UNIVERSITÁ DEGLI STUDI DI PADOVA
DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE
CORSO DI LAUREA TRIENNALE IN INGEGNERIA INFORMATICA

PARIVOIP: IL PROTOCOLLO SIP

LAUREANDO: FEDERICO SCOZZAI

RELATORE: Prof. Enoch Peserico Stecchini Negri De Slavi

CORRELATORE: Dott. Paolo Bertasi

A.A 2009-2010

Alla mia famiglia

che mi ha sempre sostenuto

Sommario

Negli ultimi anni si sta osservando un cambiamento nel modo in cui le persone guardano a internet e ai suoi contenuti. Alla fine degli anni ô90 la rete internet era ancora giovane e non del tutto matura. Il traffico mondiale era dominato in larga percentuale dallo scambio di file, dalla consultazione di contenuti statici e dallôinvio delle prime mail. Oggi invece, anche grazie allôestendersi della banda larga, le persone tendono a partecipare in modo attivo a internet, contribuiscono e popolano il web con propri contenuti. E qual è il modo più diretto di comunicare con unôaltra persona, se non quello di parlare e vedersi in tempo reale? Questo oggi è possibile gratuitamente tramite la tecnologia VoIP.

Indice

Introduzione:

Si può immaginare che le reti telefoniche si siano evolute negli ultimi 100 anni. In parte si. Se si guarda a come funziona una rete telefonica, non è cambiato molto. Le connessioni sono ancora fatte per mezzo di circuiti "dedicati" collegati punto a punto per la durata di una chiamata. Gli operatori del centralino sono stati sostituiti dal un sistema digitale di commutazione e il concetto di collegamento dei circuiti non si è evoluto di molto.

Il telefono è sempre lo stesso strumento, costituito da un altoparlante e un microfono. Le reti sono ancora gerarchiche reti di commutazione, anche se alcuni livelli intermedi sono stati soppressi nel corso degli anni. Le chiamate sono ancora passate attraverso percorsi dedicati in base alle cifre del selettore a disco o del tastierino numerico più recentemente. Tutto questo è ben lontano dalle reti a pacchetto.

Quando è arrivato Internet, tutto è cambiato. Improvvisamente abbiamo tutti avuto accesso ad una gigantesca rete pubblica, che ci ha permesso di connetterci e scambiare tutti i tipi di dati con chiunque avesse una connessione. Nuovi fornitori di servizi Internet (ISP) hanno iniziato con offerte di posta elettronica, e poi con servizi di news, bacheche e altro ancora. Questi ISP, continuano a crescere e ad offrire nuovi servizi di pari passo all'evoluzione di Internet.

Quando il World Wide Web (WWW) è stato introdotto, il modo di comunicare è stato modificato per sempre. All'improvviso possiamo comprare online, creare la nostra presenza sui ñsocial networkò e socializzare con persone provenienti da tutto il mondo.

Era naturale che il grande servizio successivo sarebbe stato il Voice over IP (VoIP).

Oggi possiamo finalmente goderci le molte caratteristiche e funzionalità che consentono (con le reti a pacchetto) il servizio combinato di telefonia con la nostra e-mail e servizi di dati.

Per molti anni è stato come il far west, con molte diverse implementazioni e molte configurazioni diverse a sostegno delle reti VoIP.

C'era chi cercava di rendere la rete a pacchetto emulativa di una rete commutata, e chi cercava di far passare la voce sulla rete a pacchetto.

Alla fine, si è trovata una via di mezzo. Sono stati sviluppati nuovi protocolli per contribuire e rendere il VoIP più robusto e affidabile, cercando di mantenere la qualità eguale o superiore alla vecchia rete. Si vedono reti che cominciano a maturare e

standardizzare le loro tecnologie in modo che possano interoperare con altre reti e apparecchi di altri fornitori.

E ora, il Session Initiation Protocol (SIP), emerge dalla polvere, come il vincitore per il controllo della sessione. Il fatto che la 3rd Generation Partnership Project (3GPP) abbia definito SIP come standard per il controllo della sessione di chiamata per la loro IP Multimedia Subsystem (IMS), non è da meno. Ad oggi vi è in corso un lavoro importante per aggiungere funzionalità e aumentare le prestazioni del protocollo SIP.

Capitolo Primo

1 Chi stabilisce lo standard voip?

Le reti VoIP combinano il meglio delle tecnologie di comunicazione voce e dati. Ma spesso questa combinazione genera delle sfide non indifferenti quando lôindustria cerca di unire in una singola tecnologia il meglio dei circuiti integrati con il meglio della commutazione a pacchetti.

Forse la più grande sfida arriva nellôarea dellôinteroperabilità ï ovvero il concetto secondo il quale si consente allôhardware e al software di due diversi fornitori di essere integrati e operare in un unico sistema. Ma dal momento che i fornitori di solito hanno un approccio competitivo, piuttosto che di collaborazione, alcune parti neutrali sono tenute ad arbitrare queste interazioni.

Sono riconosciute a livello internazionale delle Organizzazioni il cui scopo è quello di definire e documentare le norme di attuazione e implementazione per lôintegrazione, i cosiddetti standard.

Gli standard di rete sono tipicamente sviluppati da un comitato, che si compone di inventori, sviluppatori e venditori, che hanno un interesse in una tecnologia specifica. La maggior parte dei comitati sono di portata internazionale, e si incontrano di persona solo per discutere su questioni importanti, fanno invece molto affidamento sulla collaborazione online per la maggior parte della loro ricerca.

Ci sono due gruppi principali che sono produttori e amministratori di norme che influenzano le tecnologie VoIP. Il primo è **International Telecomunication Union**, o **ITU**, che ha sede a Ginevra, Svizzera. Il lavoro della ITU risale al 1865 quando gli accordi sono stati sviluppati per supportare le connessioni tra gli impianti dei telegrafi su scala nazionale.

Da quando le nuove tecnologie radio, televisione, satellite, telefonia digitale, e ora VoIP sono emerse, l'ITU si è ampliato e accresciuto. Al momento attuale, il lavoro dell'ITU è diviso in tre settori: il settore delle radiocomunicazioni (chiamato **ITU-R**), che gestisce lo spettro delle frequenze wireless disponibili; il settore delle telecomunicazioni (**ITU-T**), che sviluppa a livello internazionale gli standard di rete; il terzo settore è lo sviluppo delle telecomunicazioni (**ITU-D**), che si sforza di rendere moderni i servizi di

telecomunicazione a disposizione di persone in paesi in via di sviluppo. Gli sforzi di ITU-T hanno prodotto molti standard di rete internazionali, tra cui *Integrated Services Digital Network* (ISDN) e *Asynchronous Transfer Mode* (ATM), con un focus sulle tecnologie di rete di ampia area (rifacendosi ai loro primi giorni di interconnessioni come telegrafo internazionale). Gli standard ITU-T sono indicati con una lettera, che identifica una specifica area della tecnologia, seguita da una serie di numeri che identificano il tipo particolare. Ad esempio, le norme che iniziano con la lettera H trattano di sistemi audiovisivi e multimediali, compreso il VoIP. Uno degli standard spesso citati in questo settore per il VoIP è H.323, intitolato packet-based Multimedia Communications Systems.
L'altro ente chiave negli standard VoIP è la **Internet Society** (ISOC).
Questa organizzazione è stata la camera di compensazione a livello mondiale per le tecnologie legate a Internet dal 1992, e come tale è sostanzialmente più giovane di ITU. Questa differenza di età causa un diverso orientamento, nonché, dove l'ITU ha una storia ricca di standard per le reti a circuiti commutati, come quello voce, il più giovane ISOC si concentra sulla commutazione a pacchetto e sulla trasmissione dati.
Come ITU, tuttavia, ISOC partiziona il suo lavoro in gruppi più piccoli, tra cui Internet Architecture Board (**IAB**), la Internet Research Task Force (**IRTF**), l'Internet Engineering Steering Group, e l'Internet Engineering Task Force (**IETF**).
L'IETF è responsabile dello sviluppo e della pubblicazione degli standard Internet, che sono chiamati *Request for Comments*, o documenti RFC. RFC inizia come progetto di documenti provenienti da uno specifico gruppo di lavoro, e dopo un'ampia rassegna e approvazioni viene assegnato un numero e quindi reso disponibile online dal curatore RFC. Esempi di RFC includono il *Protocollo Internet* (IP), RFC 791; il *Transmission Control Protocol* (TCP), RFC 793, lô *Hypertext Transmission Protocol* (HTTP), RFC 2616, e il *Session Initiation Protocol* (SIP), RFC 3261. Altre organizzazioni possono anche influenzare gli standard VoIP, ma con un focus più regionale o su specifiche tecnologie. Questi includono: l'American National Standards Institute (**ANSI**), l'European Telecommunications Standards Institute (**ETSI**), il World Wide Web Consortium (**W3C**), e il consorzio internazionale di teleconferenza multimediale (**IMTC).**

1.1 Perchè il TCP/IP non è sufficiente per il voip?

Voice over Internet Protocol (VoIP), permette di combinare entrambe le comunicazioni voce e dati. Questa combinazione è un poôcome un matrimonio, in cui due sistemi unici

si adoperano per creare un certo tipo di sinergica convivenza. Ma capire i punti di forza e di debolezza di ciascun componente è la chiave fondamentale per far funzionare questa relazione, lo stesso vale per lôunione tra voce e dati.

Le tradizionali reti voce sono classificate come reti orientate alla connessione, in cui un percorso dalla sorgente alla destinazione è stabilito, prima di qualsiasi trasferimento di informazioni. Quando l'utente solleva il telefono, lôapparecchio informa la rete che viene richiesto un servizio. La rete dunque restituisce il segnale di linea, e l'utente compone il numero di destinazione. Quando l'interlocutore risponde, la connessione end-to-end è confermata attraverso i vari uffici di commutazione lungo il percorso. Quando la conversazione finisce, le due parti riagganciano, e le loro risorse di rete possono essere ri-assegnate per unôaltra conversazione.

Uno degli svantaggi di questo processo è il consumo delle risorse utilizzate per istituire la chiamata. Uno dei vantaggi, tuttavia, è che una volta che la chiamata è stata stabilita, e un percorso attraverso la rete è stato definito, le caratteristiche di tale percorso, come ritardo di propagazione, la sequenza di informazione, ecc dovrebbero rimanere costanti per tutta la durata della chiamata. Queste costanti aggiungono affidabilità al sistema; il termine spesso usato per descrivere un ambiente orientato alla connessione è dunque affidabile (*reliable*). Il Transmission Control Protocol (TCP) è un esempio di un protocollo orientato alla connessione.

Al contrario, le reti di dati sono classificate come reti non orientate alla connessione (connectionless). Nel pacchetto di informazioni è allegata la fonte e l'indirizzo completo di destinazione, e poi quel pacchetto è inviato nella rete per la consegna alla destinazione finale. Una analogia alle reti *connessionless* è il sistema postale, in cui si fa cadere una lettera nella cassetta postale, e se tutto funziona secondo i piani, la lettera viene trasportata a destinazione. Non conosciamo il percorso che il pacchetto (o lettera) avrà, ma a seconda del percorso, il ritardo potrebbe variare notevolmente. E 'anche possibile che il nostro pacchetto possa perdersi o essere mal consegnato all'interno della rete, e quindi non raggiungere la destinazione. Per questi motivi, i termini di *best effort* (massimo sforzo) e *unreliable* (non affidabile) sono spesso usati per descrivere un ambiente non orientato alla connessione. Il protocollo Internet (IP) e l'User Datagram Protocol (UDP) sono esempi di protocolli connectionless.

I protocolli TCP, IP e UDP sono stati sviluppati negli anni 1970 e 1980 per sostenere tre applicazioni chiave:

- trasferimenti di file utilizzando il **File Transfer Protocol** (FTP)
- la posta elettronica utilizzando **Simple Mail Transfer Protocol** (SMTP)
- lôaccesso remoto a computer usando il protocollo **TELNET**.

Tutte queste applicazioni sono orientate ai dati e non alla voce, ed erano quindi progettate sulla base di una rete connectionless. Mettere TCP su IP ha dato a tutto il sistema maggiore affidabilità, anche se con un overhead (costo aggiuntivo) di protocollo. Quindi la qualità e la puntualità di un vera infrastruttura orientata alla connessione (come la rete telefonica) non era necessaria per sostenere queste applicazioni.

La previsione di un utilizzo sempre maggiore di applicazioni voce e video su protocollo IP però ha fatto cambiare direzione. Tali applicazioni domineranno il panorama e i flussi di dati attraverso le reti. Applicazioni come quelle che sfruttano la tecnologia VoIP saranno sensibili allôordinamento e alle questioni di ritardo (*delay*) e l'idea di un servizio "*best effort"*, (specialmente se la conversazione vocale è una chiamata alla polizia o vigili del fuoco) non raccoglie molti sostenitori.

Queste considerazioni portano a questa domanda: Come possiamo supportare applicazioni orientate alla connessione (come voce e video) in un ambiente *connectionless* (come IP), senza riprogettare completamente l'infrastruttura di rete?

La soluzione è rafforzare IP con protocolli aggiuntivi che riempiono alcune delle sue lacune. Alcuni di questi sono:

- **Real-time Transport Protocol** (RTP), definita nella RFC 3350.

Fornisce funzioni come l'identificazione del carico utile, sequenza di numerazione, e timestamp sulle informazioni.

- **RTP Control Protocol** (RTCP), anche definito nella RFC 3350.

Controlla la qualità della connessione RTP.

- **Multicast Internet Protocol** (IP Multicast), definito in RFC 1112 e 2236.

Permette di inviare informazioni da una singola fonte e raggiungere più destinazioni (come può essere richiesto per le conferenze).

- **Session Description Protocol** (SDP), definita nella RFC 2327.

Fornisce informazioni sui flussi multimediali per una particolare sessione, incluso il nome della sessione, il tempo della sessione, e ciò che i media (voce, video, ecc) devono utilizzare, la larghezza di banda richiesta, e così via.

Capitolo Secondo

2 Il protocollo SIP

Alcuni protocolli già conosciuti e largamente utilizzati per comunicare via internet hanno influenzato la nascita del protocollo SIP.

ITU-T e IETF si approcciano all'architettura di rete e al disegno di protocolli da punti di vista diversi. ITU-T essendo un'entità internazionale affiliata alle Nazioni Unite considera il consenso come priorità, e come risultato gli standard che produce arrivano attraverso una serie di iterazioni e rinnovi prima di produrre un risultato finale concordato da tutti. IETF invece è un ente più pragmatico che segue la filosofia: "*rough consensus and running code*" per dire che qualcuno può anche dissentire ma quello che conta è ottenere un sistema funzionante che possa essere velocemente implementato. Il motto proviene da un discorso di David D. Clark che nel 1992 nello specifico dice: "*We reject: kings, presidents and voting. We believe in: rough consensus and running code*". Questo illustre scienziato tra il 1981 e il 1989 era a capo dell'ente IAB (Internet Architecture Board) per lo sviluppo e la creazione di Internet. Attualmente presta servizio presso il *MIT Computer Science and Artificial Intelligence Laborators* in qualità di ricercatore senior.

In molti casi dunque, lo sviluppo di standard da parte dell'IETF segue dei cicli più brevi rispetto alla ITU-T. Questo è dovuto anche al fatto che IETF ogni tre mesi organizza degli incontri aperti a sviluppatori e ricercatori di tutto il mondo. Se dopo un'incontro su un argomento rimangono dei dubbi, nell'arco di pochi mesi si ha modo di riaffrontare e risolvere la questione. Questo approccio pragmatico è chiaramente rispecchiato nell'architettura del Session Initiation Protocol.

2.1 Architettura del protcollo SIP

Lôarchitettura del protocollo SIP è costruita sulla base di altri due protocolli fondamentali per Internet. LôSMTP e lôHTTP.

- Il primo stabilisce il formato con il quale sono trasferiti i messaggi di posta elettronica.
- Il secondo stabilisce il formato secondo il quale avviene la comunicazione multimediale via internet.

Similmente a questi due protocolli anche il SIP si appoggia ai protocolli UDP o TCP del livello sottostante, quello di trasporto. SIP inoltre viene coadiuvato dal Realtime Transport Protocol RTP e dal Session Description Protocol SDP.

RTP per trasferire i dati in tempo reale e SDP per trasferire lôinformazione del tipo di media al destinatario. Si può pensare dunque al SIP come un ulteriore componente di una *suite* di protocolli tutti legati a internet.

SIP, in quanto protocollo del livello applicazione, provvede a fornire dei servizi allôutente finale. Il servizio operativo che contraddistingue SIP è la *Sessione,* la quale può essere definita come un scambio ordinato di informazioni tra due o più partecipanti. Guardando nel dettaglio, il SIP deve per primo agire per creare la sessione e poi mantenerla per tutto il tempo che la comunicazione è necessaria. Anche se sembra semplice, può sorgere qualche complicazione:

- La prima può essere lôesistenza di più di due partecipanti. Vuol dire che la comunicazione sarà una conferenza invece che una chiamata end-user-to-end-user.

- La seconda è che lôutente non sempre inizia la chiamata dalla stessa posizione geografica, aggiungendo la necessità di tener traccia di questi utenti.

- La terza è che lôutente può utilizzare un misto di tecnologie multimediali tra testo, voce e video, tutte con parametri diversi sia per la banda sia per il massimo *delay* consentito

Per dare supporto a questo servizio di sessione, la RFC 3261 descrive cinque aspetti di gestione:

1. User location: registra la posizione dellôutente per la comunicazione
2. User availability: determina se lôutente è disponibile oppure no
3. User capabilities: determina i parametri multimediali che possono essere utilizzati per la comunicazione verso lôutente

4. -Session setup: stabilisce i parametri di sessione per il chiamante e il chiamato.
5. -Session management: include l'inizio e la fine della sessione , la modifica dei parametri di sessione o l'invocazione di servizi aggiuntivi.

Questi cinque aspetti vengono affrontati e risolti dal protocollo sip tramite un'architettura sviluppata e costruita su due pilastri fondamentali: client e server.

2.1.1 Client/Server

Similmente all'SMTP e all'HTTP anche il SIP descrive due meccanismi funzionali: i client e i server.

- Un *client* è un elemento di rete che invia richieste SIP e riceve risposte SIP.

- Un *server* è un elemento di rete che riceve richieste SIP in modo da servirle e poi rispondere a queste.

L'interazione tra client e server è modellata sul paradigma di richiesta/risposta proprio dell'http. Nel protocollo http il client manda una richiesta per una risorsa di rete a un server. Il server risponde con la risorsa cercata, ad esempio una pagina web.

Nel caso del SIP il client invia una richiesta al server per una risorsa con cui comunicare.

Un esempio di richiesta SIP è un messaggio di INVITO dal client al server che indica che l'utente è chiamato a partecipare a una sessione di comunicazione voce.

Così il SIP può operare come un membro effettivo della rete internet.

Alcuni esempi di client e server sono:

- **User Agent Client:** la funzione logica che crea una richiesta, e poi usa le funzionalità del client per inviare la richiesta.
- **Proxy:** un intermediario che funge sia da client che da server e può inviare richieste a nome di un altro client
- **User Agent Server:** la funzione logica che genera la risposta alla richiesta SIP.
- **Redirect Server:** un server che redirige il client verso un altro server per completare la sua richiesta.
- **Registrar:** un server che accetta richieste del tipo REGISTER e che inserisce questa informazione all'interno del servizio di localizzazione.

2.1.2 Formato dei messaggi

Il tipo di messaggio *richiesta* definisce le operazioni da parte del client, mentre il messaggio di *risposta* contiene le informazioni da parte del server che indicano al client lo stato della richiesta.

Siccome le risorse per una sessione SIP sono risorse di comunicazione e non pagine web o file, si rende necessario uno schema di indirizzamento o identificazione per tutte le risorse prima dell'invio della richiesta e della risposta. L'identificatore si chiama SIP Uniform Resource Indicator (**SIPURI**). Esso contiene sufficienti informazioni per iniziare e mantenere una sessione di comunicazione. Il SIPURI è molto simile a un indirizzo di posta elettronica (in prestito dal protocollo SMTP) e tipicamente contiene due parti: una parte scelta dall'utente, ad esempio il nome, un'altra parte fissa, ad esempio il dominio di rete. Ad esempio: sip:Marconi@paripari.it. Sono possibili altri tipi di SIPURI, se in una stessa rete due utenti hanno scelto lo stesso nome, si pone una stringa casuale vicino al nome, ad esempio sip:Marconi0xfGh@paripari.it.

Con questo schema di indirizzi ci sono sei tipi di messaggi di richiesta che un client può inviare.

- REGISTER: utilizzato per registrare un indirizzo con un SIP server
- INVITE: indica che un utente invita qualcuno a partecipare a una sessione. Il corpo del messaggio conterrà una descrizione del tipo di sessione alla quale si è invitati.
- ACK: messaggio di conferma a un INVITE
- CANCEL: usato per cancellare una richiesta pendente
- BYE: il client indica al server che desidera terminare la sessione
- OPTIONS: utilizzato per domandare al server circa le sue capacità e caratteristiche.

I messaggi di risposta contengono dei codici di stato che indicano la condizione della richiesta corrente. Questi codici sono divisi in sei categorie generali.

- 1xx: Provisional: la richiesta è stata ricevuta e sta continuando
- 2xx Success: la richiesta è stata ricevuta, capita e accettata con successo.
- 3xx Redirection: serve un'altra azione per processare la richiesta
- 4xx Client Error: la sintassi della richiesta è sbagliata e non può essere gestita dal server
- 5xx Server Error: Il server ha sbagliato a processare una richiesta che sembrava valida.

- 6xx Global Failure: La richiesta non può essere processata da nessun server.

2.2 Un semplice esempio di sessione

La figura 1 mostra lo scambio di messaggi SIP tra due end-point. I due dispositivi possono essere ad esempio telefoni SIP, palmari oppure telefoni cellulari. Si assume inoltre, che entrambi i dispositivi siano connessi ad una rete IP, ad esempio Internet, e conoscano entrambi l'indirizzo IP dell'altro.

Il chiamante, Tesla, comincia lo scambio di messaggi inviando una richiesta di INVITE alla parte chiamata, Marconi. Il messaggio di INVITE contiene i dettagli sul tipo della sessione che verrà stabilità: potrebbe trattarsi di una semplice sessione audio, oppure di una sessione multimediale quale una videoconferenza, oppure ancora potrebbe essere una sessione di gioco.

Il messaggio di INVITE si presenterà così:

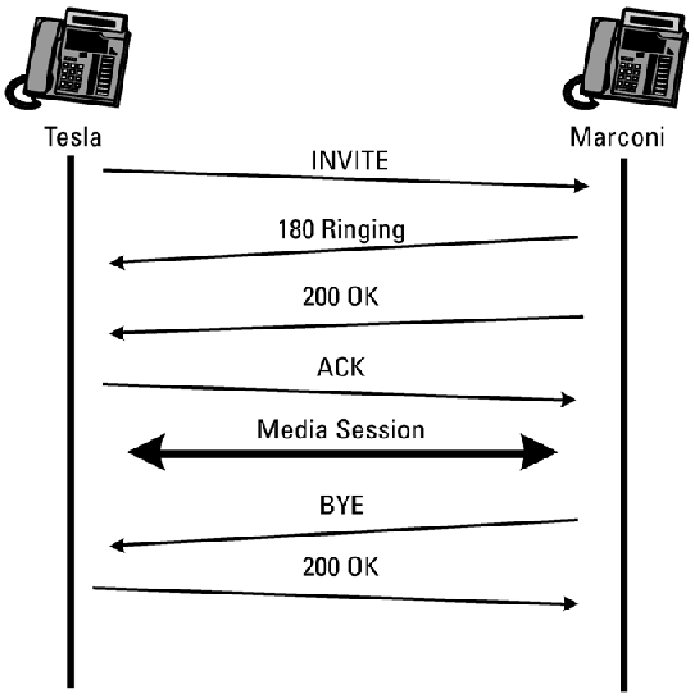

Figura 1 - Esempio di sessione SIP

INVITE sip:marconi@radio.org SIP/2.0
Via: SIP/2.0/UDP lab.high-voltage.org:5060; branch=z9hG4bKfw19b
Max-Forwards:70 To: G. Marconi <sip:Marconi@radio.org>
From: Nikola Tesla <sip: n.tesla@high-voltage.org>; tag=76341
Call-ID: 123456789@lab.high-voltage.org
Cseq: 1 INVITE
Subject: About That Power Outage...
Contact: <sip: n.tesla@high-voltage.org>
Content-Type: application/sdp
Content-Length: 158
v=0
o=Tesla 2890844526 2890844526 IN IP4 lab.high-voltage.org
s=Phone Call
c=IN IP4 100.101.102.103
t=0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000

I campi elencati nel messaggio INVITE sono chiamati campi d'intestazione, o header fields. Il messaggio successivo, è un messaggio 180 Ringing inviato in risposta ad INVITE. Questo messaggio indica che Marconi, la parte chiamata, ha ricevuto l'INVITE e sta avvenendo la segnalazione. Tale segnalazione può consistere nello squillo di un telefono, nella visualizzazione di un messaggio a video, oppure in un qualsiasi altro metodo attragga l'attenzione di Marconi.
180 Ringing è un esempio di messaggio SIP di risposta. Le risposte sono tutte identificate da un numero, e vengono classificate in base alla prima cifra di tale numero. Una risposta 180 appartiene alla Informational class di risposte, categoria che raggruppa tutte le risposte la cui prima cifra è 1. Le risposte informative, vengono utilizzate per trasmettere informazioni non critiche sullo stato di avanzamento della chiamata.
La risposta 180 Ringing ha la seguente struttura:

SIP/2.0 180 Ringing
Via: SIP/2.0/UDP lab.high-voltage.org:5060;branch=z9hGbKfw19b;
received=100.101.102.103

To: G. Marconi <sip:marconi@radio.org>; tag=7634

From: Nikola Tesla <sip: n.tesla@high-voltage.org>; tag=76341

Call-ID: 123456789@lab.high-voltage.org

Cseq: 1 INVITE

Content-Lenght: 0

Quando Marconi accetta la chiamata, allora viene spedita una risposta di tipo 200 OK. Tale risposta indica inoltre che il tipo di sessione multimediale proposta dal chiamante è accettabile per il destinatario. 200 OK appartiene alla Success class di risposte SIP.
Il passo finale è confermare la sessione multimediale con un acknowledgement (ACK). Tale conferma sta a significare che Tesla ha ricevuto con successo la risposta di Marconi. Questo cambio di informazioni, consente alla sessione multimediale di essere stabilita utilizzando un altro protocollo: RTP, Real Time Protocol. Sempre in Figura 1, si può notare come una richiesta BYE sia inviata da Marconi per terminare la sessione multimediale. La conferma a questa richiesta sarà nuovamente 200 OK.

2.3 SIP e Trasporto UDP

Quando usiamo UDP, ogni messaggio di richiesta o risposta SIP viene trasportata in un unico datagramma UDP. La figura 2 illustra lo scambio del messaggio BYE durante una sessione SIP, usando UDP.

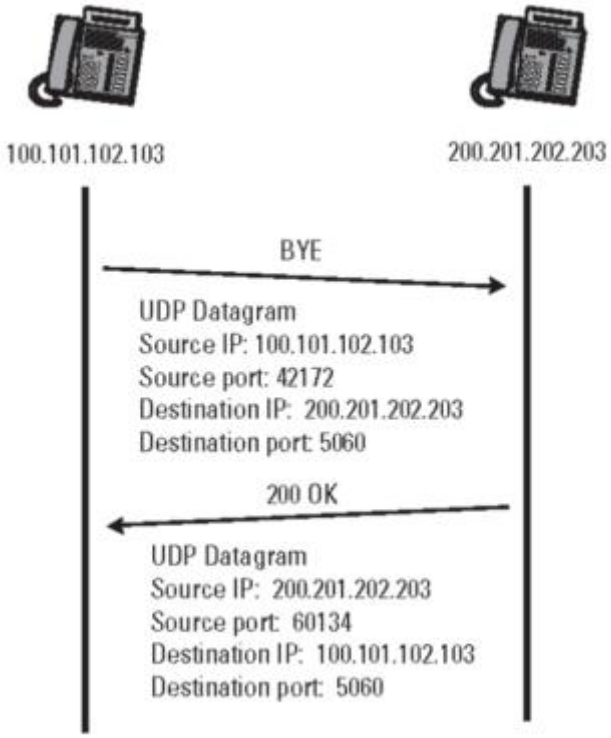

Figura 2 - Messaggio SIP usando UDP

La porta sorgente viene scelta da un insieme di porte disponibili (nell'esempio 42172), oppure qualche volte viene utilizzata la porta di default del servizio SIP: la porta 5060. La mancanza di handshaking o acknowledgement nel trasporto UDP, sta a significare che un pacchetto può essere perso, e con esso può andare perduto anche il messaggio SIP. L'utilizzo di checksum, comunque, consente ad UDP di scartare i pacchetti contenenti errori, permettendo così a SIP di assumere che un messaggio venga ricevuto completo e privo di errori. La risposta, viene anch'essa mandata alla porta 5060 (la destination port sarà sempre 5060, perchè è quella propria del servizio SIP), oppure al numero di porta presente nel campo di intestazione Via.

UDP fornisce quindi un semplice trasporto per User Agent e server, tuttavia non offre alcun controllo di congestione. Un serie di pacchetti persi, su un collegamento IP già pesantemente carico, causerà ritrasmissioni, le quali produrranno un numero ancora maggiore di pacchetti persi, portando così la linea al collasso.

2.4 SIP Headers

In questa sezione descriveremo i campi di intestazione presenti nei messaggi SIP che si è scelto di implementare, in quanto ritenuti necessari dallo standard.
I SIP header fields sono simili ai campi di intestazione HTTP, sia nella sintassi che nella semantica; essi sono definiti con Header: field, dove Header è la parte case-insensitive (ma per convenzione è tutto minuscolo con alcune maiuscole) che rappresenta il nome del campo d'intestazione, mentre field è la parte, sempre case-insensitive, che contiene le informazioni. Salvo dove altrimenti sottolineato, l'ordine dei campi d'intestazione in un messaggio non è importante.
Come considerazione finale, gli header fields possono essere o end-to-end oppure hop-by-hop. I campi di intestazione hop-by-hop sono gli unici che un proxy può inserire o, con qualche eccezione, modificare. Dato che SIP tipicamente coinvolge il controllo end-to-end, la maggior parte dei campi d'intestazione è per l'appunto end-to-end. L'unico ad essere hop-by-hop, tra quelli di nostro interesse è Via.
Gli header fields implementati sono: Call-ID, CSeq, From, To, Via.

2.4.1 Call-ID

Il campo di intestazione Call-ID è obbligatorio in tutte le richieste e risposte SIP; esso viene usato per identificare univocamente una chiamata tra due utenti.
Il Call-ID deve essere unico tra le chiamate, eccetto nel caso di un Call-ID nelle richieste di registrazione: tutte le registrazioni di uno stesso UA devono usare il medesimo Call-ID. Tale header field viene sempre creato da uno UA e non viene mai modificato da un server. Call-ID è solitamente composto da un local-id, oppure da un local-id seguito dal simbolo @ e da un indirizzo IP oppure da un host name. Il suddetto local-id è un cryptographically random identifier - Identifcativo Casuale ottenuto con procedimento Crittografico.
Dato che uno UA può assicurare che il suo local-id sia unico all'interno del suo dominio, l'aggiunta di un host name globalmente unico rende anche Call-ID globalmente unico. Una certa sicurezza viene fornita dalla casualità di Call-ID, in quanto essa ostacola una terza parte dall'indovinare l'identificativo e quindi, di conseguenza, dal presentare false richieste.
La forma compatta del campo di intestazione Call-ID è i. Si riportano qui alcuni esempi:

Call-ID: 34a5d553192cc35@15.34.3.1

Call-ID: 44fer23ei4291dekfer34231232

i: 35866383092031257@port34.carrier.com

Il codice che costruisce il CallId è il seguente, e per esso sono stati scritti test in JUnit per un perfetto funzionamento.

```
public String callIdBuilder(){
        int length = 32;
        char[] symbols = new char[36];
        for (int idx = 0; idx < 10; ++idx)
                symbols[idx] = (char) ('0' + idx);
        for (int idx = 10; idx < 36; ++idx)
                symbols[idx] = (char) ('a' + idx - 10);
        final Random random = new Random();
        final char[] buf;
        if (length < 1)
                throw new IllegalArgumentException("length < 1: " + length);
        buf = new char[length];
        for (int idx = 0; idx < buf.length; ++idx)
                buf[idx] = symbols[random.nextInt(symbols.length)];
        return new String(buf);

}
```

All'interno del codice si può vedere come la stringa che compone l'identificativo univoco sia create attraverso una selezione casuale di simboli e numeri. Questa selezione passa attraverso un vettore di caratteri che poi vengono concatenati per formare la stringa effettiva che viene restituita in uscita.

2.4.2 CSeq

Il campo d'intestazione CSeq è necessario in ogni richiesta SIP (la parola CSeq sta per command sequence). Questo header field contiene un numero decimale che è incrementato a ogni richiesta: di solito, esso cresce di uno a ogni nuova richiesta, ad eccezione delle CANCEL request e delle ACK request, per le quali è usato il numero di CSeq dell'INVITE al quale si riferiscono.
Questo sistema di conteggio è utilizzato dagli UAS per determinare quali richieste arrivano fuori sequenza oppure per differenziare tra una nuova richiesta (diverso CSeq) oppure una ritrasmissione (stesso CSeq). Il campo CSeq viene invece utilizzato dagli UAC per far corrispondere una risposta alla richiesta a cui essa fa riferimento.
Degno di nota è che ogni UA mantiene il suo spazio numerico per il command sequence. Per esempio, consideriamo il caso in cui uno User Agent 1 stabilisce una sessione con una User Agent 2 e inizializza il suo CSeq ad 1. Quando lo User Agent 2 vorrà mandare una richiesta (come un INVITE, un INFO o anche un BYE), inizializzerà un suo spazio per CSeq, totalmente indipendente dal conteggio usato dallo User Agent 1.
Alcuni esempi di campo CSeq sono:

CSeq: 1 INVITE Il numero di command sequence è stato inizializzato a 1 per l'INVITE iniziale.

CSeq: 432 REFER Il numero di command sequence è impostato a 432 per questa REFER.

CSeq: 6787 INVITE Se questa fosse stata la prima richiesta dello UA nella comunicazione, allora o CSeq veniva inizializzato a 6787, oppure la precedente richiesta generata per questo stesso Call-ID aveva un CSeq pari a 6786, o inferiore.

2.4.3 From

Anche il campo d'intestazione From è obbligatorio e fornisce un' indicazione in merito all'identità logica del promotore di una richiesta; esso è principalmente costituito da un URI.
Il campo From può contenere un tag, usato per identificare una particolare chiamata, e può contenere anche la visualizzazione del nome del mandante la richiesta (in tal caso l'URI è racchiuso tra < >). Se sono presenti sia l'URI che il tag, allora l'URI e qualunque altro suo parametro, vanno inclusi tra < >.
Il tag nel campo From era opzionale nella RFC 2543, ma è diventato obbligatorio includerlo dalla RFC 3261. Esempi di campo From sono:
From: <sip:armstrong@hetrodyne.com>; tag=3342436
From: Thomas Edison <sips:edison@electric.com>; tag=532
f: James Bardeen <sip:555.1313@telephone.com;

Come si può vedere dall'ultimo esempio, la forma compatta del campo From è f.

2.4.4 To

Il campo d'intestazione To è anch'esso obbligatorio in ogni messaggio SIP, e in primo luogo esso specifica il destinatario desiderato di una richiesta. Qualsiasi risposta generata da uno UA conterrà questo campo d'intestazione, con l'aggiunta di un tag (per quanto riguarda risposte generate da proxy l'aggiunta del tag è obbligatoria). Un tag aggiunto al campo To in una risposta 200 OK, viene usato per tutta la chiamata ed è incorporato nella comunicazione. Il campo To non viene mai usato per l'instradamento (routing), a tale scopo viene utilizzato Request-URI.
Come il campo From, anche il campo To consente l'opzionale visualizzazione del nome, e anch'esso ha una forma compatta: t. Ecco alcuni esempi:

To: sip:babage@engine.org; tag=2443a8f7
To: Thomas Edison <sips:edison@electric.com>
t: <+1-314-555-1212@carrier.com; user=phone>; tag=8f7f7ad6675

2.4.5 Via

Il campo Via, anch'esso necessario, viene usato per memorizzare la strada intrapresa da una richiesta e, successivamente, per instradare la risposta indietro fino al mandante della richiesta.

Un UA che genera una richiesta, memorizza il suo indirizzo nel campo Via della richiesta. Mentre l'ordine della maggior parte dei campi d'intestazione SIP non è importante, l'ordinamento del campo Via è rilevante, in quanto esso viene utilizzato per instradare le risposte. Un proxy che deve inoltrare una richiesta, aggiunge un campo Via in cima alla lista dei vari campi Via già presenti.

Un proxy o uno UA che deve generare una risposta a tale richiesta, copia nella risposta tutti i campi Via, nell'esatto ordine in cui si trovano nella richiesta; successivamente, il proxy o l'UA in questione invierà la risposta all'indirizzo specificato nel campo Via che si trova in cima alla lista.

Quando un proxy riceve una risposta, verifica il primo campo Via della lista per assicurarsi che corrisponda con il suo indirizzo: se ciò non accade, la risposta non è stata instradata correttamente e pertanto viene scartata; se invece viene verificato il match, il primo campo Via viene rimosso, e la risposta viene inoltrata all'indirizzo specificato nel prossimo campo Via in elenco.

I campi d'intestazione Via contengono inoltre il nome del protocollo, la versione e il nome del protocollo di livello transport utilizzato (SIP/2.0/UDP, SIP/2.0/TCP, eccetera), e possono contenere inoltre numeri di porte, e parametri quali ad esempio: branch, received, maddr, e ttl.

Riportiamo, come nei casi precedenti, alcuni esempi:

Via: SIP/2.0/UDP 100.101.102.103; Indirizzo IPv4; utilizzo di trasporto unicast
branch=z9hG4bK776a e della porta di default 5060
Via: SIP/2.0/TCP 192.168.1.2; L'indirizzo è un indirizzo multicast.
received=12.4.5.50; L'indirizzo IPv4 non è globalmente unico.
branch=z9hG4bK334
La richiesta è stata inoltrata attraverso
un NAT, che ha cambiato l'indirizzo IP

in uno globalmente unico.

v: SIP/2.0/TCP Viene utilizzato un nome di dominio,

cube451.office.com:60202; il trasporto TCP e la porta 60202.

branch=z9hGbK776a

Capitolo Terzo

3 Il progetto PariPari

PariPari si propone come una rete serverless che garantisce lôanonimato dei nodi e fornisce un sistema di gestione dei crediti molto avanzato. Si definisce serverless in quanto permette di non avere dei nodi sensibili, la cui indisponibilità comprometterebbe tutta la rete. Per ottenere questa caratteristica ci si è basati su una variante di un protocollo esistente e ben collaudato: Kademlia. Con la parola anonimato, invece, si intende che la rete permette di effettuare uno scambio di risorse tra due nodi senza che sia possibile identificare gli indirizzi IP di mittente e destinatario. Come in altre reti peer to peer, lo scambio di risorse tra due nodi, è regolato da un complesso sistema di crediti. Senza entrare nel dettaglio, si può dire che questo sistema favorisce i client che condividono molte risorse e, allo stesso tempo, evita che client appena entrati, prosciughino le risorse della rete senza dare nulla in cambio. I client della rete sono sviluppati in Java, linguaggio noto soprattutto per la sua portabilità su tutti i principali sistemi operativi e che, grazie alla tecnologia Java Web Start, si integra perfettamente con i più noti browser. Ogni client della rete è realizzato con unôarchitettura a plugin, concepita per permettere la realizzazione di qualsiasi servizio il web possa offrire. I plugin sono attualmente divisi in due gruppi: la cerchia interna, che comprende quelli necessari al funzionamento della rete, e la cerchia esterna che comprende altri servizi messi a disposizione dellôutente. Della cerchia interna fanno parte:

- ÅCore
- ÅConnectivity
- ÅLocal Storage
- ÅDHT (Distributed Hash Table)
- ÅCrediti

Mentre di quella esterna, si possono tra gli altri citare:

- ÅBitTorrent
- ÅIRC
- ÅVoIP

Lo sviluppo di ogni plugin segue la strategia di progettazione XP (eXtreme programming), una metodologia agile basata sul testing e che prevede la suddivisione del lavoro in team. Ogni modifica o aggiunta al codice è suddivisa in fasi: si preparano prima le interfacce necessarie, in seguito si sviluppano i test e solo successivamente si passa alla vera implementazione.

3.1 Il plugin VoIP

Il plugin VoIP fa parte della cerchia esterna dei plugin e il suo scopo è quello di fornire agli utenti la possibilità di mettersi in contatto vocale innanzitutto con gli altri utenti della rete PariPari, in futuro è prevista la possibilità di effettuare chiamate anche verso altri client VoIP. Il plugin è predisposto per entrare in contatto con altri utenti della rete PariPari attraverso i servizi forniti da DHT (Distributed hash table) il quale implementa una propria versione dell'algoritmo Kademlia.

3.1.1 I protocolli e i codec

Per l'instaurazione di una chiamata tra due utenti connessi alla rete PariPari, si è deciso di adottare un protocollo molto diffuso soprattutto in ambito VoIP: il protocollo SIP (Session Initiation Protocol). Ci sarebbe stata anche la possibilità di creare un protocollo ad hoc sfruttando i servizi messi a disposizione in PariPari, ma questo non avrebbe permesso una futura integrazione con altri client VoIP. Per lo streaming audio, invece, ci si è orientati verso il protocollo RTP (Real-Time Protocol) per la trasmissione ed il codec Speex per la codifica. Il protocollo RTP, come suggerito dal nome stesso, è adatto ad applicazioni in tempo reale e fornisce vari servizi utili al trasporto di pacchetti audio/video; inoltre ad esso è associato il protocollo RTCP (Real-Time Control Protocol), che può essere usato per monitorare la qualità e le statistiche di trasmissione. Il codec Speex, invece, è un codec open source per la compressione della voce distribuito sotto Licenza BSD; i suoi sviluppatori hanno inoltre posto molta attenzione alla sua applicazione in progetti di Voice over IP. Esiste poi un terzo protocollo, il protocollo SDP (Session Description Protocol) che ha lo scopo di definire una sessione, veicolando informazioni quali, ad

esempio:

- il nome e lo scopo
- gli istanti dôinizio e di fine
- i tipi di media che la compongono
- gli indirizzi di ricezione dei media

Esso viene usato congiuntamente al protocollo SIP che lo incapsula al suo interno durante la fase dôinstaurazione di una sessione. Tutti questi protocolli possono essere utilizzati sia con TCP che con UDP. La scelta è ricaduta su UDP, in quanto sostanzialmente più adatto ad applicazioni di streaming real-time, dove non si cerca lôaffidabilità ma ciò che interessa è che i pacchetti arrivino il più velocemente possibile e la perdita di uno di essi non causa alcun danno.

3.1.2 Librerie Java

Per tutti i protocolli citati precedentemente esistono varie librerie Java open source utilizzate nel progetto. Dopo unôattenta analisi, si è deciso di utilizzare per la parte riguardante SIP il progetto *Peers*, distribuito con licenza GPL.

Per quanto riguarda il protocollo RTP, dopo alcuni test di performance, la scelta è ricaduta sulla libreria *jRTP*, libreria non molto complessa bensì snella e facilmente personalizzabile.

Il codec *Speex* non vanta invece una grande tradizione nel mondo Java, esiste soltanto un porting del codice scritto in C, chiamato *jSpeex*, che ha ormai alcuni anni. Tuttavia dopo vari tentativi si è capito come utilizzarlo e sfruttarne tutte le caratteristiche.

Per lôimplementazione del protocollo SDP ci si è affidati allôunica libreria disponibile in java jSDP

3.1.3 Performance

Il plugin VoIP è stato pioniere nello studio delle performance e nella raccolta di statistiche riguardanti il comportamento della JVM (Java Virtual Machine) durante lôesecuzione del plugin.
Una volta raggiunta la stabilità del codice testato a dovere tramite TDD (Test Driven Development), il plugin VoIP è stato sottoposto ad unôaccurata analisi delle performance.
Tramite lôutilizzo dello strumento *jConsole* sono state presi in considerazione tre parametri fondamentali per lôanalisi del comportamento del plugin sia sotto sforzo sia in fase iniziale. I parametri sono:

- CPU Usage: quanta cpu sfrutta, in percentuale, il plugin VoIP
- Heap memory Usage: lôoccupazione di memoria RAM che occupa il plugin.
- Threads: il numero di thread attivi che mantengono attivo il plugin.

Le prime misurazioni che sono state fatte mostravano come lôutilizzo di CPU si aggirasse attorno al 10% allôavvio del pugin, la memoria utilizzata fosse di circa 20 Mb e i thread attivi fossero 5.
Perché dunque il plugin, allôavvio, occupa così tanto il processore?
Da una approfondita analisi effettuata tramite gli strumenti quali jConsole, si è risaliti ai thread attivi allôavvio del plugin. Dopo aver fatto alcune considerazioni su cosa comportassero le modifiche per migliorare la situazione e cosa ci si aspettasse, si è passati allôeffettiva modifica. In sostanza si è riorganizzato lo schema dei thread utilizzati per gestire il flusso audio. Modificando la sincronia con la quale questi processi collaborano tra loro si sono raggiunte performance di molto migliori. Lôoperazione di refactoring ha coinvolto le classi principali del plugin quali Voip, VoipCore, Focus, VoipStreamSender , Call e Conference.
Dopo le modifiche, lôutilizzo di CPU si è attestato sullo 0,5%, la memoria utilizzata sui 7Mb con solo 2 thread attivi.
Le performance sono state misurate anche durante una chiamata point-to-point durata circa unôora, il grafico dellôutilizzo della CPU lo si può vedere in figura 3.

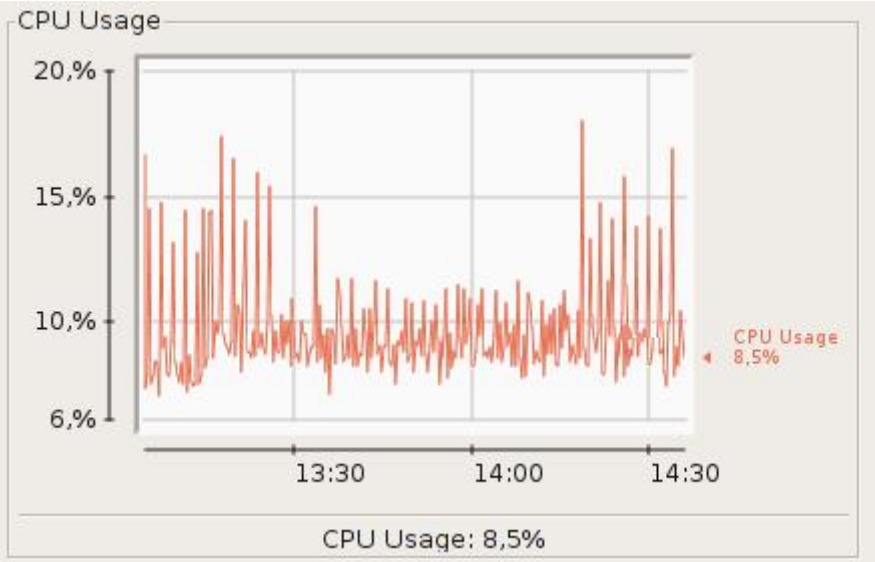

Figura 3 - Utilizzo di CPU durante una chiamata

Lôutilizzo di CPU si è attestato intorno allô8% con picchi del 16% dovuti al rinnovo dei socket UDP ogni 2 minuti. La memoria occupata non ha mai superato i 9Mb, mentre i thread attivi per mantenere la comunicazione sono 7. Le performance ottenute sono di molto migliori rispetto alla situazione attuale.

Conclusioni

Nel tempo durante il quale sono stato a capo del plugin, il VoIP è stato sottoposto più volte a refactoring, sia per quel che riguarda la parte core sia per quel che riguarda lôimplementazione dei protocolli (SIP, RTP, UDP, SDP). Oltre che del protocollo SIP mi sono occupato di tutti gli aspetti di funzionamento e di interazione delle varie componenti sia allôinterno del plugin che allôesterno. Ad esempio della parte socket UDP, della riorganizzazione delle librerie, dellôutilizzo dei thread, dellôinterfaccia grafica, dellôanalisi delle prestazioni. Per ultimo, ma non per importanza, mi sono occupato del testing del plugin attraverso le librerie easyMock e PowerMock.

Questo plugin ha seguito un percorso evolutivo che lôha portato a una versione stabile e funzionante. Lôaffinamento del VoIP può proseguire partendo proprio da qui.

Bibliografia

[1] PaoloBertasi. Progettazione e realizzazione in Java di una rete peer to peer anonima e funzionale., 2006.

[2] Next-Gen VoIP Services and Applications Using SIP and Java, 2001

[3] Alan B. Johnston. SIP: Understanding the Session Initiation Protocol.Second Edition, Artech House, 2004.

[4] Federico Cian. PariPari: VoIP 2008., 2008.

[5] David A. Bryan and Bruce B. Lowekamp: SOSIMPLE:AServerless,Standards-based, P2PSIPCommunicationSystem, 2005

[6] John L. Hennessy, David A. Patterson - Computer Architecture - A Quantitative Approach

[7] Andrew Tanenbaum - Reti Di Calcolatori, 2003