SIP CLIENT IN JAVA PROGRAMMING LANGUAGE

Martin Zukal
Bachelor Degree Programme (1), FEEC BUT
E-mail: xzukal02@stud.feec.vutbr.cz

Supervised by: Petr Číka
E-mail: cika@feec.vutbr.cz

ABSTRACT

This contribution deals with the idea why to create a SIP client and how to implement it in Java programming language. The first part describes the background of a voice transfer over the Internet with emphasis on the Session Initiation Protocol. Necessary libraries which will be used in the application are described in the next part. Other parts deal with the design and implementation of the SIP client and its utilization in a real application.

1 ÚVOD

Pro přenos hlasu přes Internet existuje několik možností. I přesto, že se jedná většinou o zcela odlišná řešení, jejich společným jmenovatelem je oddělení signalizace hovoru od přenosu samotných multimediálních dat. Signalizace je řešena jednotlivými architekturami různě, ale pro přenos multimediálních dat je téměř vždy použit protokol RTP (Real-time Transport Protocol). V dnešní době jsou velice populární řešení založená na protokolu SIP (Session Initiation Protocol). Současná verze tohoto protokolu je standardizována v RFC 3261 [1]. SIP je textově orientovaný, signalizační protokol podporující přítomnost uživatelů (presence) a posílání instantních zpráv (instant messaging).
I přesto, že SIP je poměrně mladý protokol, klientů, kteří jej podporují, je celá řada. Ne všechny však pracují přesně v souladu s doporučením RFC 3261 a jen málo z nich je multiplatformních. Vzhledem k uvedeným skutečnostem vznikla myšlenka vytvořit korektně pracujícího SIP klienta spustitelného na jakékoli platformě. K tomu je velmi vhodný jazyk Java, který je navržen tak, aby pracoval na jakékoli dnes používané platformě, a navíc umožňuje vytvořit jak desktopovou aplikaci, tak i aplikaci ve formě apletu spustitelného ve webovém prohlížeči.

2 SPECIFIKACE POŽADAVKŮ

Vyvíjená aplikace bude schopna provádět základní obsluhu hovorů s využitím protokolu SIP a bude umožňovat zabezpečenou autentizaci na serveru. Dále bude schopna pracovat na více platformách, s čímž souvisí i možnost spouštění aplikace z internetového prohlížeče v podobě apletu. Výhodou tohoto řešení je fakt, že aplikace nevyžaduje instalaci a je přístupná odkudkoli (kde je umožněn přístup na Internet).

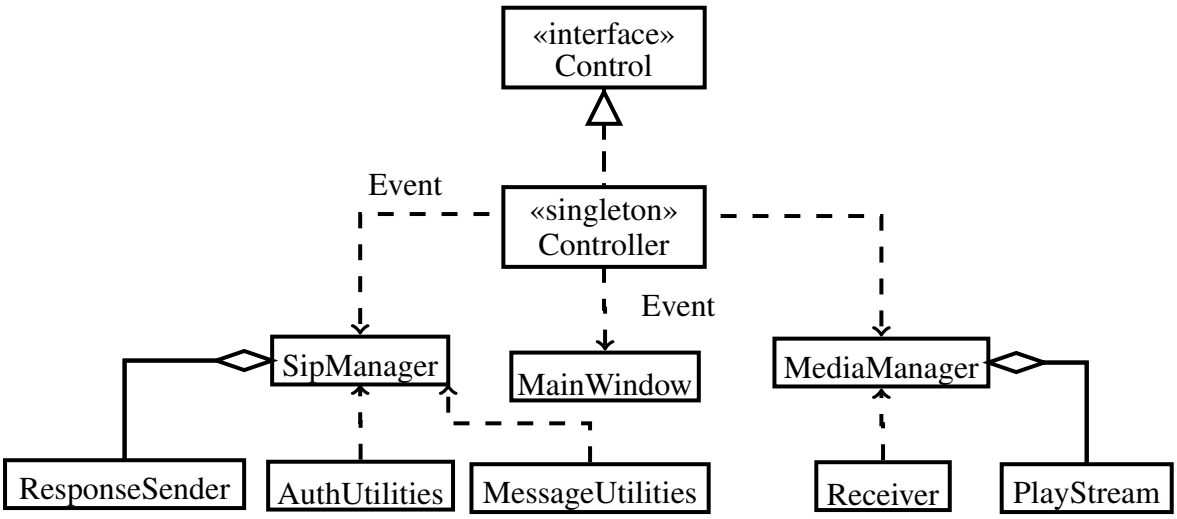
3 ŘEŠENÍ

3.1 VYUŽITÍ KNIHOVEN

Z důvodu složitosti a komplexnosti řešené úlohy bylo v aplikaci využito již existujících knihoven. Tyto knihovny řeší příjem a odesílání SIP zpráv, práci se zvukem a jeho následný přenos přes Internet.
Pro jazyk Java existují API (Application Programming Interface) s názvem JAIN (Java APIs for Integrated Networks), které obsahují rozhraní pro práci s protokolem SIP. Jednou z mála implementací JAIN SIP API je referenční implementace, která pochází z USA, konrétně za ní stojí National Institute of Standards and Technology (NIST). Tato knihovna byla z dostupných možností vyhodnocena jako nejlépe vyhovující pro vyvíjenou aplikaci. Jejím úkolem je zajistit veškeré funkce týkající se signalizace.
Kromě signalizace bylo nutné v aplikaci vyřešit práci s audiem (jeho kódování do podoby vhodné pro přenos přes Internet a samotné vysílání těchto dat pomocí protokolu RTP). Stejně tak bylo nutné zajistit příjem zvukových dat v podobě RTP streamu jejich dekódvání a nakonec přehrávání. Pro práci s audiem bylo využito Java Media Framework [2].

3.2 ARCHITEKTURA

Návrh vychází z návrhového vzoru Model View Controller (MVC) [3], který byl mírně upraven tak, aby aplikace nebyla přiliš složitá a zároveň zahrnovala výhody, které tento návrhový vzor nabízí. Největším přínosem návrhového vzoru MVC je bezpochyby oddělení vzhledu od aplikační logiky. Návrh byl proto proveden s tímto požadavkem. Obrázek 1 ukazuje class diagram aplikace. Rozhraní Control obsahuje pouze dvě metody: processEvent (zpracovává události



Obrázek 1: Class diagram aplikace

vzniklé v aplikaci) a processException (zpracovává výjimky vzniklé v aplikaci). Třída Controller implementuje rozhraní control a je srdcem celé aplikace. Veškerá komunikace mezi jednotlivými částmi aplikace probíhá právě přes tuto třídu. Třída Controller komunikuje s ostatními částmi aplikace buď prostřenictvím událostí (Events) nebo volá metody příslušných tříd. Každá událost implementuje rozhraní Event, což umožňuje metodě processEvent přijmout jakýkoli

typ události. Rozhraní Event obsahuje dvě metody: getInfoMessage pro získání popisu vzniklé události a getSource pro získání odkazu na zdroj události.
Ostatní třídy budou popsány jen velmi stručně.

- SipManager – implementuje rozhraní SipListener z knihovny JAIN SIP (jeho metody jsou volány při příchodu zpráv). Obsahuje aplikační logiku reagující na příchozí zprávy dle doporučení RFC 3261 [1]. V případě, že je vyžadována akce od uživatele, je poslána odpovídající událost třídě Controller.
- ResponseSender – vnitřní třída, která vytváří a odesílá odpovědi na příchozí žádosti.
- MessageUtilities – obsahuje metody pro tvorbu jednotlivých částí SIP zpráv. K tomu využívá tovární třídy knihovny JAIN SIP pro tvorbu jednotlivých polí SIP zpráv.
- AuthUtilities – sdružuje metody využívané při autentizaci uživatelů na serveru.
- MainWindow – zahrnuje všechny prvky grafického rozhraní, které se dynamicky mění podle aktuálního stavu aplikace.
- MediaManager – spravuje veškeré funkce týkající se multimediální relace [2].
- PlayStream – vnitřní třída zajišťující identifikaci příchozího RTP streamu a jeho přehrávání.
- Sender – kóduje a odesílá data v podobě RTP streamu na určenou adresu.

4 ZÁVĚR

Podle popsaného návrhu byla provedena implementace aplikace a proběhlo testování základních funkcí (jednoduché spojení s přenosem RTP streamu). V dalších testech bylo ověřeno správné chování aplikace při spouštění z prohlížeče a funkčnost autentizačních mechanizmů. Aplikace ve všech testech obstála.
Pro plné využití této práce se předpokládá napojení na aplikaci Webmail firmy IceWarp. Napojení bude provedeno tak, že při volbě `SIP call` z kontextové nabídky u kontaktu se spustí aplikace ve formě apletu s parametry: `volající` (uživatelské jméno a heslo) a `volaný` (uživatelské jméno). Okamžitě po spuštění naváže aplikace spojení s druhým klientem a zprostředkuje tak hovor mezi dvěma uživateli.

REFERENCE

[1] ROSENBERG, Jonathan, et al. *SIP: Session Initiation Protocol* [online]. 2002 [cit. 2008-02-17]. Dostupný z WWW: <http://tools.ietf.org/html/rfc3261>.

[2] Sun Microsystems, Inc.. *Java Media Framework API (JMF)* [online]. [2003] [cit. 2008-02-17]. Dostupný z WWW: <http://java.sun.com/products/java-media/jmf/>.

[3] DEACON, John. *Model-View-Controller (MVC) Architecture* [online]. August 1995, revised August 2000 and April 2005 [cit. 2008-02-17]. Dostupný z WWW: <http://www.jdl.co.uk/briefings/mvc.html>.