, měl by být tedy celý přes něj slyšet.

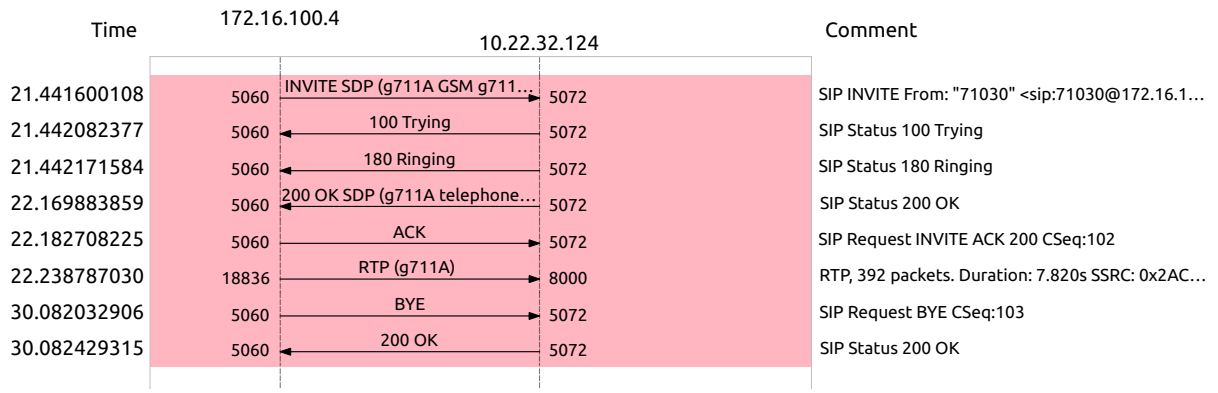
10.7.1.2 SIPREC hovor mezi terminálem a telefonní ústřednou Asterisk



Obrázek 10.9: Nahrávací relace mezi terminálem a Asteriskem

V dispečerském řešení „KONOS“ jsou určité účty a hovory řešeny přes ústřednu Asterisk [7.9], která je SIP registrátorem. Tím pádem hovor mezi účty, které mají registraci, probíhá přes tuto ústřednu. Takový přechod v případě skutečné záznamové jednotky „ReDat“ existovat nebude, je vyžadován jen pro to, že pro testování je používán Twinkle, který má registrovaný účet. Ten hovor již probíhá podle protokolu SIPREC - nahrávací klient posílá jednosměrně RTP pakety příslušné komunikační relace na Asterisk, který je pak předá softphonu.

10.7.1.3 SIPREC hovor mezi Asteriskem a softphonem Twinkle



Obrázek 10.10: Nahrávací relace mezi Asteriskem a Twinkle

Tento hovor je v podstatě stejný, jako v předchozí sekci, jen v tomto případě Asterisk posílá RTP pakety získané od terminálu softphonu Twinkle. Probíhající hovor z sekce 10.7.1.1 je slyšet přes Twinkle a to v správném pořadí paketů, takže RTP přenos je v nově přidaných třídách implementován správně.

10.8 Metadata

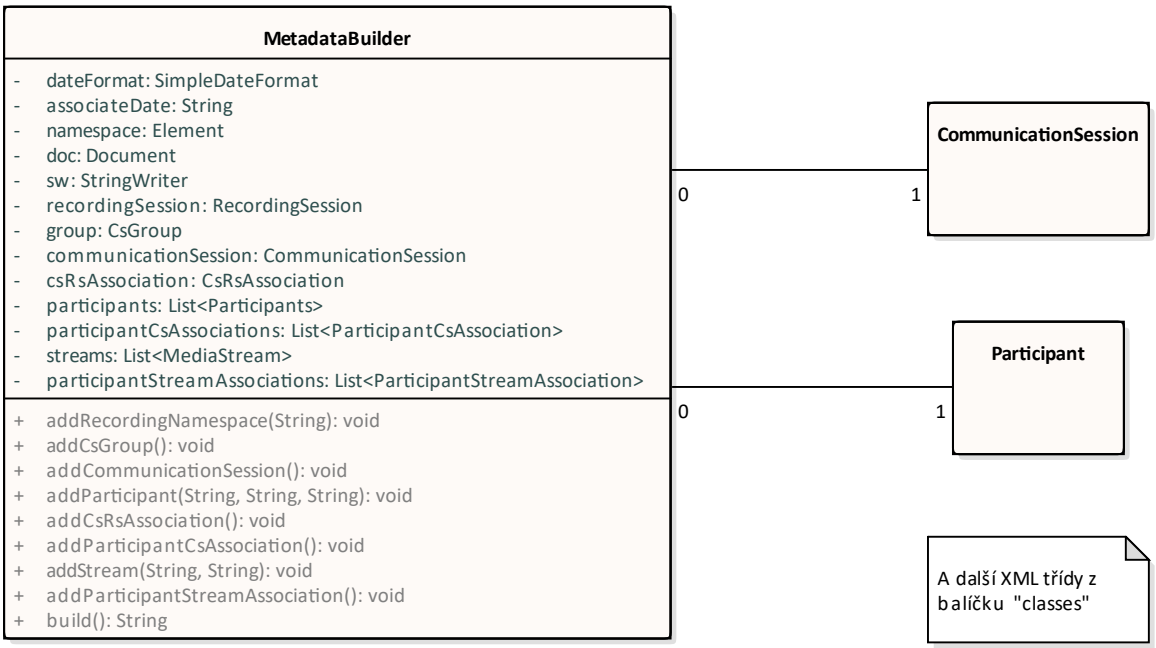
Tato sekce se věnuje krokům třídy `SiprecClient`, které se týkají posílání metadat komunikační relace v rámci nahrávací relace. Všechny popsané metody a atributy jsou zobrazeny na obrázku 10.3.

10.8.1 Hlavička metadat

Pomocí knihovny PJSIP byl do řešení přidán další konstruktor objektu `AdditionalSipMessageData`. Přijímá teď dodatečný parametr, který nastavuje hodnoty pole hlavičky `Content-Disposition` podle 6.5.

10.8.2 Tělo metadat

`SiprecClient` používá v privátní metodě `createCompleteMetadata()` nově přidanou třídu `MetadataBuilder`, která dovoluje postupně sestavit potřebné části metadat.



Obrázek 10.11: Třída MetadataBuilder

- Atributy `dateFormat` a `associateDate` nesou v sobě informace o datu a času komunikační relace.
- Atributy `namespace`, `doc` a `sw` slouží k sestavení výsledného XML souboru podle [18].
- Atributy `recordingSession`, `group` a všechny další jsou instancemi tříd z balíčku "classes", které představují jednotlivé XML třídy podle [4].
- Metody, jejichž název začíná klíčovým slovem `add`, jsou použity pro přidání příslušných částí do metadat.
- Metoda `build()` vrací konečný výsledek ve formátu `String`, který bude klientem přidán do výsledné zprávy `INVITE`.

Kapitola 11

Porovnání

Tato kapitola obsahuje porovnání možností stávajícího a navrženého řešení z hlediska architektury a integrace. Budou také shrnuty výhody a nevýhody protokolů H.323 a SIPREC.

11.1 Architektura

V předchozí kapitole v sekci 10.2 byly navrženy kroky ke zlepšení stávající architektury. Jsou založeny na tom, že stávající architektura počítá jen s implementací protokolu H.323 a neexistuje možnost flexibilně přidávat další protokoly a jednoduše mezi nimi přepínat. Navržená architektura to již však dovoluje, což umožňuje integraci s většinou dostupných nahrávacích systémů a zvyšuje potenciál celého řešení.

Zavedení nového interfacu `RecordingProtocol` má za výsledek i to, že řídící komponenty projektu už nepotřebují znát, s kterou implementací nahrávacího protokolu pracují. Toto je v souladu s jedním z hlavních konceptů objektově orientovaného programování - zapouzdření. Takový kód je snadno pochopitelný a znovu použitelný, což je velmi důležité při práci ve vývojářském týmu spolu s dalšími programátory.

11.2 Protokoly

Co se týká samotných protokolů pro nahrávání hlasové komunikace přes internet, byla navržena implementace protokolu SIPREC. Ten by měl v budoucnu nahradit protokol H.323, který zatím ale má v řešení zůstat z důvodu zpětné kompatibility.

SIPREC protokol je rozšířený SIP protokol, takže by bylo korektnější H.323 porovnávat s druhým z uvedených. Mezi těmito protokoly existují následující důležité rozdíly:

1. H.323 je založen na klasické telefonii, zatímco SIP je založen na internetu. Jelikož

dispečerský terminál „KONOS“ používá internetovou hlasovou komunikaci, měl by používat i odpovídající protokol.

2. H.323 je navržen ITU (International Telecommunication Union), zatímco SIP je navržen IETF (Internet Engineering Task Force). Toto jsou dva různé zdroje standardů pro komunikační protokoly a IETF je považován za primární standard [19].

3. H.323 pro zprávy používá binární formát. Naopak SIP používá formát ASCII. Každý z těchto formátů má svoje výhody a nevýhody. Lidsky čitelný text (ASCII) je pomalejší pro kódování a dekódování než binární, ale je snadnější pro práci a ladění.

4. Architektura H.323 je monolitická, SIP je postaven na modulární architektuře.

5. SIP nabízí možnost posílání rychlých (instantních) zpráv. Naopak v H.323 taková možnost neexistuje. V dispečerském řešení „KONOS“ jsou instantní SIP zprávy často používány při komunikaci, takže je vhodné mít tuto možnost i při nahrávání hovorů.

6. SIP je škálovatelný, flexibilní a snadno implementovatelný do nových aplikací. Toto však neplatí o H.323 z důvodu jeho komplexnosti.

Z tohoto porovnání plyne, že pro projekt „KONOS“ protokol SIP a jeho rozšíření pro nahrávání hovorů SIPREC je lepším řešením. H.323 je telefonní průmyslový standard, který je obvykle považován za složitý. Není flexibilní a těžko se přizpůsobuje novým aplikacím. Naopak SIP je standardní internetový protokol, který je vysoce modulární, jednoduchý a flexibilní. Jeho nevýhodou je, že neumí spolupracovat s telefonickými protokoly, ale dobře spolupracuje s dalšími internetovými protokoly, ale nevýhodou je, že neumí spolupracovat s telefonickými protokoly. [20]

Kapitola 12

Závěr

Cílem této bakalářské práce bylo navrhnout a implementovat klientskou část protokolu SIPREC pro nahrávání hovorů do dispečerského terminálu „KONOS“ a tím nahradit stávající řešení podle H.323 protokolu se zachováním zpětné kompatibility. Před započetím práce byl prostudován velký objem potřebné teorie týkající se protokolů SIP, RTP, SDP, SIPREC a H.323.

V praktické části práce bylo prozkoumáno stávající řešení nahrávání hovorů a navržena architektura počítající s existencí několika nahrávacích protokolů. Ta by dovolila v budoucnu přidávat nové protokoly s minimálním zásahem do již existující implementace. Poté byly implementovány jednotlivé části SIPREC protokolu (SIP signalizace, přenos RTP paketů, potřebná databázová konfigurace, stavový automat, vytvoření metadat komunikační relace) a otestovány pomocí nejrůznějších nástrojů, open source projektů, unit testů a testovací infrastruktury společnosti „TTC Marconi“.

Jelikož implementace kvůli technickým problémům na straně společnosti „Retia“ nemohla být okamžitě otestována proti jejímu nahrávacímu serveru „ReDat“, bylo třeba najít vhodné náhradní řešení. Z tohoto důvodu vnikla potřeba využít zmíněné nástroje a open source projekty pro otestování jednotlivých kroků implementace, což nakonec přispělo k zajímavějšímu a přínosnějšímu obsahu práce.

Výsledkem této bakalářské práce je tedy fungující prototyp klienta nahrávajícího hovory podle protokolu SIPREC. Ten zatím podporuje jenom standardní scénář, tj. nahrávání hovoru mezi dvěma účastníky bez dodatečných funkcí jako jsou pozastavení a obnovení komunikační relace, přidání dalšího účastníka do skupiny, atd. V implementaci těchto a dalších scénářů budu pokračovat dál v rámci práce ve vývojářském týmu společnosti „TTC Marconi“.

Příloha A

Obsah elektronické přílohy

- **src/** - složka obsahující zdrojový kód nahrávacího modulu dispečerského řešení „KONOS“.
- **uac_siprec_pcap.xml** - XML scénář pro testování SIP signalizace se SIPREC rozšířeními přes aplikaci SIPp.

Seznam zdrojů

[1] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, E. Schooler., *SIP: Session Initiation Protocol*, 6.2002, RFC 3261.

[2] K. Rehor, L. Portman, A. Hutton, R. Jain, *Use Cases and Requirements for SIP-Based Media Recording (SIPREC)*, 8.2011, RFC 6341.

[3] K. Rehor, L. Portman, A. Hutton, R. Jain, *An Architecture for Media Recording Using the Session Initiation Protocol*, 5.2014, RFC 7245.

[4] R. Ravindranath, P. Ravindran, P. Kyzivat, *Session Initiation Protocol (SIP) Recording Metadata*, 5.2016, RFC 7865.

[5] L. Portman, H. Lum, C. Eckel, A. Johnston, A. Hutton, *Session Recording Protocol*, 5.2016, RFC 7866.

[6] R. Ravindranath, P. Ravindran, P. Kyzivat, *Session Initiation Protocol (SIP) Recording Call Flows*, 2.2017, RFC 8068.

[7] Teluu Ltd., *PJSIP*, https://www.pjsip.org/, [online].

[8] A. Orebaugh, G. Ramirez, J. Beale, *Wireshark & Ethereal Network Protocol Analyzer Toolkit*, 2.2007.

[9] R. Gayraud, O. Jacques, *SIPp*, http://sipp.sourceforge.net/, [online].

[10] QNX, *Phindows*, 2010, http://www.qnx.com/developers/docs/6.5.0/index.jsp?topic=%2Fcom.qnx.doc.phindows%2Ftopic%2Fcon_getstarted.html, [online].

[11] D. Horton, *Drachtio Siprec Recording Server*, https://github.com/drachtio/drachtio-siprec-recording-server, [online].

[12] S. Duffner, R. Strobel, *Qfsm*, http://qfsm.sourceforge.net/, [online].

[13] Paul E. Jones, *H.323 Protocol Overview*, 10.2007.

[14] Oracle, *SIPREC Call Flows*, 2019, https://docs.oracle.com/cd/E95618_01/html/sbc_scz810_monitoring/GUID-BA6A47DB-7691-4709-9E5A-6D2AED5ABD91.htm#GUID-BA6A47DB-7691-4709-9E5A-6D2AED5ABD91, [online].

[15] TTC MARCONI s.r.o., *Dispečerský systém KONOS*, 2020, https://ttc-marconi.com/produkty-a-sluzby/dispecerske-reseni-pro-krizovou-komunikaci-dispecersky-system-konos/, [online].

[16] C. Bauer, G. King, *Java Persistence with Hibernate*, 10.2006.

[17] Russell Bryant, *Asterisk*, 2018, https://wiki.asterisk.org/wiki/display/AST/Home/, [online].

[18] Oracle, *Interface Document*, 2020, https://docs.oracle.com/javase/7/docs/api/org/w3c/dom/Document.html, [online].

[19] S. Trowbridge, E. Lear, G. Fishman, S. Bradner, *Internet Engineering Task Force and International Telecommunication Union - Telecommunication Standardization Sector Collaboration Guidelines*, 9.2012, RFC 6756.

[20] H. Schulzrinne, J. Rosenberg, *A Comparison of SIP and H.323 for Internet Telephony*, 7.1998.