Session Initiation Protocol

SIP protocol and its extensions
SIP Service Architecture
SIP in 3G

Main Sources

IETF:

RFC 3261: SIP: Session Initiation Protocol
RFC 3262: Reliability of Provisional Responses in SIP (PRACK)
RFC 3265: SIP Specific Event Notification
RFC 3311: SIP UPDATE method
RFC 3398: ISUP to SIP mapping
RFC 3428: SIP Extension for Instant Messaging
RFC 2327: SDP: Session Description Protocol
RFC 3264: An Offer/Answer Model with Session Description Protocol (SDP)

3G Release 5:

3GPP TS 24.228 v5.15.0 (2006-10) Signaling flows for the IP MM call control based on SIP and SDP; stage 3 (Release 5)
3GPP TS 24.229 v7.6.0 (2006-12) IP multimedia call control protocol based on SIP (rel 7) and SDP, Stage 3 (Release 7)
3GPP TS 29.228 v7.4.0 (2006-12) IMS Cx and Dx interfaces, Signaling flows and (for Rel 7) message contents; (Release 7)

Etc…

Some SIP WG Documents

Internet-Drafts (total of 22 on Jan 14th 2007) E.g.:
Conference Establishment Using Request-Contained Lists in the Session Initiation Protocol (SIP) (28252 bytes)
Multiple-Recipient MESSAGE Requests in the Session Initiation Protocol (SIP) (41755 bytes)
A Framework for Consent-Based Communications in the Session Initiation Protocol (SIP) (61833 bytes)
A Framework for Session Initiation Protocol (SIP) Session Policies (66905 bytes)
Indicating Support for Interactive Connectivity Establishment (ICE) in the Session Initiation Protocol (SIP)

Request For Comments (total of 47 on Jan 14th 2007), E.g.:
The SIP INFO Method (RFC 2976) (17736 bytes)
MIME media types for ISUP and QSIG Objects (RFC 3204) (19712 bytes) updated by RFC 3459
SIP: Session Initiation Protocol (RFC 3261) (647976 bytes) obsoletes RFC 2543/ updated by RFC 3853,RFC 4320
Reliability of Provisional Responses in SIP (RFC 3262) (29643 bytes) obsoletes RFC 2543
SIP: Locating SIP Servers (RFC 3263) (42310 bytes) obsoletes RFC 2543
SIP-Specific Event Notification (RFC 3265) (89005 bytes) obsoletes RFC 2543
The Session Initiation Protocol UPDATE Method (RFC 3311) (28125 bytes)
Private Extensions to the Session Initiation Protocol (SIP) for Asserted Identity within Trusted Networks (RFC 3325) (36170 bytes)
A Privacy Mechanism for the Session Initiation Protocol (SIP) (RFC 3323) (54116 bytes)
Session Initiation Protocol Extension for Instant Messaging (RFC 3428) (41475 bytes)
Compressing the Session Initiation Protocol (RFC 3486) (24181 bytes)
The Session Initiation Protocol (SIP) Refer Method (RFC 3515) (47788 bytes)
Caller Preferences for the Session Initiation Protocol (SIP) (RFC 3841) (61382 bytes)
The Stream Control Transmission Protocol (SCTP) as a Transport for the Session Initiation Protocol (SIP) (RFC 4168) (21079 bytes)
Guidelines for Authors of Extensions to the Session Initiation Protocol (SIP) (RFC 4485) (57278 bytes)
A Mechanism for Content Indirection in Session Initiation Protocol (SIP) Messages (RFC 4483) (36794 bytes)
Enhancements for Authenticated Identity Management in the Session Initiation Protocol (SIP) (RFC 4474) (104952 bytes)

Source: http://www.ietf.org/html.charters/sip-charter.html

Some SIMPLE and XCON WG Documents

SIMPLE Internet-Drafts (total of 11 in Jan 14th, 2007): http://www.ietf.org/html.charters/simple-charter.html
The Message Session Relay Protocol (144369 bytes)
The Extensible Markup Language (XML) Configuration Access Protocol (XCAP) (168385 bytes)
Extensible Markup Language (XML) Formats for Representing Resource Lists (71157 bytes)
Session Initiation Protocol (SIP) extension for Partial Notification of Presence Information (32320 bytes)
Presence Information Data format (PIDF) Extension for Partial Presence (28348 bytes)
Session Initiation Protocol (SIP) User Agent Capability Extension to Presence Information Data Format (PIDF) (58089 bytes)
Presence Authorization Rules (64257 bytes)
Relay Extensions for the Message Sessions Relay Protocol (MSRP) (85753 bytes)
Publication of Partial Presence Information (31599 bytes)

SIMPLE Request For Comments (total of 11 in Jan 14th, 2007) :
A Presence Event Package for the Session Initiation Protocol (SIP) (RFC 3856) (62956 bytes)
A Watcher Information Event Template-Package for the Session Initiation Protocol (SIP) (RFC 3857) (46221 bytes)
A Data Model for Presence (RFC 4479) (88399 bytes)
A Session Initiation Protocol (SIP) Event Notification Extension for Resource Lists (RFC 4662) (82778 bytes)
Functional Description of Event Notification Filtering (RFC 4660) (63886 bytes)

XCON (Centralized Conferencing) Internet-Drafts (Jan 14th, 2007):
A Framework and Data Model for Centralized Conferencing (152016 bytes)
Connection Establishment in the Binary Floor Control Protocol (BFCP) (20394 bytes)
A Common Conference Information Data Model for Centralized Conferencing (XCON) (116656 bytes)

XCON Request For Comments: http://www.ietf.org/html.charters/xcon-charter.html
Requirements for Floor Control Protocol (RFC 4376) (30021 bytes)
Conferencing Scenarios (RFC 4597) (38428 bytes)
The Binary Floor Control Protocol (BFCP) (RFC 4582) (154497 bytes)

SIPPING WG documents

Source: http://www.ietf.org/html.charters/sipping-charter.html

Internet-Drafts (total of 23 in Jan14th, 2007):
The Session Initiation Protocol (SIP) and Spam (71927 bytes)
A Session Initiation Protocol (SIP) Event Package for Session-Specific Session Policies. (41547 bytes)
SIP (Session Initiation Protocol) Usage of Offer/Answer Model (32696 bytes)

Request For Comments(total of 31 in Jan14th, 2007):
Session Initiation Protocol (SIP) for Telephones (SIP-T): Context and Architectures (RFC 3372) (49893 bytes)
Short Term Requirements for Network Asserted Identity (RFC 3324) (21964 bytes)
Integrated Services Digital Network (ISDN) User Part (ISUP) to Session Initiation Protocol (SIP) Mapping (RFC 3398) (166207 bytes)
The Session Initiation Protocol (SIP) and Session Description Protocol (SDP) Static Dictionary for Signaling Compression (SigComp) (RFC 3485))
Mapping of of Integrated Services Digital Network (ISUP) Overlap Signalling to the Session Initiation Protocol (SIP) (RFC 3578)
Session Initiation Protocol PSTN Call Flows (RFC 3666) (200478 bytes)
Session Initiation Protocol Basic Call Flow Examples (RFC 3665) (163159 bytes)
Authentication, Authorization and Accounting Requirements for the Session Initiation Protocol (RFC 3702) (31243 bytes)
Best Current Practices for Third Party Call Control in the Session Initiation Protocol (RFC 3725) (77308 bytes)
Using E.164 numbers with the Session Initiation Protocol (SIP) (RFC 3824) (36535 bytes)
A Message Summary and Message Waiting Indication Event Package for the Session Initiation Protocol (SIP) (RFC 3842) (36877 bytes)
A Framework for Conferencing with the Session Initiation Protocol (SIP) (RFC 4353) (67405 bytes)
Requirements for Consent-Based Communications in the Session Initiation Protocol (SIP) (RFC 4453) (16833 bytes)
Interworking between the Session Initiation Protocol (SIP) and QSIG (RFC 4497) (149992 bytes)
Guidelines for Usage of the Session Initiation Protocol (SIP) Caller Preferences Extension (RFC 4596) (82954 bytes)
A Session Initiation Protocol (SIP) Event Package for Key Press Stimulus (KPML) (RFC 4730) (120186 bytes)

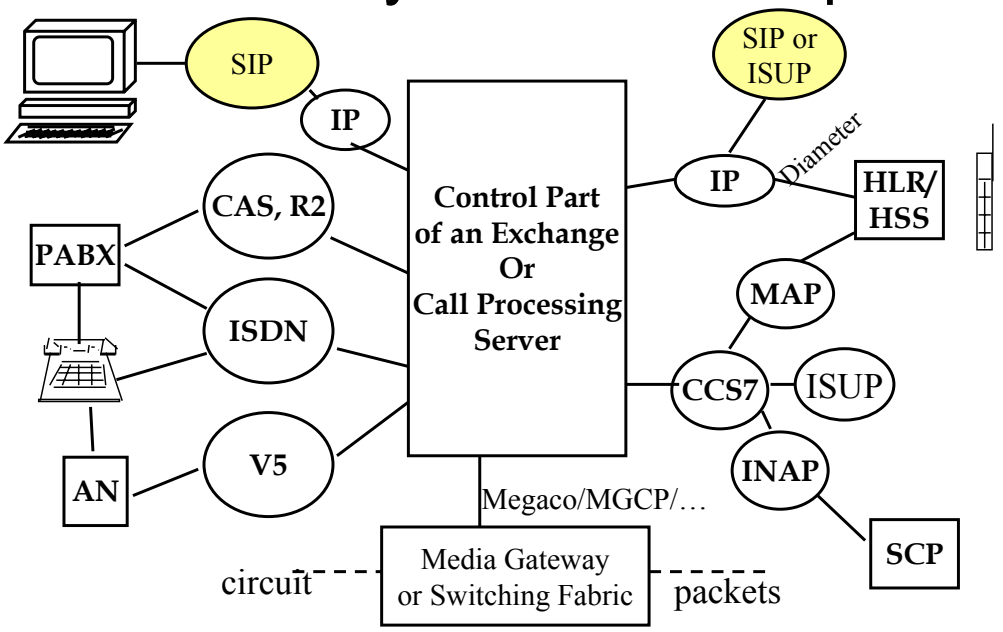
Some mmusic Documents

http://www.ietf.org/html.charters/mmusic-charter.html

Internet-Drafts (total of 10 on Jan 14th, 2007):
SDPng Transition (40981 bytes)
Real Time Streaming Protocol 2.0 (RTSP) (442431 bytes)
Interactive Connectivity Establishment (ICE): A Methodology for Network Address Translator (NAT) Traversal for Offer/Answer Protocols
TCP Candidates with Interactive Connectivity Establishment (ICE) (34761 bytes)
A Session Description Protocol (SDP) Offer/Answer Mechanism to Enable File Transfer (74554 bytes)
SDP Capability Negotiation (74278 bytes)

Request For Comments (21 current total on Jan 14th, 2007):
Real Time Streaming Protocol (RTSP) (RFC 2326) (195011 bytes)
Session Announcement Protocol (RFC 2974) (40129 bytes)
An Offer/Answer Model with SDP (RFC 3264) (60854 bytes) obsoletes RFC 2543
Session Description Protocol (SDP) Offer/Answer Examples (RFC 4317) (32262 bytes)
A Framework for the Usage of Internet Media Guides (IMGs) (RFC 4435) (51687 bytes)
Requirements for Internet Media Guides (IMGs) (RFC 4473) (53864 bytes)
SDP: Session Description Protocol (RFC 4566) (108820 bytes) obsoletes RFC 2327,RFC 3266
Session Description Protocol (SDP) Format for Binary Floor Control Protocol (BFCP) Streams (RFC 4583) (24150 bytes)

Summary of course scope



SIP Requirements and fundamentals

- Part of IETF toolkit
  - Reusing other protocols & mechanisms: HTTP, etc.
  - Flexible
  - Extensible
- Moves intelligence to End System entities
  - End-to-end protocol
- Interoperability
- Scalability (although some state in network)
- Service creation easy
- URLs and Addresses are reused
- Same routing as SMTP
- Reuses infrastructure (all applications will use SIP entities for different services)

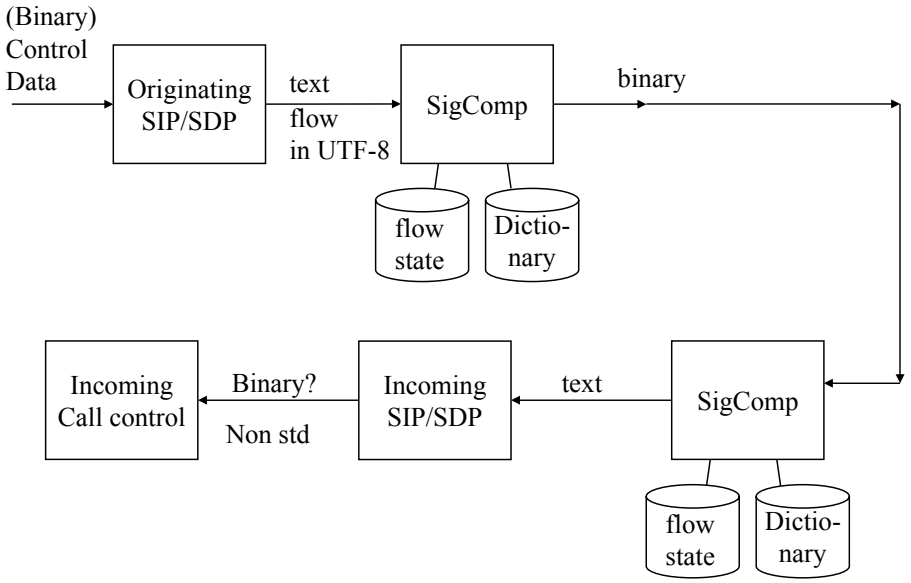
SIP overview

- Simplicity
  - Ascii based - simple tools for development but long messages
  - Lower call setup time than in H.323 (?)
  - basic protocol + extensions structure adopted
- Caller preferences, Ability to support many media types
- Runs over UDP or TCP (or SCTP)
- Used between both service and call control entities
- Has been adopted for 3G IP Multimedia signaling
- Originally subscriber signaling, in 3G also network to network signaling
- A lot of development during the last 3…5 years!
- Market view: there are 3 important variants of SIP: 3G, IETF and Microsoft Messanger.

SigComp allows cmpression of Signaling Messages

- RFC 3320 and RFC 3321 specify a layer between the signaling transport and the signaling application
- Uses Global and User Specific Dictionaries to store state data over many SIP sessions
  - Dictionary is enlarged dynamically
  - Efficiency is improved by using all previous messages in a flow as source of the dictionary – incremental application of compression – this requires quite a lot of memory
  - After dictionary replacement, Huffman coding can be applied.
- Overall Compression/decompression architecture is based on a bytecode driven Universal Decompression Virtual Machine
  - Bytecode can be sent in SigComp messages by the Compressior
  - leaves a lot of detail for the implementor

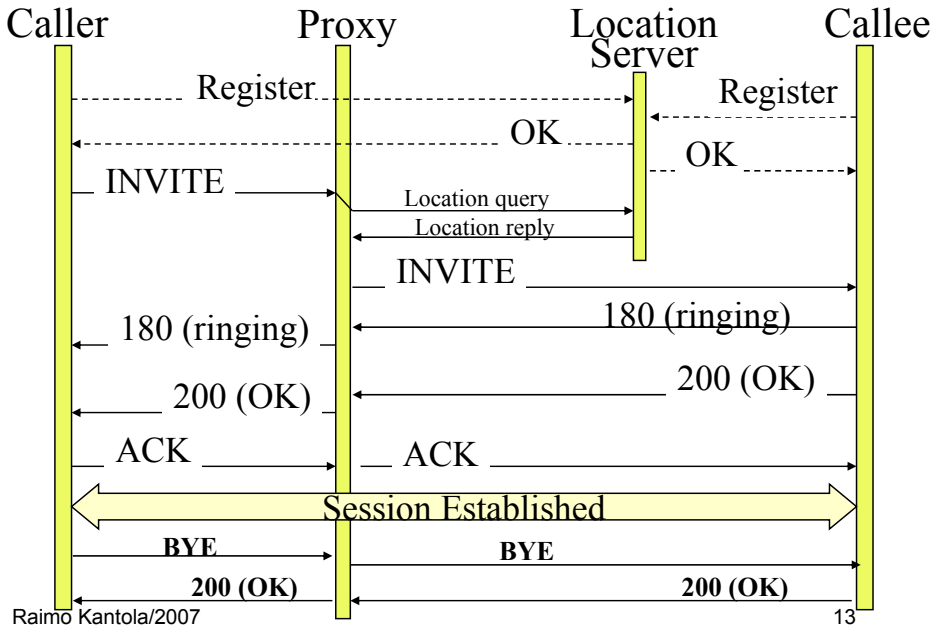
Signaling compression model



Sip Entities

- User Agents
  - Can act as client and as server
- Servers:
  - Redirect Servers
    - Send back alternative location of the user (similar as HTTP servers)
  - Proxy servers
    - Act on behalf of client (forwards requests)
    - Forking proxies
  - Registrars
    - Accepts registrations and maps public SIP URIs to user locations.
  - Location Servers (not part of SIP architecture)
    - Gives back location of user (received from registrars)
    - E.g. HSS in 3GPP IMS architecture
    - Protocol between Location server and SIP server not defined by SIP specs (e.g. LDAP)

Basic SIP call setup and release



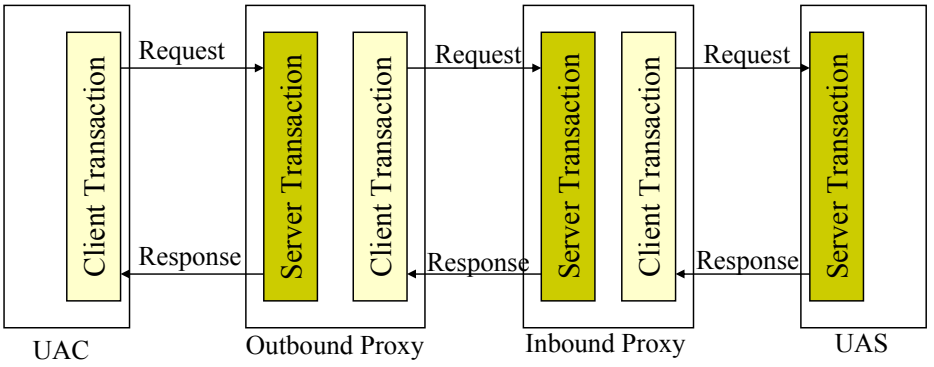
”Basic call” Example (IETF)

- Caller sends INVITE
- Callee can accept, reject, forward the call
- If the callee accepts the call: responds with an optional provisional (1xx), and a final (≥200) response
- The caller confirms final response via ACK
- Conversation
- Caller or callee sends BYE
- BYE is acknowledged by 200 OK
- Low call setup times, post dial delay: 1.5 RTT !

SIP messages have headers and a body

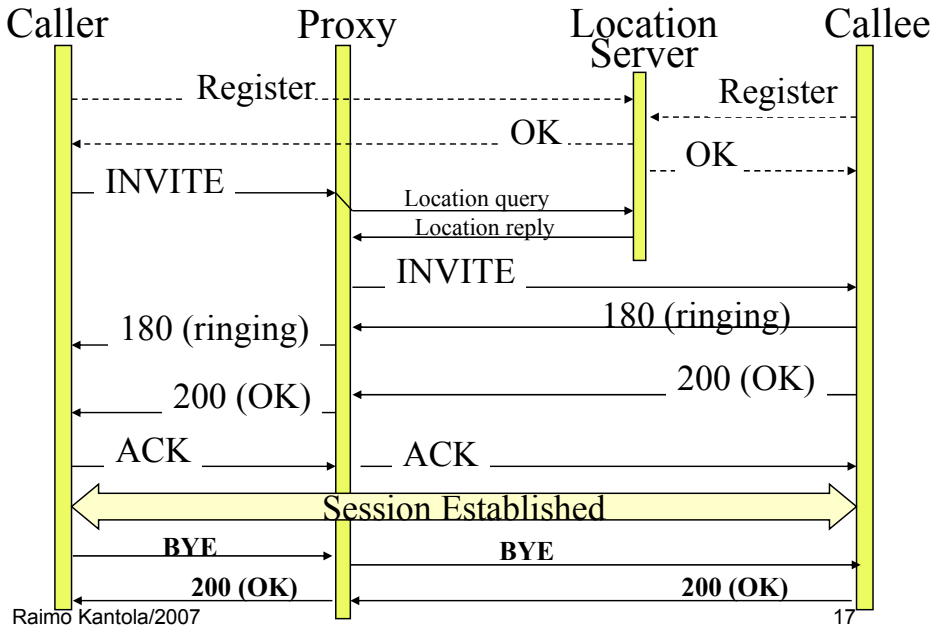
- Headers carry control information and are processed e.g. by Proxies
- Body can be e.g. SDP – session description protocol
  - end-to-end information (cmp H.245) describing session requirements e.g. coding methods, etc
- Message delivery is transaction oriented= have request + reply: e.g INVITE+200 OK

User Agent is split into User Agent Client (UAC) and User Agent Server(UAS)



- All Communication follows this transaction model, except 2xx and ACK.
- Server transactions filter incoming requests, absorbing retransmissions
- Only UAS can generate 2xx and only UAC can generate ACK (to success).

Basic SIP call setup and release



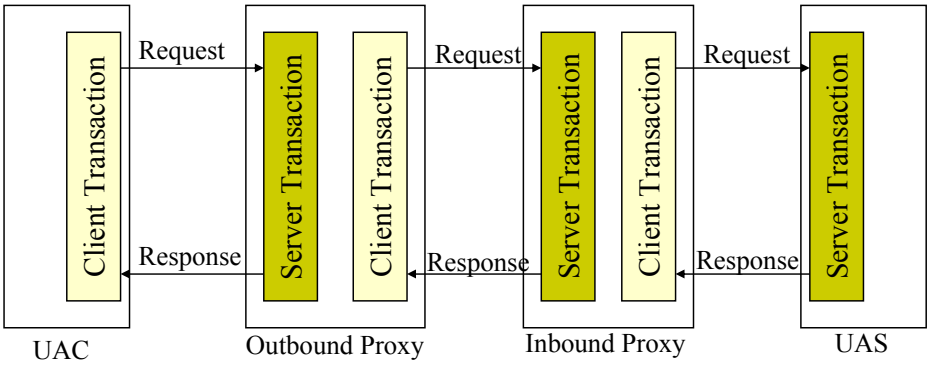
”Basic call” Example (IETF)

- Caller sends INVITE
- Callee can accept, reject, forward the call
- If the callee accepts the call: responds with an optional provisional (1xx), and a final (≥200) response
- The caller confirms final response via ACK
- Conversation
- Caller or callee sends BYE
- BYE is acknowledged by 200 OK
- Low call setup times, post dial delay: 1.5 RTT !

SIP messages have headers and a body

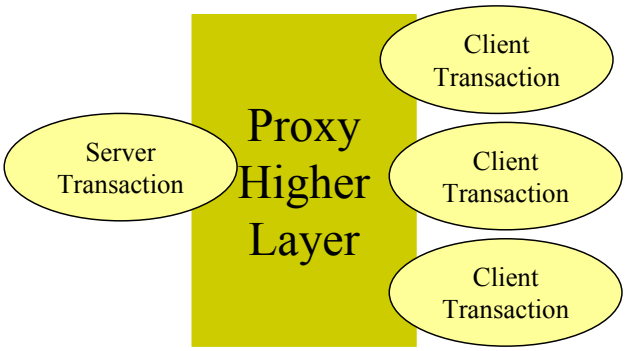
- Headers carry control information and are processed e.g. by Proxies
- Body can be e.g. SDP – session description protocol
  - end-to-end information (cmp H.245) describing session requirements e.g. coding methods, etc
- Message delivery is transaction oriented= have request + reply: e.g INVITE+200 OK

User Agent is split into User Agent Client (UAC) and User Agent Server(UAS)



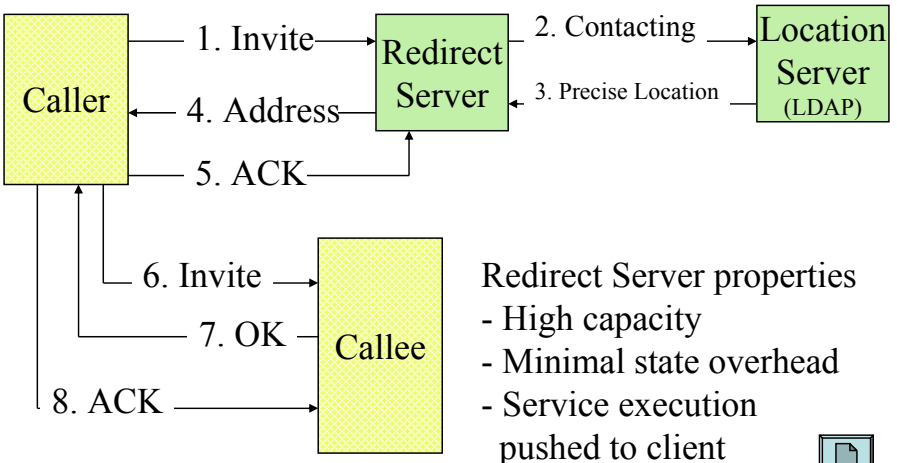
- All Communication follows this transaction model, except 2xx and ACK.
- Server transactions filter incoming requests, absorbing retransmissions
- Only UAS can generate 2xx and only UAC can generate ACK (to success).

A Stateful Proxy can fork a transaction



Forking = multicast of INVITEs to N addresses

Redirect Server pushes processing to clients



Redirect Server properties
- High capacity
- Minimal state overhead
- Service execution pushed to client

17.3.04

Stateful Proxy vs Stateless Proxy

- Maintains call context
- Replicates UAS/UAC to process requests and responses
- Call state and transaction state can be maintained
- Forking proxies require state
- TCP proxies must be stateful for reliability
- Enhanced services require state
- Can collect charging info
- All transactions in a session are processed by the same proxy (computer)

- No call context
- Response is not based on UA replication
- Provides client anonymity
- Restricted gateway access
- High processing capacity
- Easier to replicate than the stateful proxy
- Consecutive transactions in a session can be processed on different computers → load sharing is easy to arrange for e.g. x00 million users
- Also semi-stateful is possible

UA = User Agent, UAC = UA Client
UAS = UA Server

Full list of SIP methods

ACK	Acknowledges the establisment of a session
BYE	Terminates a session
CANCEL	Cancels a pending request
INFO	Transports PSTN telephony signaling
INVITE	Establishes a session
NOTIFY	Notifies a User Agent of a particular event
OPTIONS	Queries a server about its capabilities
PRACK	Acknowledges a provisional response
PUBLISH	Uploads information to a server
REGISTER	Maps a public URI with the current location of the user
SUBSCRIBE	Requests to be notified about a particular event
UPDATE	Modifies some characteristic of a session
MESSAGE	Carries an instant message
REFER	Instructs a server to send a request

Blue methods are candidates for AS processing
includes both the base protocol and current(2004) extensions

Notes on SIP methods: INVITE

- Requests users to participate in a session. Body contains description of the session. If someone wants to modify the parameters of the session, he/she must re-INVITE carrying the modified parameters.
- One or more Provisional and one Final response are expected.

Notes on SIP methods: ACK

- Acknowledges the Final Response to INVITE even if INVITE was cancelled → result is 3-way handshake: INVITE-final-resp – ACK.
- Proxies can only ACK non-successful Final Resp.
- Purpose:
  - Let's the server know that session establishment was successful.
  - Forking may result in many final responses. Sending ACKs to every destination that sent a final response is essential to ensure working over UDP.
  - Allows sending INVITEs without session description. In this case the description is postponed to ACK.
- Has the same Cseq as the INVITE it acknowledges (see later for SIP headers).

Notes on SIP methods: CANCEL

- Purpose: to cancel pending transactions. Will be ignored by completed transactions = final response already sent.
  - useful for forking proxies. If one destination answered, the forking proxy can cancel all other pending INVITEs.
- Has the same Cseq as the request it calcels (see later for SIP headers).

Notes on SIP methods: REGISTER

- Purpose: to register the user's current location.
- A user can be registered in several locations at the same time. Forking is used to find out where the user wants to answer the session invitation.
- A user can register from anywhere to his registrar → provides mobility.

Notes on SIP methods: OPTIONS

- UA can query a server: which
  - methods and
  - extensions and
  - which session description protocols it supports.
  - which encoding for message bodies the server understands (e.g. compression to save bw).

Some SIP issues

- Parties can release the “call session” but since they have obtained each others IP-addresses, they can continue sending media streams to each other!!
- How to push INVITE to B-party, if B-party does not have a permanent IP address which is most often the case!

} Integration of Proxy with Firewall and NAT

- Response messages (e.g. 180) are not reliably delivered. This may cause tear down of the call if it was initiated from ISDN

PRACK method

- If BYE is lost, Proxy does not know that call has ended

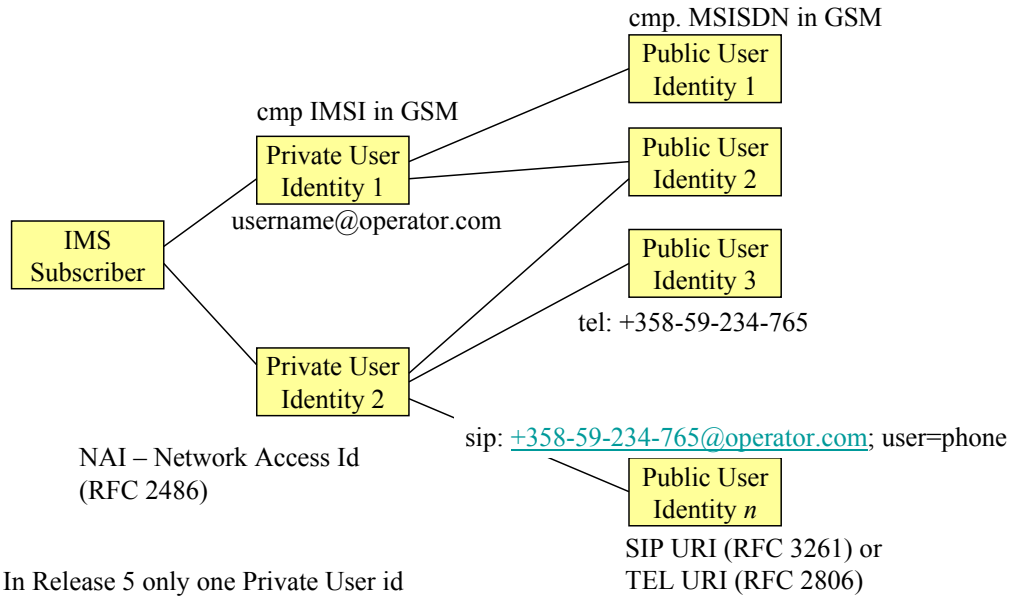
KeepAlive = re-INVITE mechanism

- Text based coding increases the signaling overhead → problem in Radio access

Identification of users

- **sip:user@host[parameters][headers]**
- SIP URIs are like URLs, with prefix sip: which gives schema
  - sip:joe.smith@hut.fi
  - sip:joe.smith@hut.fi?subject=Protocol
  - sip:sales@hotel.xy;geo.position:=48.54_-123.84_120
- Address must include host, other parameters are optional (username, port, etc…)
- Email-addresses can be reused
- “Click-to-call” on web-pages, MM messages, etc… are easily implemented

Identification of users in 3G IMS in R6



In Release 5 only one Private User id

Requests invoke SIP methods

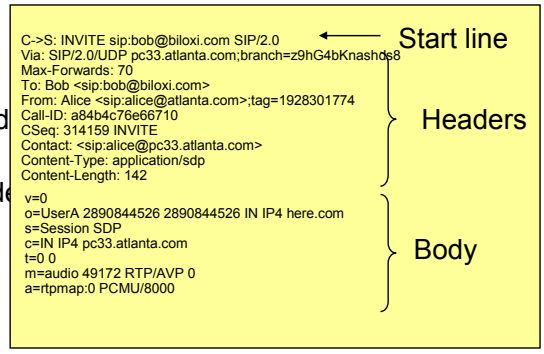
- SIP methods are invoked on servers when requests arrive:
  - A REGISTER request sends location information of users to Registrars, registers with the location service
  - An INVITE request invites a user to participate in a session or conference
    - The message body contains a description of the session (usually SDP)
  - ACK requests are used to confirm responses for INVITE, for reliable message exchanges
  - CANCEL requests cancel the pending request of the session
  - BYE requests are used to terminate active sessions
    - Any party of the session can send it
  - OPTIONS requests are used to query information about servers' capabilities
  - PRACK requests are used to confirm provisional responses

SIP responses are classified by first digit

- HTTP look-alike
- Hierarchically organized three digit codes: status code - text associated with the code
- Provisional and final responses:
  - 1xx responses are informational messages e.g., 180 Ringing
  - 2xx response shows a successful transaction e.g., 200 OK
  - 3xx responses are redirect messages e.g., 301 Moved Permanently
  - 4xx responses indicate errors in requests e.g., 400 Bad Request
  - 5xx responses indicate server errors e.g., 500 Version not supported
  - 6xx responses indicate global failures e.g., 600 Busy everywhere

SIP Message Format

- START-LINE
  - SIP version used
  - In requests: address and method used
  - In responses: status code
- HEADERS
  - Information about call
- BODY (payload)
  - Usually SDP message



A Request line contains the Request URI = the name of the user that is the destination. This request URI is used for SIP routing.

In this example a typical INVITE message from Alice to Bob is shown. Alice is registered at domain atlanta.com, and has sip address sip:alice@atlanta.com.

Alice tries to invite Bob at his sip address bob@biloxi.com

Headers will be explained later. It is important to know that there are several mandatory headers in every SIP message. Headers always end with Content-Lenght header, that shows how long is the body of the message (in bytes)

In this example, body of the message is SDP (Session Description Protocol). It gives the description of the session that Alice wants to establish (types of codecs, session parameters, etc).

To and From header fields

- **To: specifies the logical call destination**
- **From: specifies the logical call source**
- **Present in all SIP messages**

```
INVITE sip:bob@biloxi.com SIP/2.0
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKnashds8
Max-Forwards: 70
To: Bob <sip:bob@biloxi.com>
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 314159 INVITE
Contact: <sip:alice@pc33.atlanta.com>
Content-Type: application/sdp
Content-Length: 142
```

NB: In IMS To and From header field are end to end information and not used nor asserted by the network. Instead the P-CSCF inserts the **P-Asserted-Identity** header field into the SIP messages indicating the network asserted identity of the originator of the call

To: header specifies the logical call destination. In other words, this is the recipient's address. Optional parameter "display-name" (in this case "Bob") is just for human-user interface, while sip address is sip:bob@biloxi.com. The value of To –field is not used for SIP routing, rather it is used for filtering at the destination and for human consumption. A tag –parameter can be appended to distinguish different UAs that are identified by the same URI.

From: header is the logical address of the originator of the request. "Logical" address means that it does not necessarily mean that it is the current address where Alice is reachable. In this case, Alice registered herself, and is contactable on sip:alice@pc33.atlanta.com. But Alice also has her permanent logical sip address sip:alice@atlanta.com. This value can be used e.g. for filtering purposes by the callee UA. Proxy server will forward request to her logical addresses to her current address.

"tag" parameter is used in 'To' and 'From' header fields. Tag is used to identify a dialog* between parties. The dialog is identified by combination of tags from both parties (in 'To' and in 'From' heders), plus Call-ID parameter.

In this example, Alice's User Agent added 'tag=1928301774' (which is unique value, created by Alice's User Agent, and it is always different)

Call-ID and CSeq header fields

- **Call-ID: It helps to uniquely identify a particular SIP dialog or registration**
  - It helps to match requests and responses
  - It helps to detect duplicated messages
- **CSeq: It is a number that uniquely identifies the transaction in a call**
- **Present in all SIP messages**

```
INVITE sip:bob@biloxi.com SIP/2.0
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKnashds8
Max-Forwards: 70
To: Bob <sip:bob@biloxi.com>
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 314159 INVITE
Contact: <sip:alice@pc33.atlanta.com>
Content-Type: application/sdp
Content-Length: 142
```


Call-ID header consists of a locally unique number (generated by the User Agent) and @domain parameter, making it globally unique. The algorithm for making locally unique numbers is implementation issue. There are specifications available how to make algorithms in a way that will give unique numbers at the end.

Call-ID, together with tags in From: and To: headers identifies particular dialog. (see previous slides's notes for details)

Cseq: header contains single decimal sequence number and the request method. This number helps to order thransactions within one dialog, and to help differentiate between retransmissions and new requests. For example, callee can receive one request with CSeq:31 INVITE. Callee may answer to this request (e.g. with "200 OK" response). If callee receives again INVITE with same CSeq: 31 INVITE, it will know that its response was lost. It is also sure that this is not reinvitation, because the CSeq number is the same.

Content-Type and Content-Length header fields

- **Content-Type: It describes the media type of the message body**
- **Content-Length: The number of octets in the message body**
  - It is mandatory in all SIP messages.

```
INVITE sip:bob@biloxi.com SIP/2.0
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKnashds8
Max-Forwards: 70
To: Bob <sip:bob@biloxi.com>
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 314159 INVITE
Contact: <sip:alice@pc33.atlanta.com>
Content-Type: application/sdp
Content-Length: 142
```


Content-Type header gives the media type of the body of the message. It must be present if there is some data in the body. Content-Type is header reused from HTTP and it has same values, e.g. text/html for html body or application/sdp for SDP.

Content-Length gives the size of the body. It does not include CRLF separating header fields and body (headers and body are always separated with one blank line).

If there is no body in the message, then Content-Length is set to 0.

Max-Forwards

- **Max-Forwards field must be used with any SIP method**
- **It limits the number for proxies or gateways on the way of SIP message to the destination.**

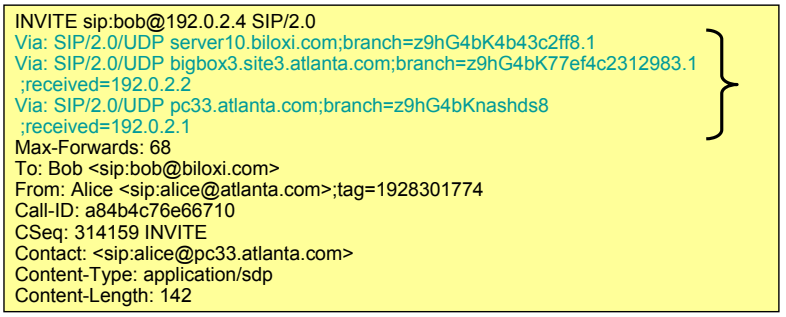
```
INVITE sip:bob@biloxi.com SIP/2.0
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKnashds8
Max-Forwards: 70
To: Bob <sip:bob@biloxi.com>
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 314159 INVITE
Contact: <sip:alice@pc33.atlanta.com>
Content-Type: application/sdp
Content-Length: 142
```


This field is used to prevent looping of the message. Every proxy that accepts the message will decrease this field by 1. If it reaches 0, then message is discarded, and message ”483 Too Many Hops” is sent to originator.

Every User Agent Client must insert Max-Forwards header field into each request. Value to be set is recommended to be 70. This number is big enough to prevent message being discarded unnecessarily (if message cannot reach destination in 70 hops, it is assumed that something is not configured properly anyway).

VIA header indicates path taken by the request so far

- **Branch parameter is used to detect loops**
- **Contains transport protocol, client's host name and possibly port number, and can contain other parameters**



Via: header is added by every entity that the request passes. It also determines the path for the response.

Every relay on the path of the request will put its address in the topmost Via: header.

In the response, when a relay receives the message, it will check if the topmost Via: header is its address. It can determine what is particular call by branch parameter.

It will then examine where to send response back (topmost Via: header value, after removing Via: with its own address)

Generally, for the requests every proxy on the way adds one Via: header with its address as value.

In sending response back, Via: header value is used for routing the response back.

Record-route and Route

- **Record-Route: header is added by proxy, when proxy wants to stay in the route of all sip messaging**
- **Route is added by User Agent Client, after response came, with all Record-route headers in it (then UAC knows which relays want to stay on the signaling path)**
- **NOT the same as Via: headers**

```
INVITE sip:callee@u2.domain.com SIP/2.0
Contact: sip:caller@u1.example.com
Record-Route: <sip:p2.domain.com;lr>
Record-Route: <sip:p1.example.com;lr>
```

Inserted by proxies p1.example.com and p2.example.com.

```
BYE sip:callee@u2.domain.com SIP/2.0
Route: <sip:p1.example.com;lr>,<sip:p2.domain.com;lr>
```

UA can specify through which proxies this message must go

(explanation from RFC 3261) U1 sends:

```
INVITE sip:callee@domain.com SIP/2.0
Contact: sip:caller@u1.example.com
```

to P1. P1 is an outbound proxy. P1 is not responsible for domain.com, so it looks it up in DNS and sends it there. It also adds a Record-Route header field value:

```
INVITE sip:callee@domain.com SIP/2.0
Contact: sip:caller@u1.example.com
Record-Route: <sip:p1.example.com;lr>
```

P2 gets this. It is responsible for domain.com so it runs a location service and rewrites the Request-URI. It also adds a Record-Route header field value. There is no Route header field, so it resolves the new Request-URI to determine where to send the request:

```
INVITE sip:callee@u2.domain.com SIP/2.0
Contact: sip:caller@u1.example.com
Record-Route: <sip:p2.domain.com;lr>
Record-Route: <sip:p1.example.com;lr>
```

The callee at u2.domain.com gets this and responds with a 200 OK:

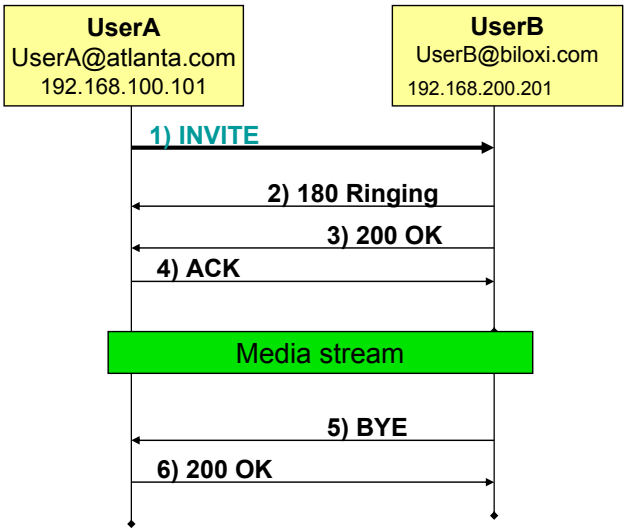
```
SIP/2.0 200 OK
Contact: sip:callee@u2.domain.com
Record-Route: <sip:p2.domain.com;lr>
Record-Route: <sip:p1.example.com;lr>
```

The callee at u2 also sets its dialog state's remote target URI to sip:caller@u1.example.com and its route set to:

(<sip:p2.domain.com;lr>,<sip:p1.example.com;lr>)

This is forwarded by P2 to P1 to U1 as normal. Now, U1 sets its dialog

”Basic Call” call flow



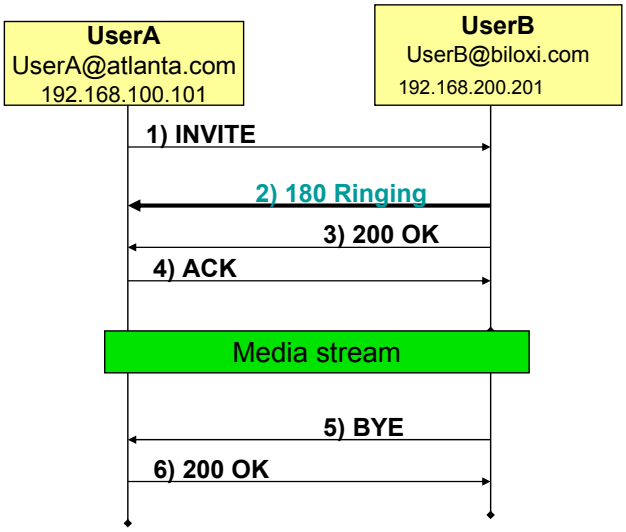

```
   INVITE sip:UserB@biloxi.com SIP/2.0
   Via: SIP/2.0/UDP
client.atlanta.com:5060;branch=z9hG4bK74bf9
   Max-Forwards: 70
   From: BigGuy <sip:UserA@atlanta.com>;tag=9fxced76sl
   To: LittleGuy <sip:UserB@biloxi.com>
   Call-ID: 3848276298220188511@atlanta.com
   CSeq: 1 INVITE
   Contact: <sip:UserA@192.168.100.101>
   Content-Type: application/sdp
   Content-Length: 147

   v=0
   o=UserA 2890844526 2890844526 IN IP4 client.atlanta.com
   s=-
   c=IN IP4 192.168.100.101
   t=0 0
   m=audio 49172 RTP/AVP 0
   a=rtpmap:0 PCMU/8000
```

In this example, UserA sends INVIT to UserB. In the initial INVITE UserA puts Max-Forwards 70. In Contact: header UserA puts his address where I can be reached in the moment.

In SDP body UserA specifies what kind session he wants to establish. In this case it would be audio session on po 49172 (RTP over UDP as transport) with PCM μ coding with 8000 samples per second.

”Basic Call” call flow

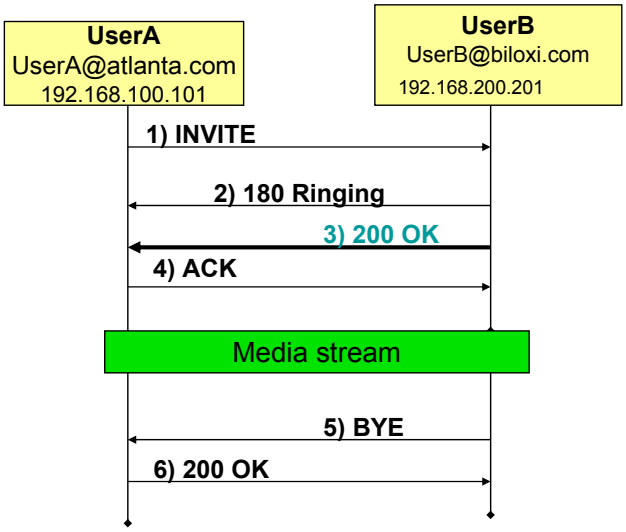



```
   SIP/2.0 180 Ringing
   Via: SIP/2.0/UDP
client.atlanta.com:5060;branch=z9hG4bK74bf9
    ;received=192.168.100.101
   From: BigGuy <sip:UserA@atlanta.com>;tag=9fxced76sl
   To: LittleGuy <sip:UserB@biloxi.com>;tag=8321234356
   Call-ID: 3848276298220188511@atlanta.com
   CSeq: 1 INVITE
   Content-Length: 0
```

UserB answers with 180 Ringing provisional answer. UserB also adds ta to To: header, but maintains Call-ID a CSeq

Note that From: and To: fields are not changed. That is because UserB acts as Server (sending responses back), a call is started by UserA (in From: field). Order will be preserved untill transaction is complete

”Basic Call” call flow



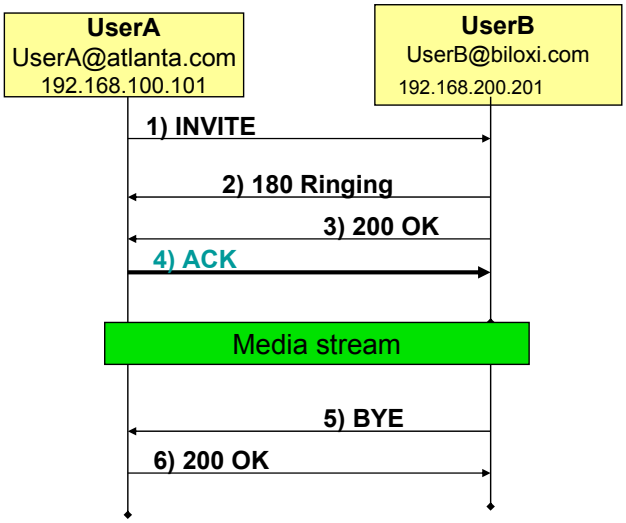

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP
client.atlanta.com:5060;branch=z9hG4bK74bf9
 ;received=192.168.100.101
From: BigGuy <sip:UserA@atlanta.com>;tag=9fxced76sl
To: LittleGuy <sip:UserB@biloxi.com>;tag=8321234356
Call-ID: 3848276298220188511@atlanta.com
CSeq: 1 INVITE
Contact: <sip:UserB@192.168.200.201>
Content-Type: application/sdp
Content-Length: 145

v=0
o=UserB 2890844527 2890844527 IN IP4 client.biloxi.com
s=-
c=IN IP4 192.168.200.201
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

UserB now sends 200 OK final response. He also adds Contact: field (where he could be reached).

In SDP body of the message UserB gives his preference for session establishment (audi RTP over UDP, por 3456, PCM μ modulation with 80 samples per second and also his IP address in c= field.

”Basic Call” call flow

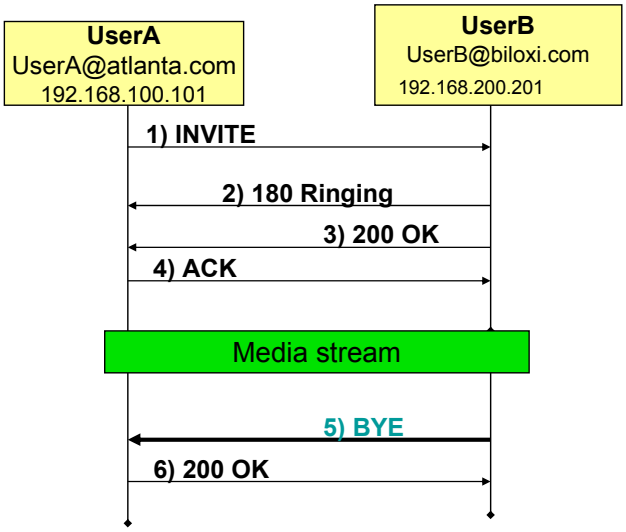



```
   ACK sip:UserB@biloxi.com SIP/2.0
   Via: SIP/2.0/UDP
client.atlanta.com:5060;branch=z9hG4bK74bf9
   Max-Forwards: 70
   From: BigGuy <sip:UserA@atlanta.com>;tag=9fxced76sl
   To: LittleGuy <sip:UserB@biloxi.com>;tag=8321234356
   Call-ID: 3848276298220188511@atlanta.com
   CSeq: 1 ACK
   Content-Length: 0
```

UserA acknowledge OK response. After ACK media session can be established, using parameters given in the handshake before (INVITE and in 200 OK response)

CSeq header value still 1 (because this part of the same transaction), but method name has been changed to ACK

”Basic Call” call flow

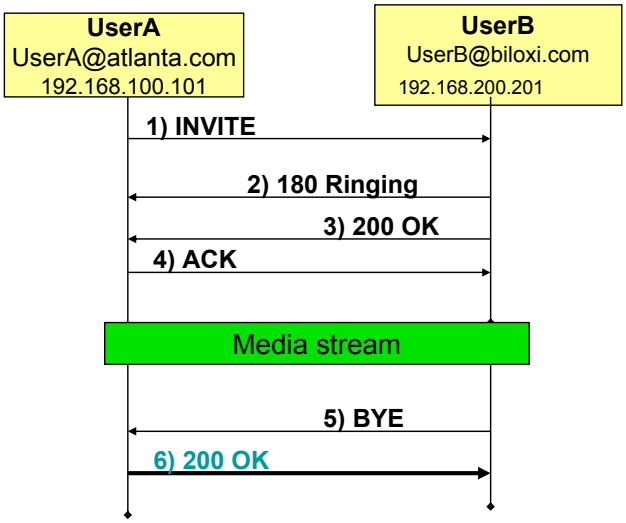



```
BYE sip:UserA@192.168.100.101 SIP/2.0
  Via: SIP/2.0/UDP
client.biloxi.com:5060;branch=z9hG4bKnashds7
  Max-Forwards: 70
  From: LittleGuy <sip:UserB@biloxi.com>;tag=8321234356
  To: BigGuy <sip:UserA@atlanta.com>;tag=9fxced76sl
  Call-ID: 3848276298220188511@atlanta.com
  CSeq: 121 BYE
  Content-Length: 0
```

After the conversation UserB wishes to end the call. It sends BYE request to UserA.

Notice that CSeq of UserB has totally different sequence. That is because UserB maintains its own CSeq numbering independently.

”Basic Call” call flow




```
  SIP/2.0 200 OK
   Via: SIP/2.0/UDP
client.biloxi.com:5060;branch=z9hG4bKnashds7
    ;received=192.168.200.201
   From: LittleGuy <sip:UserB@biloxi.com>;tag=8321234356
   To: BigGuy <sip:UserA@atlanta.com>;tag=9fxced76sl
   Call-ID: 3848276298220188511@atlanta.com
   CSeq: 121 BYE
   Content-Length: 0
```

UserA sends 200 O
and conversation is
ended. Note that
method name (BYE
in CSeq header
helps UserB to kno
that UserA sent OK
for BYE request.

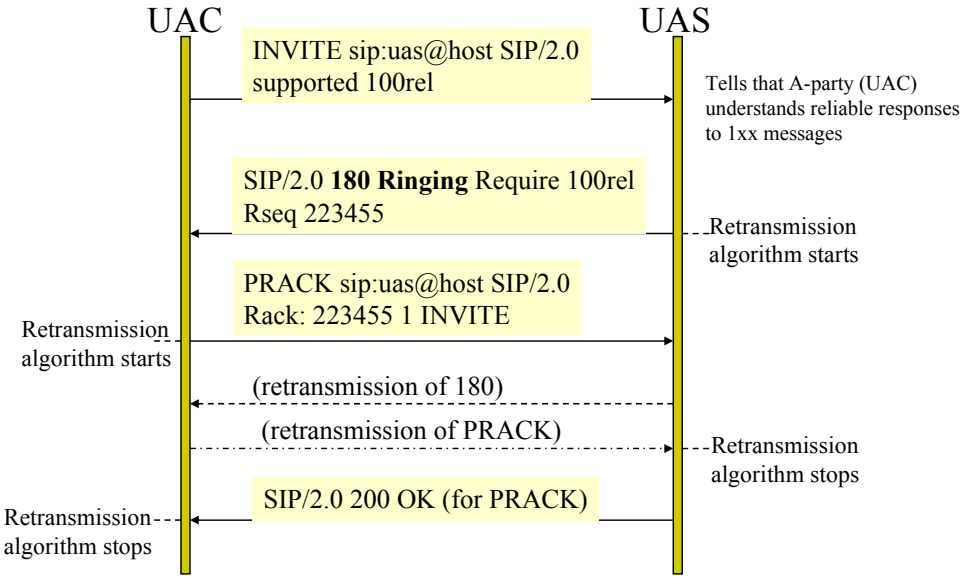
SIP Extensions

- Needed to satisfy additional requirements
- Must conform to design rules
- SIP is not intended to solve every problem (another protocol might be used instead)

Feature Negotiation (OPTIONS)

- *Supported* features can be specified in request and response
  - **Supported** UAC and UAS tell features they support
- *Required* features can be specified in request and response
  - **Require** UAC tells UAS about required options
  - **Proxy-Require** required options for proxy/redirect servers
  - Many extensions use Require and Proxy-Require to specify their support
- New methods can be added without changing the protocol
  - server can respond with **405 Not Supported**
  - returns list of supported methods in **Allow** header
  - client can ask which methods are supported using OPTIONS

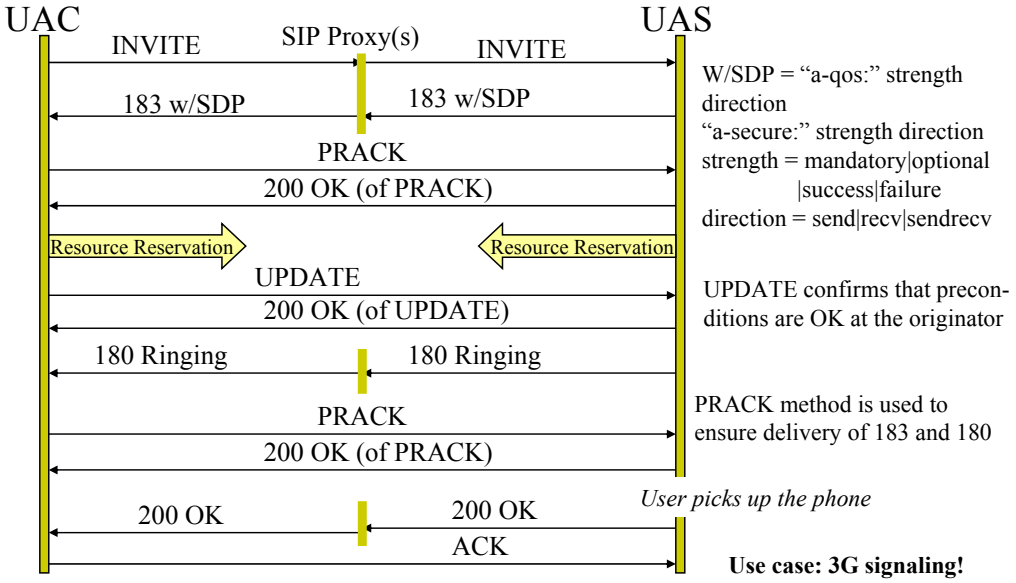
Reliable Provisional response in SIP



QoS support - UPDATE

- Usage rule for 183-Session-Progress
  - If “a=qos” appeared in SDP, UAS sends 183 with “Session: qos” and SDP
- Additional Method - UPDATE
  - If “a=qos” appeared in SDP with “confirm” attribute, UAS/UAC sends UPDATE with success/failure status of each precondition.
  - 200 OK must acknowledge the UPDATE message
  - user B does not need to be prompted
- Additional Status Response - 580 Precondition Failure
  - If a mandatory precondition can’t be met, UAS terminates INVITE with this status response

Phone should not ring before QoS and Security are OK

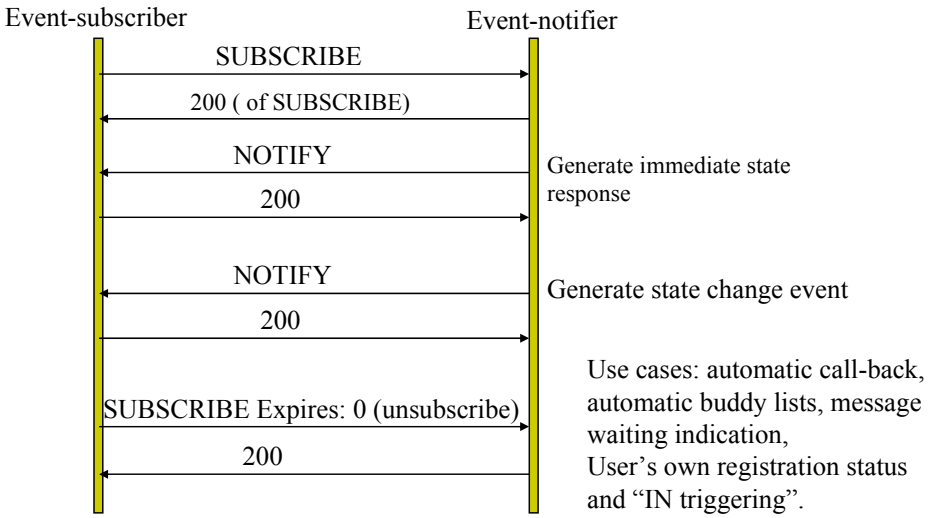


SDP = Session Description Protocol (carried in SIP message body)

For IMS the this use case is similar to what is found in RFC 3312 "Integration of Resource management and SIP.

This approach allows finding out whether the callee is reachable before any resources are dedicated for the call and thus saving radio resources at the originating end for a period that can not be billed. When ringing is sent by the callee, preconditions are OK also at the callee, i.e. the radio resources for the media plane have been reserved.

SIP event notifications tell about remote significant events to the local party

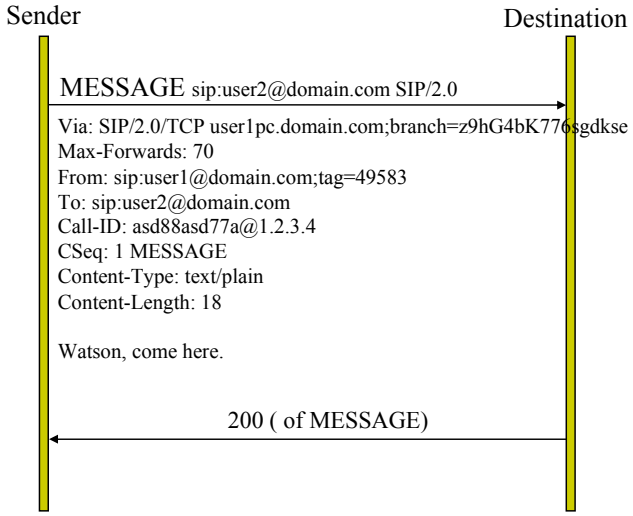


Automatic call-back = call completion on busy. A calls to a busy subscriber, subscribes to the state change of the callee. When callee becomes free, the caller is notified of the the state change. Caller INVITEs the callee again, now callee is likely to be free and the call can be established. It is important that the call is always established from the paying customer to the other party.

Message wayting indication: a user subscribes to the state of his voice mail or e-mail account. When a message arrives, the user is immediately notified that messages are waiting and the user can retrieve them at a convenient time.

A use case: A mobile customer wants to see whether she is registered in IMS in real time (cmp whether GSM phone is connected to a network). The registration may be terminated for administrative reasons at any time, a network node may die and break the registration. To have up to date information on her registration status, the UA needs to subscribe to the reg –event package at the S-CSCF that also acts as the registrar in IMS. To do this, always after the registration at S-CSCF of the home network, the UA will send the Subscribe message to the S-CSCF. Moreover, also the P-CSCF wants to have the same information. Therefore also the P-CSCF will subscribe to the same event package for all of the visiting users.

SIP MESSAGE provides Instant Messaging capability in Pager mode



Note:
-body is MIME or S/MIME
- there is no dialog!

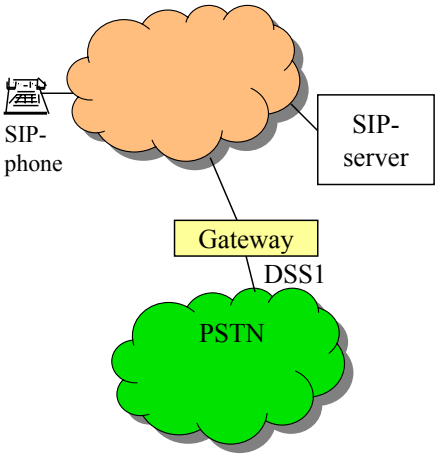
Use example:
-maps to SMS in GSM

Issue:
- authentication of sender and charging! Must use Proper security features

More SIP extensions

- MESSAGE
  - For instant messaging
- INFO
  - To transport mid-session information (very useful in SIP-PSTN gateways to carry all PSTN messages across SIP domains such that do not easily map to any other known SIP message)
- Automatic configuration
  - DHCP or Service Location Protocol (SLP)
- Caller Preferences
  - New headers: Accept-Contact, Reject-Contact, Request-Disposition (e.g. to express a preference for contacting the user at “fixed”, “business” connection).
- REFER
  - For session transfer (Refer-To: and Referred-By:)
- …

Deployment example: Elisa's experimental service for BB customers



- SIP –server recognizes a numbering block, connects calls directly from IP-phone to IP-phone in the block
- Calls to all other numbers are routed to the gateway
- = SIP-server+Gateway are like a PBX

We need VOIP peering between operators to get rid of this!