



# Intro to WebRTC and VozDigital

Iván Mosquera, Senior Software Engineer

[imosquera@tuenti.com](mailto:imosquera@tuenti.com) | @ivmos | @theevnt | @Tuentieng

# whoami

- From Bilbao
- Enjoying Tuenti since 2010
- Member of Voice Team
- @ivmos
- @TuentiEng

WebRTC

VOZDIGITAL

# WebRTC

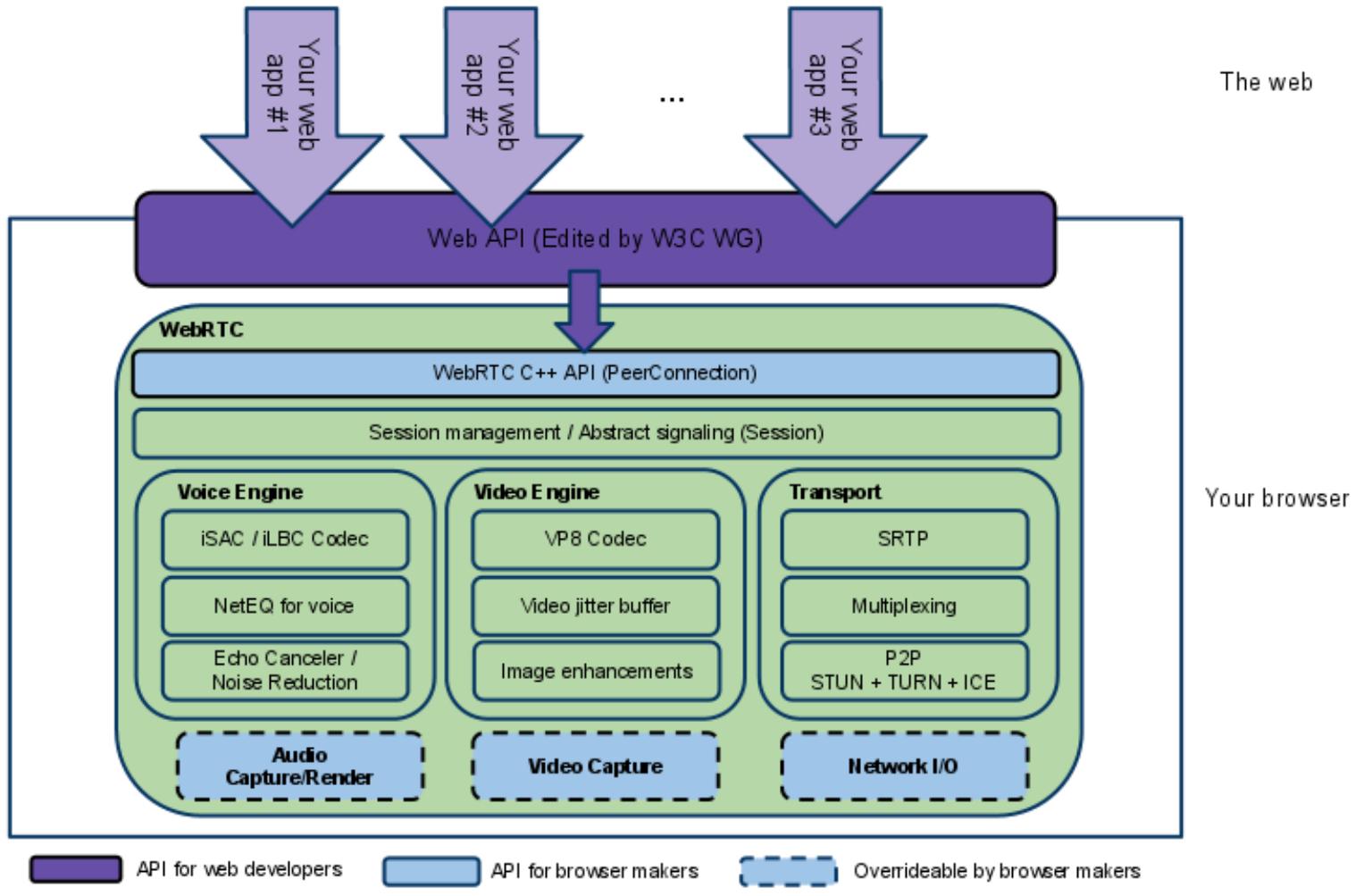
- May 2011
- “To enable rich, high quality, RTC applications to be developed in the browser via simple JavaScript APIs and HTML5.”
- Plugin-free
- Javascript API
- Native API available too

# Before WebRTC?

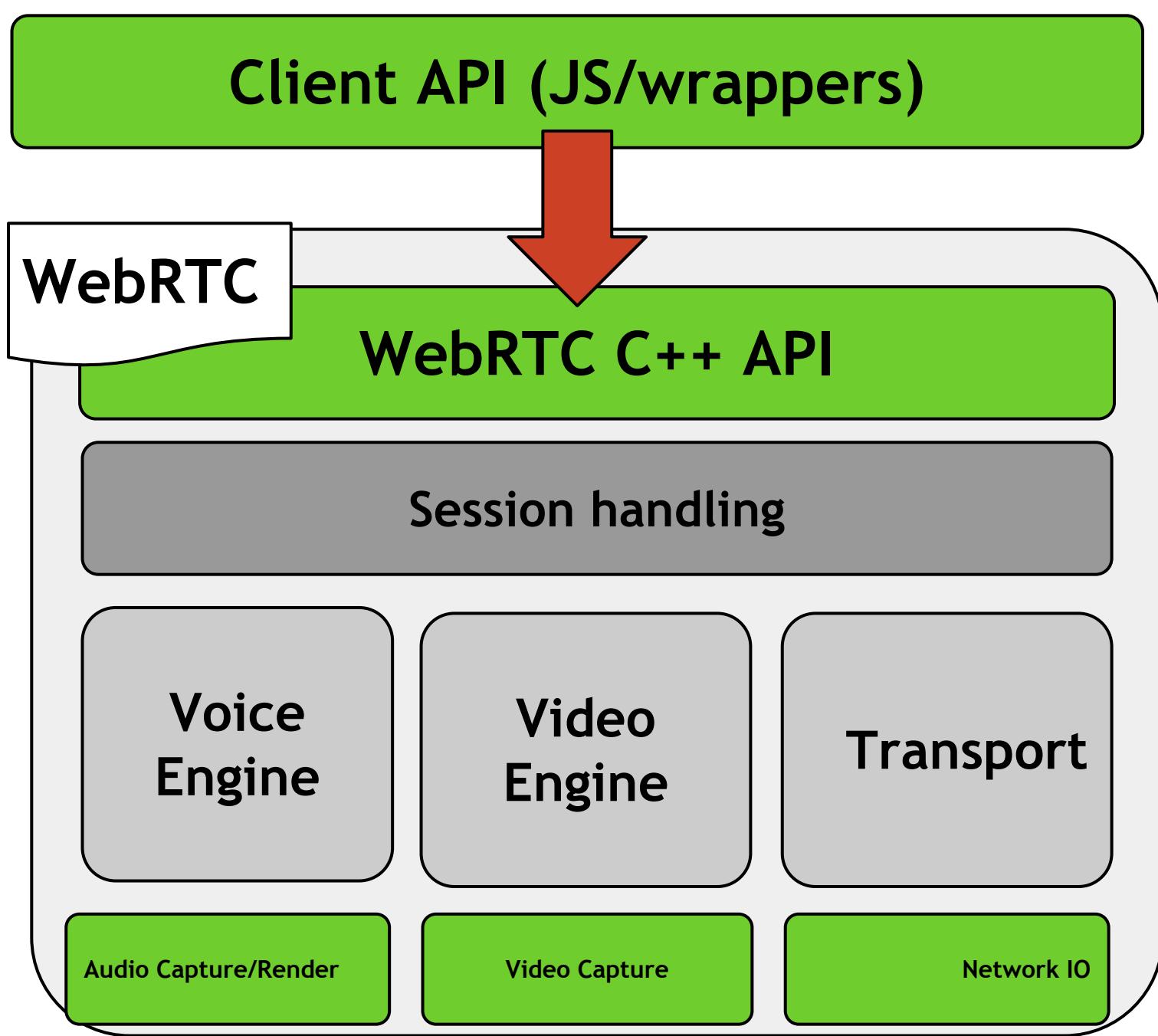
- Plugins: flash, java, native addons..
- Codecs..

# Let's go!

- We need some theory.
- APIs:
  - MediaStream/getUserMedia
  - RTC PeerConnection
  - RTC Data channel (remember, not only a/v!)



# Client API (JS/wrappers)



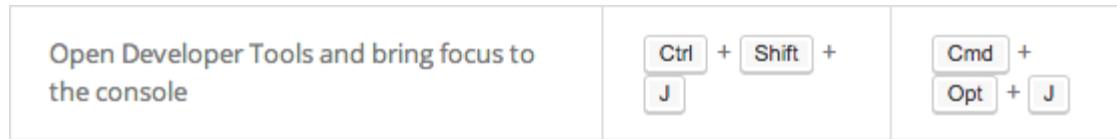
# What WebRTC does not provide...

## Signaling!!!

- WebRTC solves the media transport problem, not the high level signaling one.
- NAT traversal problem: ICE

# Workshop hints

- <https://gist.github.com/ivmos/HASH>
- <http://jsfiddle.net/gh/gist/lib/V/HASH>



```
normalize.css?j...ferinsidebar:1
button, input, textarea, select {
  text-rendering: optimizeLegibility;
  font-size: 100%;
```

Console <top frame>  
result( fiddle.jshell.net/ )



# GO GO GO: let's get media!

<http://jsfiddle.net/gh/gist/library/pure/7fb79f59ab1f2a36ff27/>

Action: check console

- Audio&Video source with [MediaStream](#) API.
- Chrome snippets ([interop](#)) ([polyfill](#))
- [navigator.webkitGetUserMedia](#)

# let's hype a bit

- You can mix with other Javascript APIs
  - [ASCII camera](#)
  - [Photobooth app](#)
- We're not studying it but [RtcDataChannel](#) API is awesome too
  - [`http://www.peer-server.com/`](#)
  - [`https://rtccopy.com/`](#)
  - [`https://github.com/feross/webtorrent`](#)

# We have the video, let's offer it

```
http://jsfiddle.net/gh/gist/library/pure/8ce6340f8870f6d91f0a/2b1596f3124037dedf7ea443413c8a0352599606/
```

Action: check offer SDP

- pc: [RTCPeerConnection](#)
- Once we have the video, with the pc we need to **offer** it with a **description**
- pc.[addStream](#)
- pc.[createOffer](#)

# RTCSessionDescription

- **pc.setLocalDescription(offer)**
- RTCSessionDescription
- offer, answer..
- **sdp** (1998, RTP, RTSP...)
- check offer.sdp :)
- sending that sdp... (no signaling defined)

```
v=0
o=- 4420499408415283528 2 IN IP4 127.0.0.1
s=- 
t=0 0
a=group:BUNDLE audio video
a=msid-semantic: WMS
VNGaWbmub1HsBjaxybQCE91gq0W5RFN3zt2Z
m=audio 1 RTP/SAVPF 111 103 104 0 8 106 105 13 126
c=IN IP4 0.0.0.0
a=rtcp:1 IN IP4 0.0.0.0
a=ice-ufrag:rTPGidiJq18Xsr21
a=ice-pwd:xrvmCOYkTGlIcqNMuCFr9yrf
a=ice-options:google-ice
a=fingerprint:sha-256 A0:15:3C:15:22:98:E9:D1:4C:60:
D0:0D:F7:34:A8:DA:9F:52:E0:D1:D0:16:43:B3:5D:95:EC:
EB:B4:9C:90:D5
a=setup:actpass
a=mid:audio
a=extmap:1 urn:ietf:params:rtp-hdrext:ssrc-audio-level
a=extmap:3 http://www.webrtc.org/experiments/rtp-hdrext/abs-send-time
```

<http://jsfiddle.net/gh/gist/library/pure/6272bd4f5c626ab7a2bc/68170b303b509d9a17d9c968a5447aaee117cbb3/>

Action: check how we're “sending” the offer SDP

<http://webrtchacks.com/sdp-anatomy/>

# Let's answer it

<http://jsfiddle.net/gh/gist/library/pure/7fbe699afc6462f22cd1/a418431770c29a00a252d48730e4613500c196df/>

Actions:

- Send offer sdp to callee
  - Send answer sdp to caller
- 
- isCaller flag
  - pc.**setRemoteDescription**
  - pc.**createAnswer**
  - pc.**setLocalDescription(answer)**

# ... and receiving the answer

<http://jsfiddle.net/gh/gist/library/pure/8d0c1e1bd09fe27761ff/>

Action:

- Offer
- Answer

- Caller does not know about callee yet
- Receiving the answer the same way the callee received the offer (signaling)
- pc.setRemoteDescription with type.answer
- check pc state..

# pc state until now, finished?

<http://jsfiddle.net/gh/gist/library/pure/8d0c1e1bd09fe27761ff/>

Same code, but this time pay attention to the pc state

- pc.localDescription OK
- pc.remoteDescription OK
- pc.onicecandidate ???
- pc.onaddstream ???

# ICE

- Purpose
- Candidate
- Types:
  - local
  - reflexive/stun
  - relay/turn
- pc.onicecandidate
- pc.iceconnectionstate

<http://jsfiddle.net/gh/gist/library/pure/0ea6abc8a6f7064895a6/f1086199fb3296a39172ac2119c05e9fd851a29/>

Action: check refactor

<http://jsfiddle.net/gh/gist/library/pure/0ea6abc8a6f7064895a6/1d4a6676e86824353b4b17aab02031690b28f5/>

Action:

- check the candidates
- check pc.iceGatheringState

<http://jsfiddle.net/gh/gist/library/pure/0ea6abc8a6f7064895a6/c9d54da712a7d69d44dafa4e1339fea3764ca01a/>

Action:

- Check candidates again

<http://jsfiddle.net/gh/gist/library/pure/0ea6abc8a6f7064895a6/798c3aba9bf521f508ad31f9d0e93e6d81ccb61f/>

Actions: do the whole flow again, offer, answer and candidates

# no video yet, what's missing?

- pc.onaddstream

[http://jsfiddle.net/gh/gist/jquery/1.11.0  
/0ea6abc8a6f7064895a6/05f7404850c1933e07fc4b72faa07841180c320f/](http://jsfiddle.net/gh/gist/jquery/1.11.0/0ea6abc8a6f7064895a6/05f7404850c1933e07fc4b72faa07841180c320f/)

Action: try again the whole flow, you should now have video in both tabs

:)

# Recap

- create pc
- create offer, pc.setLocalDescription(offer)
- send offer
- receive offer
- pc.setRemoteDescription(offer)
- pc.createAnswer, pc.setLocalDescr(answer)
- send answer
- receive answer
- pc.setRemoteDescription(answer)
- ICE!!

# Recap: caller

CREATE PEER  
CONNECTION

CREATE OFFER

SEND OFFER

RECEIVE ANSWER

SAVE ANSWER

PROFIT!

# Recap: callee

RECEIVE OFFER

CREATE PEER  
CONNECTION

SAVE OFFER

CREATE ANSWER

SEND ANSWER

PROFIT!

BOTH

SEND ICE  
CANDIDATE



# Introducción a WebRTC y su uso en nuestro servicio de VozDigital

Volvemos en unos minutos... estamos de break ;)

Iván Mosquera, Senior Software Engineer

[imosquera@tuenti.com](mailto:imosquera@tuenti.com) | @ivmos | @theevnt | @Tuentieng

# Server signaling iteration

- XHR polling example
- API:
  - startSession(sid, success)
  - getParticipants(sid, success)
  - getMessages(sid, success)
  - sendOffer(sid, to, offerSdp, success)
  - sendAnswer(sid, to, answerSdp, success)
  - sendIceCandidate(sid, to, candSdp, success)
  - sendTerminate(sid, to, success)
- MySql backend:
  - sessions
  - requests

Base snippet:

<http://jsfiddle.net/gh/gist/jquery/1.11.0/9a93a110efb9cf0ae46f/7d03c3c617753d2029950b05181d557925a4a7d4/>

# 1. Use real signaling

- Instead of copy-paste signaling, use the provided API. “s/console.log/Api.\*/g”
- mySid, otherSid hardcoded by now, we still assume we’re only working with 2 tabs.
- Candidates to be sent one by one
- Api.startSession needed

## 2. Listen for messages

- We need to listen for offer, answer and candidate type messages
- setInterval loop

### 3. adding candidates fix

- Check pc.remoteDescription before pc.addIceCandidate. Enqueue if needed.
- Do the pc.addIceCandidate after pc.setRemoteDescription
- Not a problem for caller as he starts the signaling
- Profit!

# Improvements

- unharcode otherSid
  - <select> for to (callee)
    - getParticipants loop
- Call button: isCaller and offer
  - call method
- unharcode mySid
  - <input> for sessionId (caller)
- Add your own video too

<http://jsfiddle.net/gh/gist/jquery/1.11.0/5692b5b7c0fa4d855d2b/>

# Security

- In our implementation, signaling security concerns?
- In media channel?
  - WebRTC security

# A few links

- [WebRTC 1.0 RFC](#)
- Understanding why [ORTC](#) effort ([Iñaki Baz interview](#))
- JS Lib using SIP signaling: <http://jssip.net>
- <http://webrtcchacks.com/>
- <https://www.webrtc-experiment.com/> ([Muaz Khan interview](#))
- <http://corporate.tuenti.com/en/dev/blog/Building-a-VoIP-Service-using-WebRTC>

WebRTC

VOZDIGITAL

# VOZDIGITAL

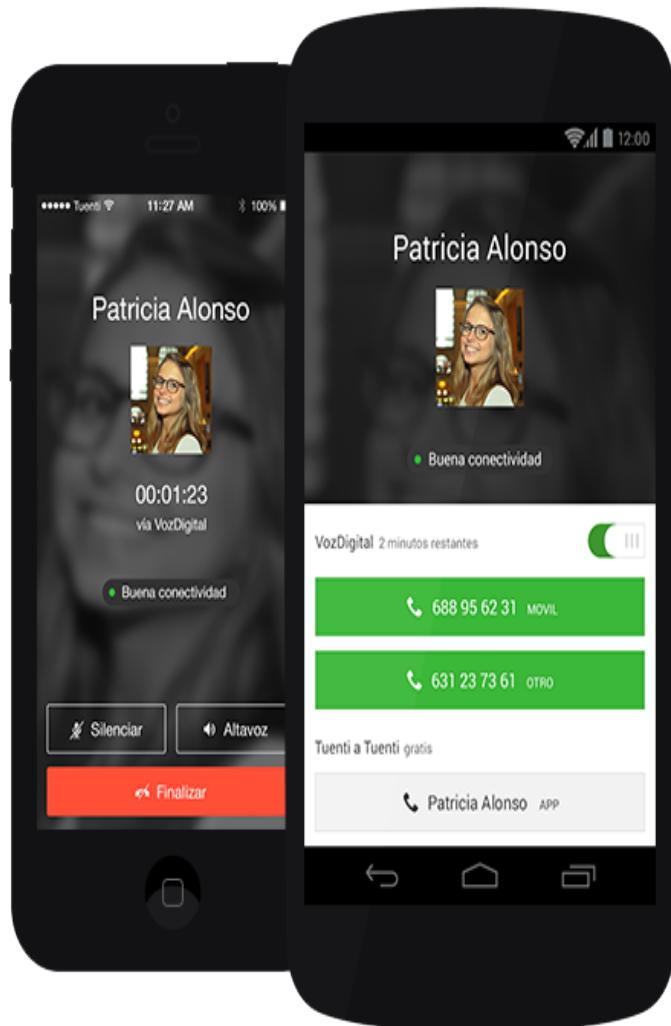
An inbound & outbound VoIP service

...using the customer's GSM number!

...with no additional data charges!

...integrated with existing Tuenti chat infrastructure

# WebRTC for App2App calls



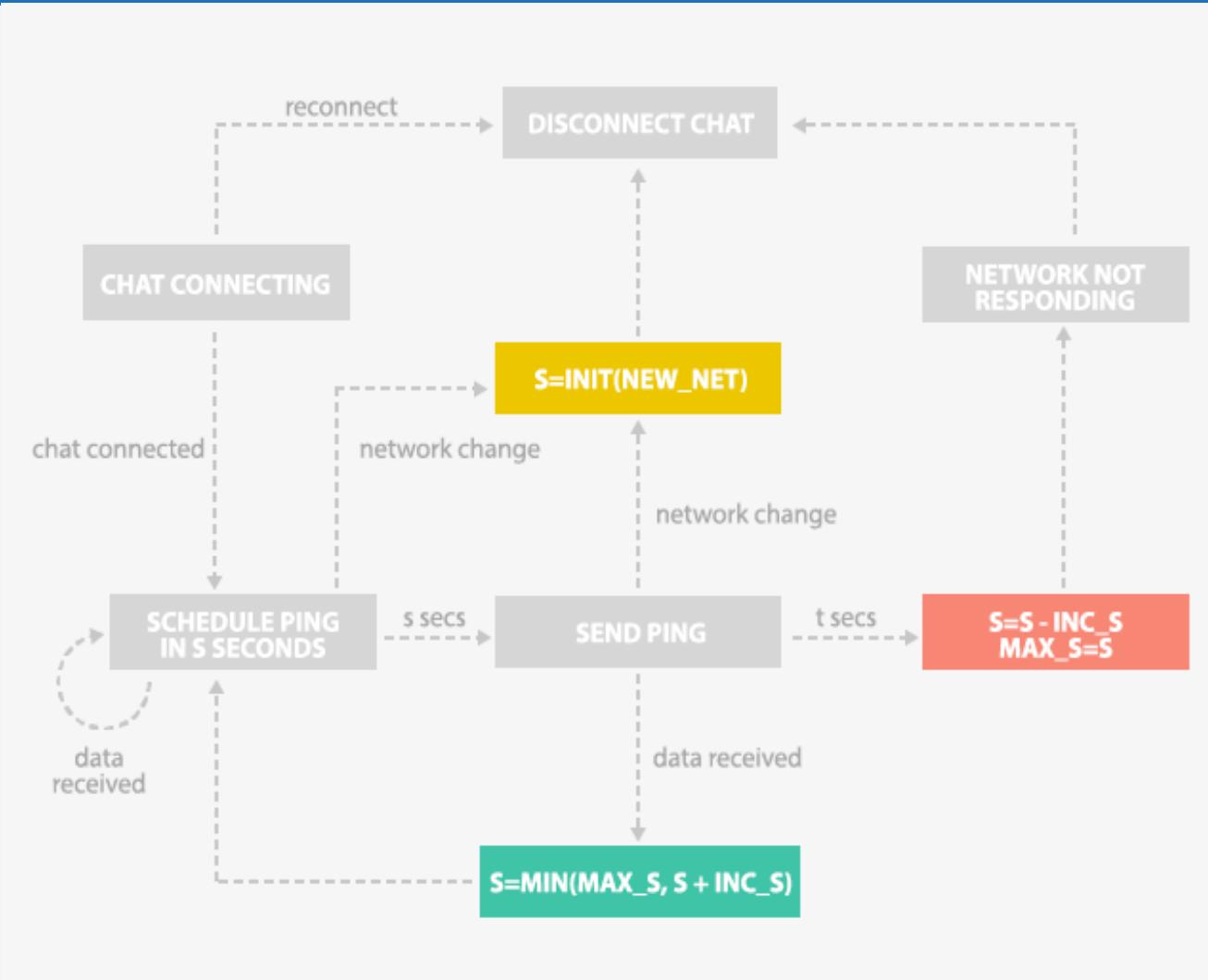
# Multiple resources



Will he be  
available??



# Always Connected Application





# ZEROLÍMITES

LLAMA, CHATEA Y COMPARTE CON TUENTI MÓVIL Y LA NUEVA APP

INCLUSO SIN SALDO, SIN CONSUMIR DATOS  
TOTALMENTE GRATIS Y SIN LÍMITES

## ZEROSALDO

Ahora, hasta sin saldo,  
también estás conectado

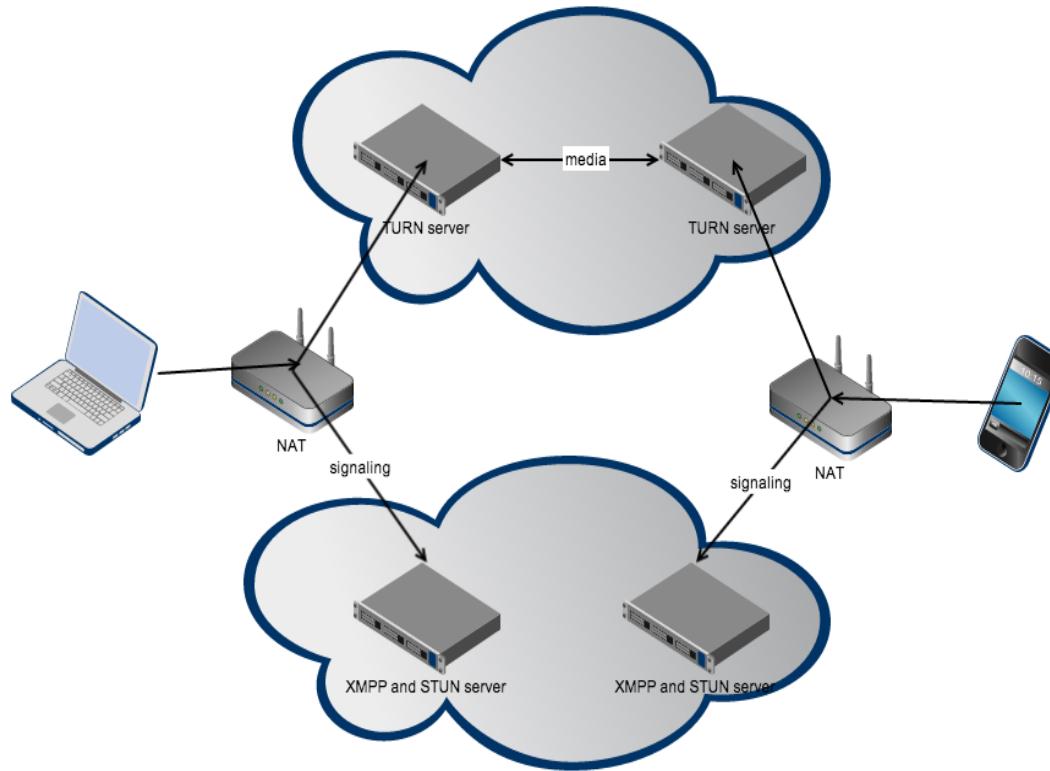
## ZEROCONSUMO

Mantente conectado  
y olvídate de contar megas

## ZEROBARRERAS

Llama gratis desde nuestra app a tus  
amigos sean del operador que sean

# Solution Overview



# Signaling over XMPP

- Jingle protocol

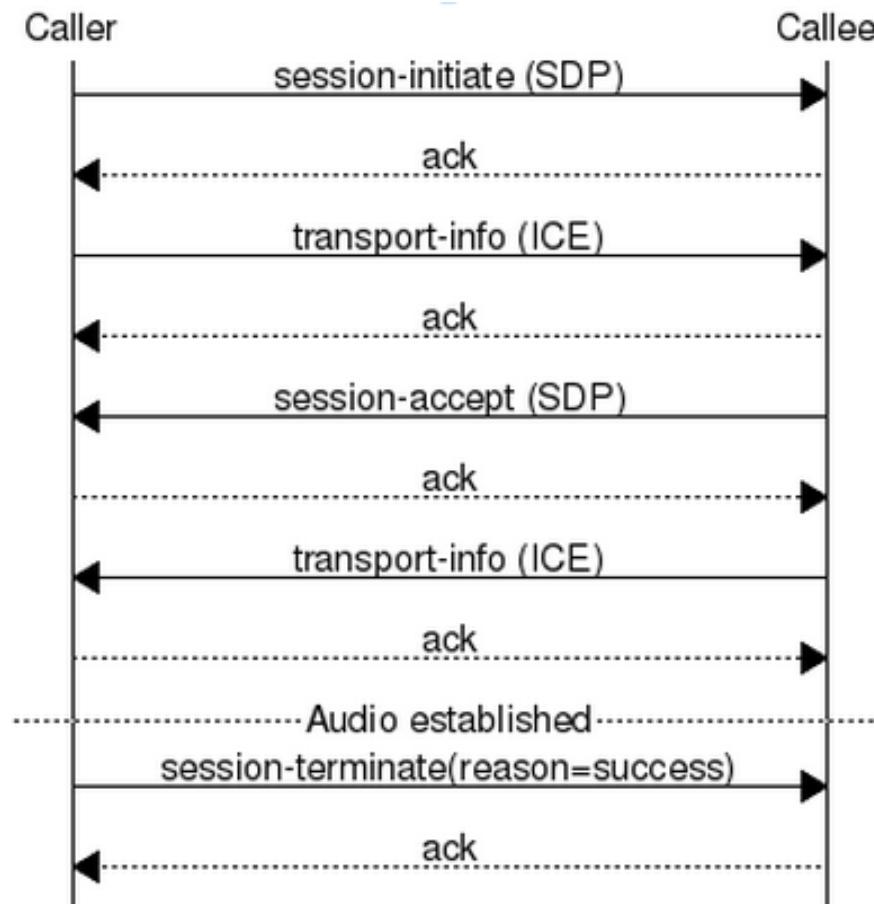
```
<jingle xmlns='urn:xmpp:jingle:1'  
        action='session-initiate'  
        initiator='romeo@montague.lit/orchard'  
        sid='a73sjjkla37jfea'>  
  <content creator='initiator' name='voice'>  
    <description xmlns='urn:xmpp:jingle:apps:rtp:1' media='audio'>  
      <payload-type id='96' name='speex' clockrate='16000'/>  
      <payload-type id='97' name='speex' clockrate='8000'/>  
      <payload-type id='18' name='G729'/>  
      <payload-type id='0' name='PCMU' />  
      <payload-type id='103' name='L16' clockrate='16000' channels='2' />  
      <payload-type id='98' name='x-ISAC' clockrate='8000' />  
    </description>  
  </content>  
</jingle>
```

# SDP over XMPP in tuenti

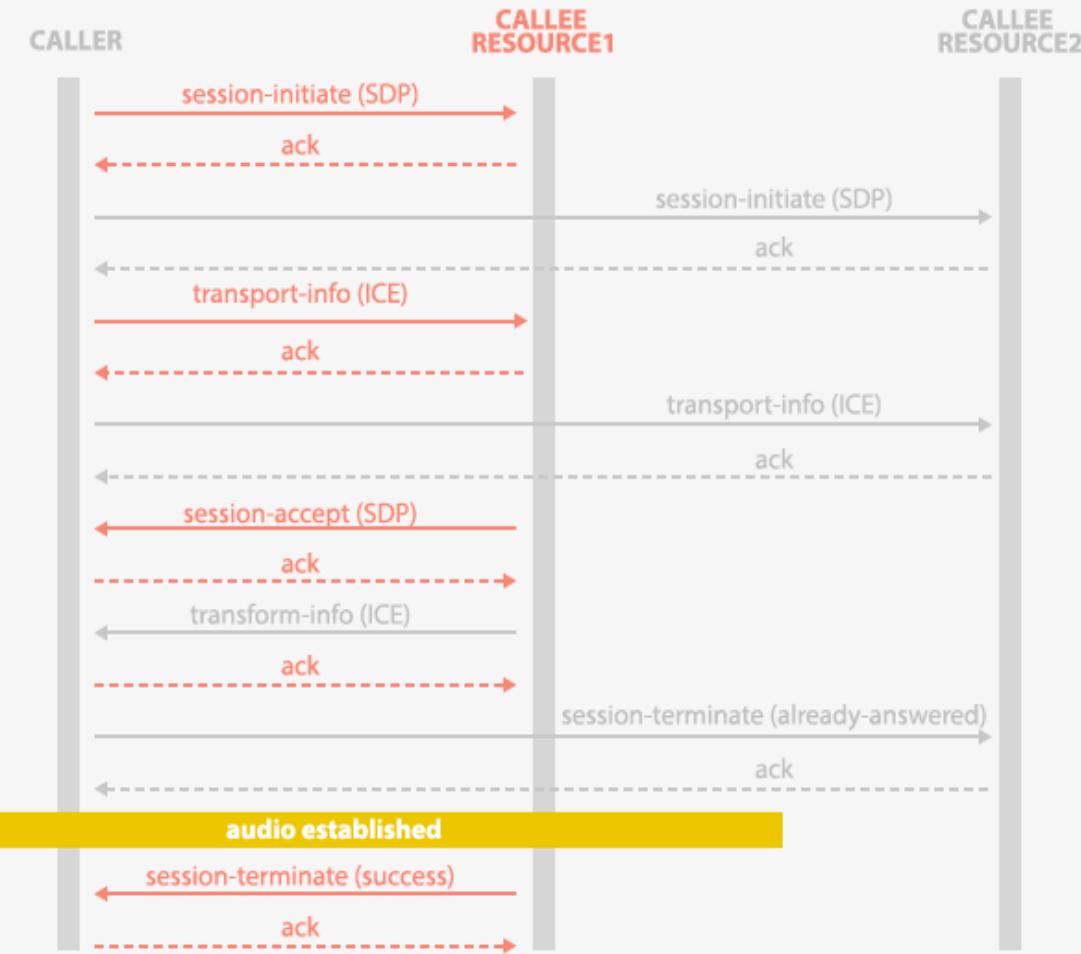
- Tangle protocol

```
<tangle xmlns='urn:tuenti:tangle:1'  
       action='session-initiate'  
       initiator='javierf@tuenti.com'  
       sid='a73sjjkla37jfea'>  
    <sdp><![CDATA[SDP]]></sdp>  
</tangle>
```

# VoIP message sequence chart



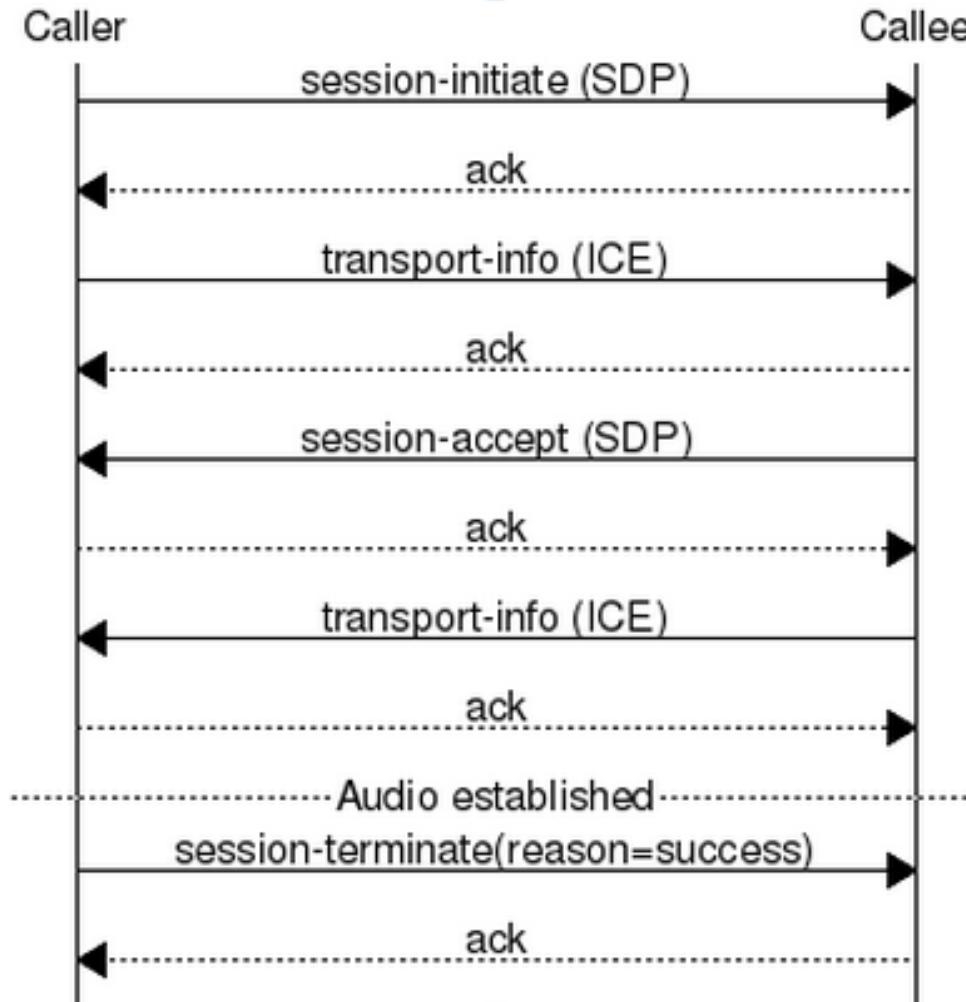
# Tangle multiple resources



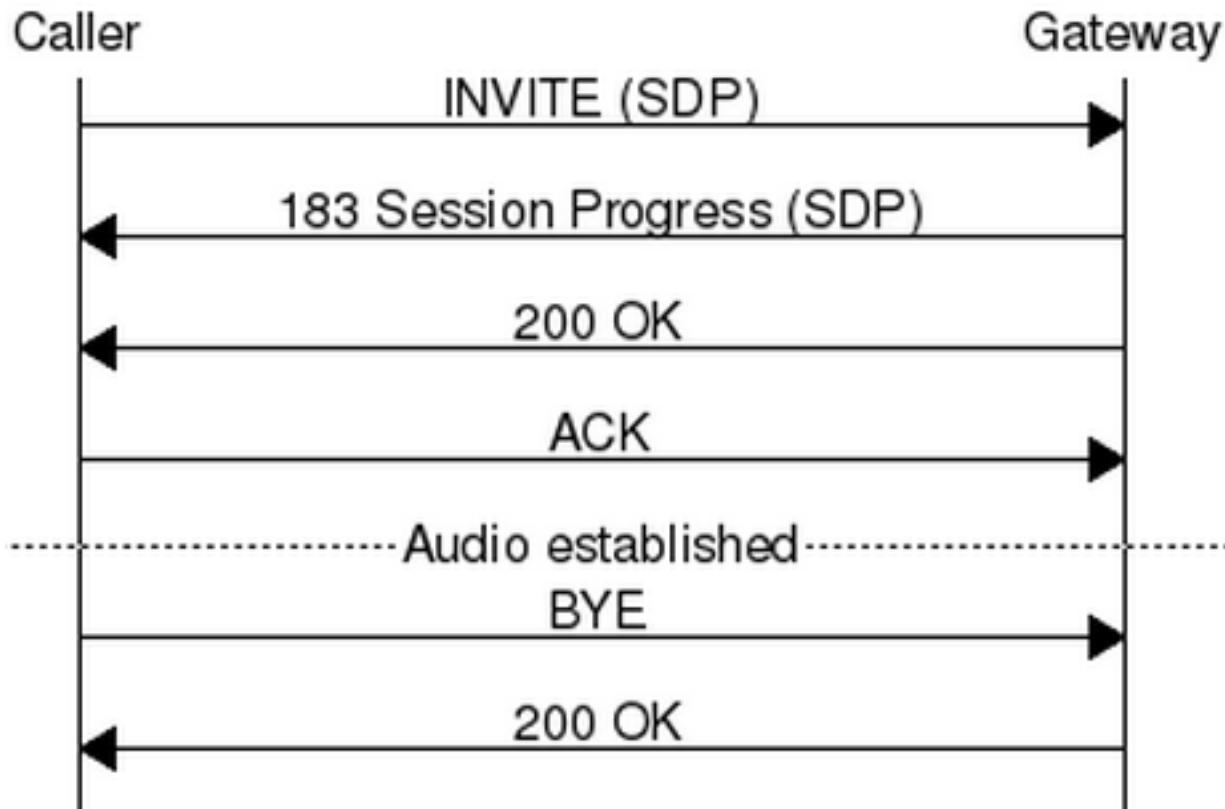
# We need a SIP stack!



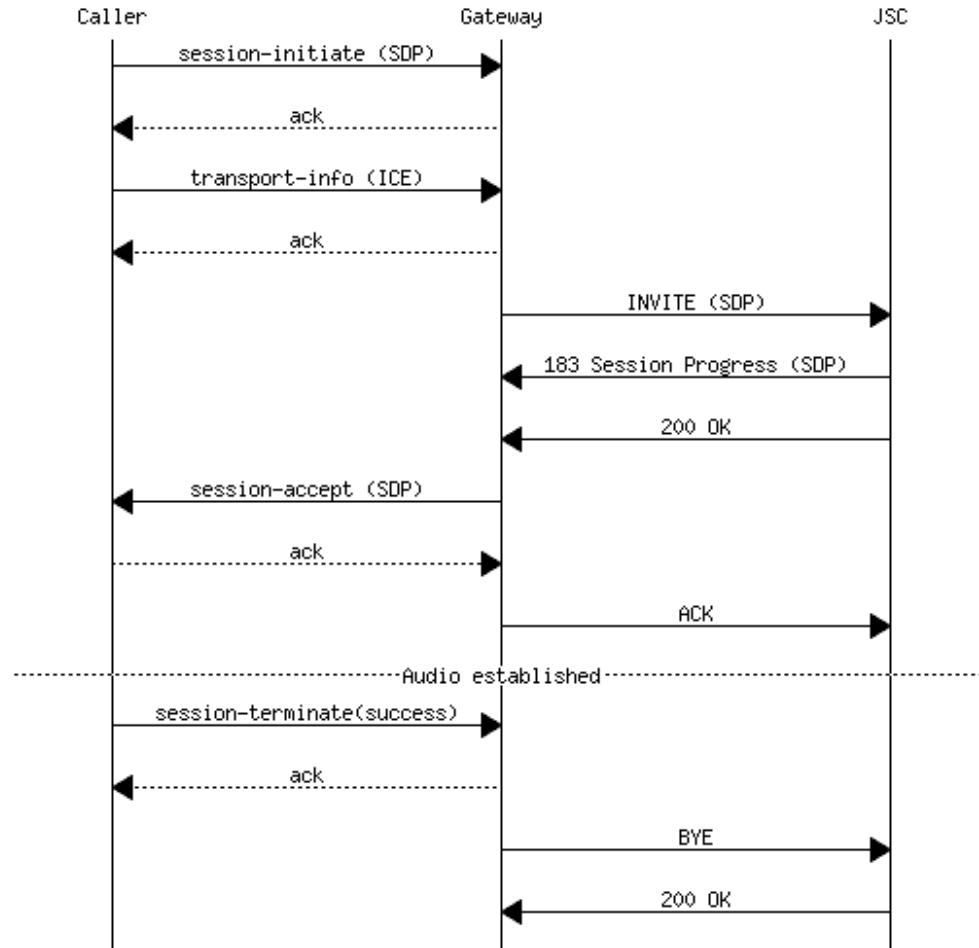
# VoIP message sequence chart



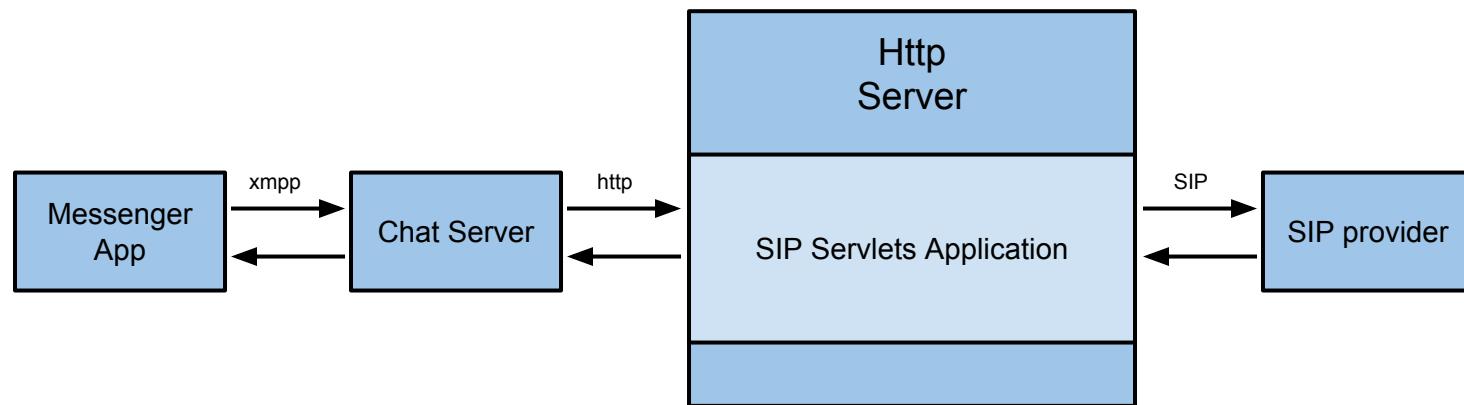
# SIP message sequence chart

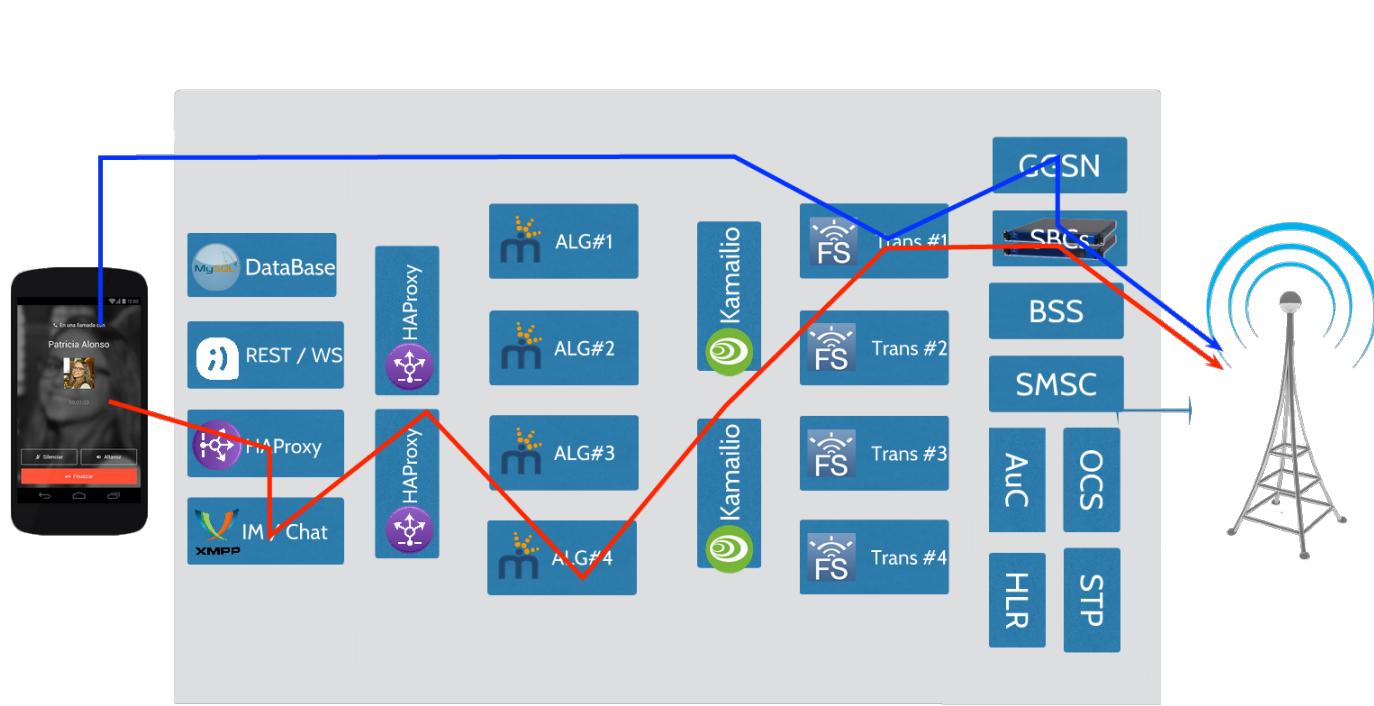


# Merging the message sequence charts



# Voice Gateway Overview





# Questions?

