Session Initiation Protocol

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

A lot of this material is based on proposals => may change quickly

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 v5.2.0 (2002-09) Signaling flows for the IP MM call control based on SIP and SDP; stage 3 (Release 5)
3GPP TS 24.229 v5.3.0 (2002-12) IP multimedia call control protocol based on SIP and SDP, Stage 3 (Release 5)
3GPP TS 29.228 v5.1.0 (2002-09) IMS Cx and Dx interfaces, Signaling flows and message contents; (Release 5)

Etc…
Summary of course scope

SIP Features

- 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 reusage
- Same routing as SMTP
- Reuses infrastructure (all applications will use SIP entities for different services)
SIP overview

• Simplicity
  – Ascii based - simple tools for development
  – 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, proposed also as network to network signaling
• A lot of development during the last 3…4 years!

SigComp allows compression 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
• Overall Compression/decompression architecture is based on a bytecode driven Universal Decompression Virtual Machine
  – Bytecode can be sent in SigComp messages by the Compressor
  – leaves a lot of detail for the implementor
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
  - 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

**Caller**

1. **Register**
2. INVITE
3. 180 (ringing)
4. 200 (OK)
5. ACK
6. BYE
7. 200 (OK)

**Proxy**

1. Register
2. OK
3. Location query
4. Location reply
5. INVITE
6. 180 (ringing)
7. 200 (OK)
8. ACK
9. Session Established

**Location Server**

1. Register
2. OK
3. INVITE
4. 180 (ringing)
5. 200 (OK)
6. ACK

**Callee**

1. Register
2. OK
3. BYE
4. 200 (OK)
”Basic call” Example

- 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.

A Stateful Proxy can fork a transaction

Forking = multicast of INVITEs to N addresses
Redirect Server pushes processing to clients

1. Invite
2. Contacting
3. Precise Location
4. Address
5. ACK
6. Invite
7. OK
8. ACK

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

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

- 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
- Also semi-stateful is possible

UA = User Agent, UAC = UA Client
UAS = UA Server
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!
- Response messages (e.g. 180) are not reliably delivered. This may cause tear down of the call if it was initiated from ISDN
- If BYE is lost, Proxy does not know that call has ended
- Ascii coding increases the signaling overhead in Radio access

Integration of Proxy with Firewall and NAT

PRACK method

KeepAlive = re-INVITE mechanism

Addressing

- **sip:user@host[parameters][headers]**
- SIP-addresses 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… is easy implemented
"Basic Call" call flow

1) INVITE
2) 180 Ringing
3) 200 OK
4) ACK
5) BYE
6) 200 OK

UserA
UserA@atlanta.com
192.168.100.101

UserB
UserB@biloxi.com
192.168.200.201

Media stream
"Basic Call" call flow

1) INVITE
2) 180 Ringing
3) 200 OK
4) ACK

Media stream

5) BYE
6) 200 OK

UserA
UserA@atlanta.com
192.168.100.101

UserB
UserB@biloxi.com
192.168.200.201
"Basic Call" call flow

1) INVITE

2) 180 Ringing

3) 200 OK

4) ACK

Media stream

5) BYE

6) 200 OK
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

---

**To and From header fields**

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

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
```
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
```

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

```
INVITE sip:bob@192.0.2.4 SIP/2.0
Via: SIP/2.0/UDP server10.biloxi.com;branch=z9hG4bK4b43c2ff8.1
Via: SIP/2.0/UDP bigbox3.site3.atlanta.com;branch=z9hG4bK77ef4c2312983.1
;received=192.0.2.2
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKnashds8
;received=192.0.2.1
Max-Forwards: 68
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
```
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 come, 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>
```

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

IU can specify through which proxies this message must go

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

- UAC
  - INVITE sip:uas@host SIP/2.0 supported 100rel
  - SIP/2.0 180 **Ringing** Require 100rel
    - Rseq 223455
  - PRACK sip:uas@host SIP/2.0
    - Rack: 223455 1 INVITE
    - (retransmission of 180)
    - (retransmission of PRACK)
  - SIP/2.0 200 OK (for PRACK)
- UAS
  - Tells that A-party (UAC) understands reliable responses to 1xx messages
  - Retransmission algorithm starts
  - Retransmission algorithm stops
  - Retransmission algorithm stops

Raimo Kantola –S- 2004 Signaling Protocols 13 - 33

Raimo Kantola –S- 2004 Signaling Protocols 13 - 34
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

• 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

[Diagram showing the signaling process with steps labeled for UAC, INVITE, SIP Proxy(s), UAS, PRACK, 200 OK, 180 Ringing, ACK, etc.]
SIP event notifications tell about remote significant events to the local party

Event-subscriber

SUBSCRIBE

200 (of SUBSCRIBE)

NOTIFY

200

NOTIFY

200

SUBSCRIBE Expires: 0 (unsubscribe)

200

Event-notifier

Generate immediate state response

Generate state change event

Use cases: automatic call-back, automatic buddy lists, message waiting indication and “IN triggering”.

SIP MESSAGE provides Instant Messaging capability in Pager mode

Sender

MESSAGE

sip:user2@domain.com SIP/2.0

Via: SIP/2.0/TCP user1pc.domain.com;branch=z9hG4bK776sgdkse

Max-Forwards: 70

From: sip:user1@domain.com;tag=49583

To: sip:user2@domain.com

Call-ID: asd88asd77a@1.2.3.4

CSeq: 1 MESSAGE

Content-Type: text/plain

Content-Length: 18

Watson, come here.

200 (of MESSAGE)

Destination

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)
- Automatic configuration
  - DHCP or Service Location Protocol (SLP)
- Caller Preferences
  - New headers: Accept-Contact, Reject-Contact, Request-Disposition
- REFER
  - For session transfer (Refer-To: and Reffered-By: )
- ...

Call Setup Examples based on Generic SIP
**Registration example with SIP**

```
REGISTER sip:registrar.biloxi.com SIP/2.0
Via: SIP/2.0/UDP bobspc.biloxi.com:5060;branch=z9hG4bKnashds7
Max-Forwards: 70
To: Bob <sip:bob@biloxi.com>
From: Bob <sip:bob@biloxi.com>;tag=456248
Call-ID: 843817637684230@998sdasdh09
CSeq: 1826 REGISTER
Expires: 7200
Content-Length: 0
```

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP bobspc.biloxi.com:5060;branch=z9hG4bKnashds7
;received=192.0.2.4
To: Bob <sip:bob@biloxi.com>;tag=2493k59kd
From: Bob <sip:bob@biloxi.com>;tag=456248
Call-ID: 843817637684230@998sdasdh09
CSeq: 1826 REGISTER
Expires: 7200
Content-Length: 0
```

**Call Setup example with one proxy**

```
1) INVITE
4) 180 Ringing
7) ACK
2) INVITE
5) 200 OK
8) ACK
3) 180 Ringing
6) 200 OK
```

```
10) BYE
11) 200 OK
12) 200 OK
```

```
Proxy.com
121.110.101.111
```

```
GuyA
UserA@here.com
100.101.102.103
```

```
GuyB
UserB@there.com
110.111.112.113
```
Call Setup example with two proxies

1a) INVITE
1c) 100 Trying
3c) 180 Ringing
4c) 200 OK
5a) ACK

2a) INVITE
2b) 100 Trying
3b) 180 Ringing
4b) 200 OK

3a) 180 Ringing
4a) 200 OK

6a) BYE

Media stream

Alice
 atlanta.com proxy
 Bob
 Biloxi.com proxy

Bob
 atlanta.com proxy
 Biloxi.com proxy

Call Setup example with two proxies

1a) INVITE
1c) 100 Trying
3c) 180 Ringing
4c) 200 OK
5a) ACK

2a) INVITE
2b) 100 Trying
3b) 180 Ringing
4b) 200 OK

3a) 180 Ringing
4a) 200 OK

6a) BYE

Media stream

Alice
 atlanta.com proxy
 Bob
 Biloxi.com proxy

Bob
 atlanta.com proxy
 Biloxi.com proxy
Call Setup example with two proxies

Alice → atlanta.com proxy → Biloxi.com proxy → Bob

1a) INVITE
1c) 100 Trying
3c) 180 Ringing
4c) 200 OK
5a) ACK

1b) INVITE
2b) 100 Trying
3b) 180 Ringing
4b) 200 OK

2a) INVITE
3a) 180 Ringing
4a) 200 OK

5a) ACK

Media stream

atlanta.com proxy

Biloxi.com proxy

6a) BYE

7a) 200 OK
Call Setup example with two proxies

1a) INVITE
1c) 100 Trying
3c) 180 Ringing
4c) 200 OK
5a) ACK

1b) INVITE
2b) 100 Trying
3b) 180 Ringing
4b) 200 OK

2a) INVITE
3a) 180 Ringing
4a) 200 OK

3a) 180 Ringing
4a) 200 OK

6a) BYE

7a) 200 OK

Media stream

atlanta.com proxy
Biloxi.com proxy
Bob
}

2004 Signaling Protocols

Raimo Kantola –S-
Registration example with SIP authentication

1) REGISTER
   Call-ID: 123@here.com

2) 401 Unauthorized
   WWW-Authenticate: <Challenge>

3) REGISTER
   Call-ID: 321@here.com
   Authorization: <Authorization info>

4) 200 OK

Call Setup example with a non-working proxy

1) INVITE
2) INVITE (6x)
3) CANCEL, BYE
4a) INVITE

5b) 180 Ringing
6b) 200 OK
7a) ACK

4b) INVITE
5a) 180 Ringing
6a) 200 OK

6b) 200 OK
7a) ACK
8b) BYE
9a) 200 OK

Media stream

8a) BYE
9b) 200 OK
Call Setup example with a Redirect server

1a) INVITE
2) INVITE
4a) 180 Ringing
4b) 180 Ringing
5b) 200 OK
6a) ACK

1b) INVITE
1c) INVITE
2) 301 Moved Temporarily
3) ACK

5a) 200 OK
6b) BYE
7a) 200 OK
7b) 200 OK

Services use many protocols

• New services and more flexible service creation should differentiate IP Communications Network from PSTN
• Services should combine different forms of communication, thus multiple protocols are needed:
  – SIP for media sessions and session related services, subscriptions and notifications?, messaging?
  – HTTP for web and transactions
  – SMTP for e-mail
  – RTSP for media streaming
• The use of these protocols is orchestrated by the service logic: context is set up using SIP.
Routing and Service Model in 3G

P1, P4: Outbound Proxies
P2, P3: Registrar Proxies
AS1, AS2: Application Servers

NB: Also AS based on direct processing of call state:
There is no Basic call state model like in IN

SIP Entities & Service Capabilities

- User Agent (= UAC + UAS)
  - Can run services, such as forwarding, filtering etc.
  - Not always connected (out of coverage/battery etc.)
- Redirect Server
  - Can do services that require only Request-URI change, e.g. translation, parameter addition etc.
- Proxy Server
  - Can change certain headers and stay in the signaling path
  - Forking, actions based on responses
- Back-to-Back User Agent (= both ways User Agent)
  - Can e.g. issue requests to a call leg or modify SDP
  - In many cases necessary
Application Server in 3G?

- Fuzzy Definition but has SIP+ interface!
- Can be a Redirect or Proxy Server or Back-to-Back UA
- The key is that it should be *programmable*
  - Routing based on service logic: what to do when user not registered or busy
  - URI translation: Reachability chains
  - Interfaces to other protocols: HTTP, SMTP, RTSP etc.
- Can be single purpose boxes, or multi-purpose boxes, or controllers who orchestrate things

3GPP Network Model (preliminary: …)
Different Kinds of CSCFs

Proxy CSCF: Provides emergency service breakout, triggers for locally-provided services, and number normalizing (per local dialing plan)

Interrogating CSCF: Queries the HSS to find the correct S-CSCF. First point of contact for incoming call signalling.
Different Kinds of CSCFs

Serving CSCF: Provides subscriber services.

SIP Proxy vs B2BUA
Overview of routing between two mobile terminals

Media stream

Appendix B – 3GPP IMS call flows
IP Multimedia Registration 1.

1. SIP REGISTER
2. SIP REGISTER
3. Cx-Query
4. Serving Network Selection
5. Cx-Query Resp
6. Cx-Select-pull
7. Cx-Select-pull Resp

IMS Registration 1a. - S-CSCF in home network

1. S-CSCF selection
2. SIP REGISTER
3. Cx-put
4. Cx-put Resp
5. Cx-Pull
6. Cx-Pull Resp
7. SIP 200 OK
8. SIP 200 OK
9. SIP 200 OK

Diameter
IMS Registration 1b. - S-CSCF in visited network

1. SIP REGISTER
2. S-CSCF Selection
3. Register
4. Cx-Put
5. Cx-Put Resp
6. Cx-Pull
7. Cx-Pull Resp
8. SIP 200 OK
9. SIP 200 OK
10. SIP 200 OK
11. SIP 200 OK

Mobile to Mobile Call

1. INVITE
2. INVITE
3. SERVICE CTRL
4. INVITE
5. CxLocQuery
6. CxLocResp
7. INVITE
8. SERVICE CTRL
9. INVITE
10. BEARER ESTABLISHMENT
11. ALERTING + SESSION OFFERING
12. INVITE
13. 200 OK
14. 200 OK
15. SERVICE CTRL
16. 200 OK
17. 200 OK
18. SERVICE CTRL
19. 200 OK
20. 200 OK
21. ACK (possibly hop-by-hop)
Call flow examples 1. - no answer

1. UE → P-CSCF → S-CSCF → HSS → S-CSCF → P-CSCF → UE
   - INVITE
   - 180 Ringing
   - CANCEL
   - 200 OK

2. INVITE
   - CxLocQuery
   - CxLocResp
   - INVITE
   - 180 Ringing
   - CANCEL
   - 200 OK

Call flow examples 1. - no answer 2.

1. UE → P-CSCF → S-CSCF → I-CSCF → HSS → S-CSCF → P-CSCF → UE
   - 487
   - ACK

2. ACK
Call flow examples 2. - busy

Calling Party

UE → INVITE → P-CSCF → INVITE → S-CSCF

Called Party

INVITE → S-CSCF → INVITE → HSS → INVITE → P-CSCF → INVITE → UE

486 Busy Here → ACK → ACK

Call flow examples 3. - no response

Calling Party

UE → INVITE → P-CSCF → INVITE → S-CSCF

Called Party

INVITE → S-CSCF → INVITE → HSS → INVITE → P-CSCF → INVITE

408 Req. Timeout → CANCEL+BYE → ACK

Cancelation + BYE
Call flow examples 4. - temporarily unavailable

How to Program Services

• Call Processing Language
• SIP CGI
• SIP Servlets
• SIP JAIN
• Soft SSF and INAP/CAP
• Parlay
• OSA

=> Whatever… Different abstraction levels
The claim is that it should be as open as flexible as creating services in the web these days
Server types for different services

- Media Server (SIP, RTSP, HTTP)
  - Announcements, IVR, Voicemail, Media on demand
- Conferencing Server (SIP)
  - Media mixer
- Presence Server (SIP)
  - Users status info, capabilities, willingness to communicate
- Web Server (HTTP), E-mail Server (SMTP), Messaging Server (SIP?), Text-to-Speech Server etc.
- Controller Server
  - Co-ordinates the overall service

=> Server resources can be addressed by URLs, no need for tight coupling a la MGCP/Megaco

Third Party Call Control is based on SIP

![Diagram](https://example.com/diagram.png)

- Details are still to be solved in the IETF
- Powerful tool e.g. for inviting users to centralized conferences or sessions with a Media Server
REFER and Call Transfer

<table>
<thead>
<tr>
<th>Transferor</th>
<th>Transferee</th>
<th>Transfer Target</th>
</tr>
</thead>
<tbody>
<tr>
<td>INVITE/200 OK/ACK</td>
<td>REFER</td>
<td>INVITE/200 OK/ACK</td>
</tr>
<tr>
<td>INVITE (hold)/200 OK/ACK</td>
<td>202 Accepted</td>
<td>200 OK</td>
</tr>
<tr>
<td>NOTIFY (200 OK)</td>
<td>200 OK</td>
<td>INVITE/200 OK/ACK</td>
</tr>
<tr>
<td>BYE/200 OK</td>
<td>BYE/200 OK</td>
<td></td>
</tr>
</tbody>
</table>

Auto-conferencing Service Example

1. One user orders the conference by filling a web form
2. Controller subscribes to each participant's presence
3. When all available, send message or start IVR session to each participant to confirm willingness
4. Connect each participant to conference server. Play announcements to conference from media server when new parties join
Problems

- How to make "service routing"?
- How to make service components really independent?
- If there is dependency, how to move parameters between the components?
- How to secure call release?