



Golang to build a real-time interactive SaaS Cloud

iTutorGroup

2019.4.23 董海冰



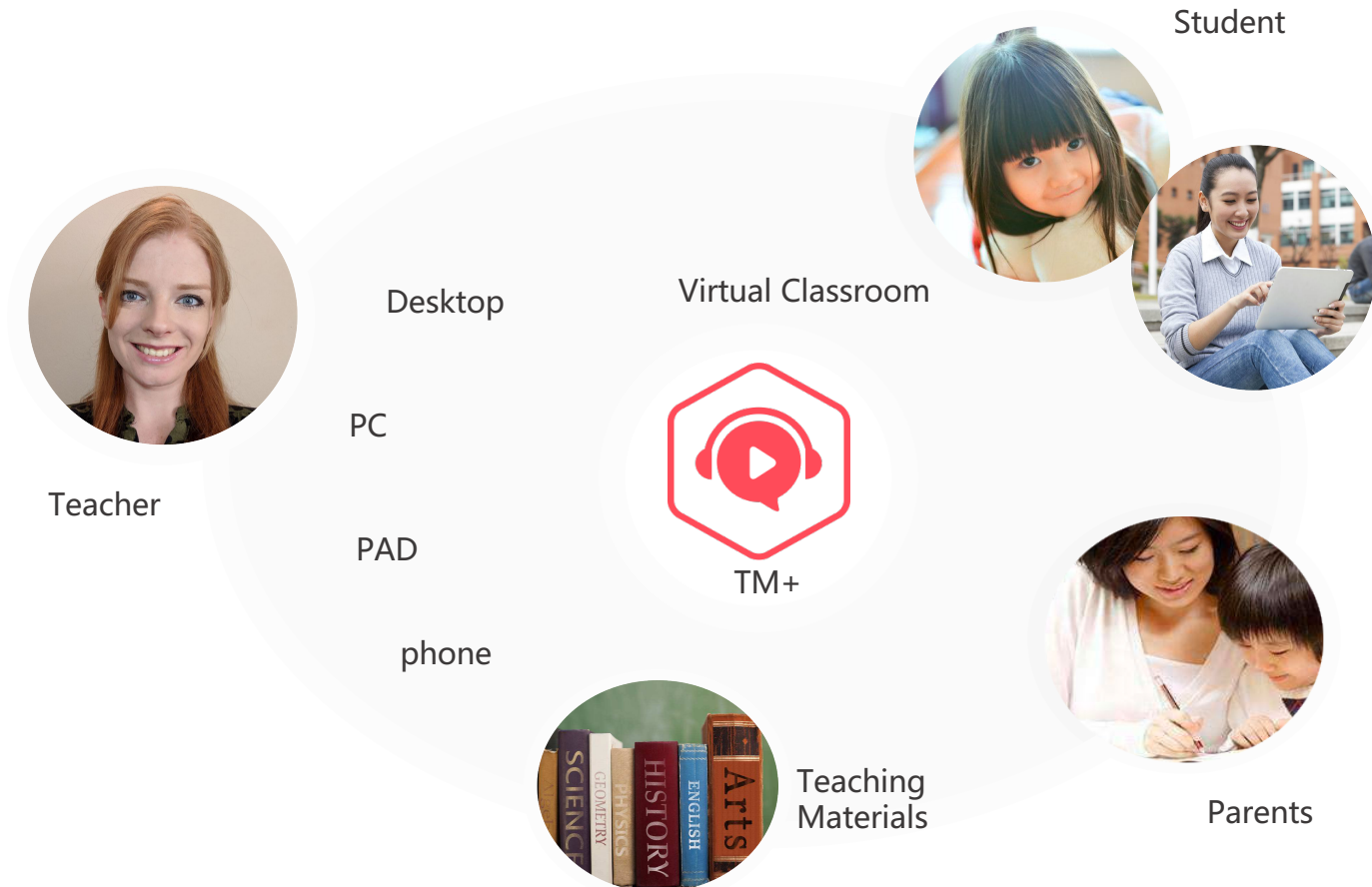
探探 Gopher China 2019

Agenda

- What's TutorMeet+
- Why use Golang
- WebRTC Example
- Problems encountered
- About SaaS
- Q&A



What's TutorMeet+ ?



TutorMeet+

The screenshot displays the TutorMeet+ interface during a lesson. At the top, the 'vipJr' logo is on the left, and a timer shows '26:35:02'. A notification bar reads: 'DO check the clients' understanding by asking for questions at the end of every 1 or 2 slides'. On the right, there are icons for refresh, info, settings, and a '结束课程' (End Lesson) button.

The main content area is titled 'Vocabulary' and shows a slide with the word 'Planet' in large orange letters. The slide is decorated with colorful illustrations of a planet with rings, a sun, a rocket, and a planet with a face. The 'vipJr' logo is in the top left of the slide.

On the left side, there is a vertical list of participants: Susan M, Emily, and Tom. Each participant has a small video thumbnail and a star icon. Below the list, there are icons for '在线学员 (3)' (3 Online Students) and '讨论区' (Discussion Area).

At the bottom right of the slide, there is a 'Session Report' button and a 'Lesson Plan Feedback' section with a 'Length' field and a star rating.



Feature

- Language : **Golang**、C++、ReactJs ;
- Video : **VP8**、VP9、**H264**、AV1 ;
- Audio : **OPUS**、AAC ;
- Deployment : **Docker**、Kubernetes、Hybird cloud ;
- Support : 1v1、**1vN**、NvN (连麦)



Why use Golang ?

- History :

Python or C (C++) ?



Golang advantage

- **Efficiency (多)**
- **Performance (快)**
- **Engineering (好)**
- **Less is more (省)**

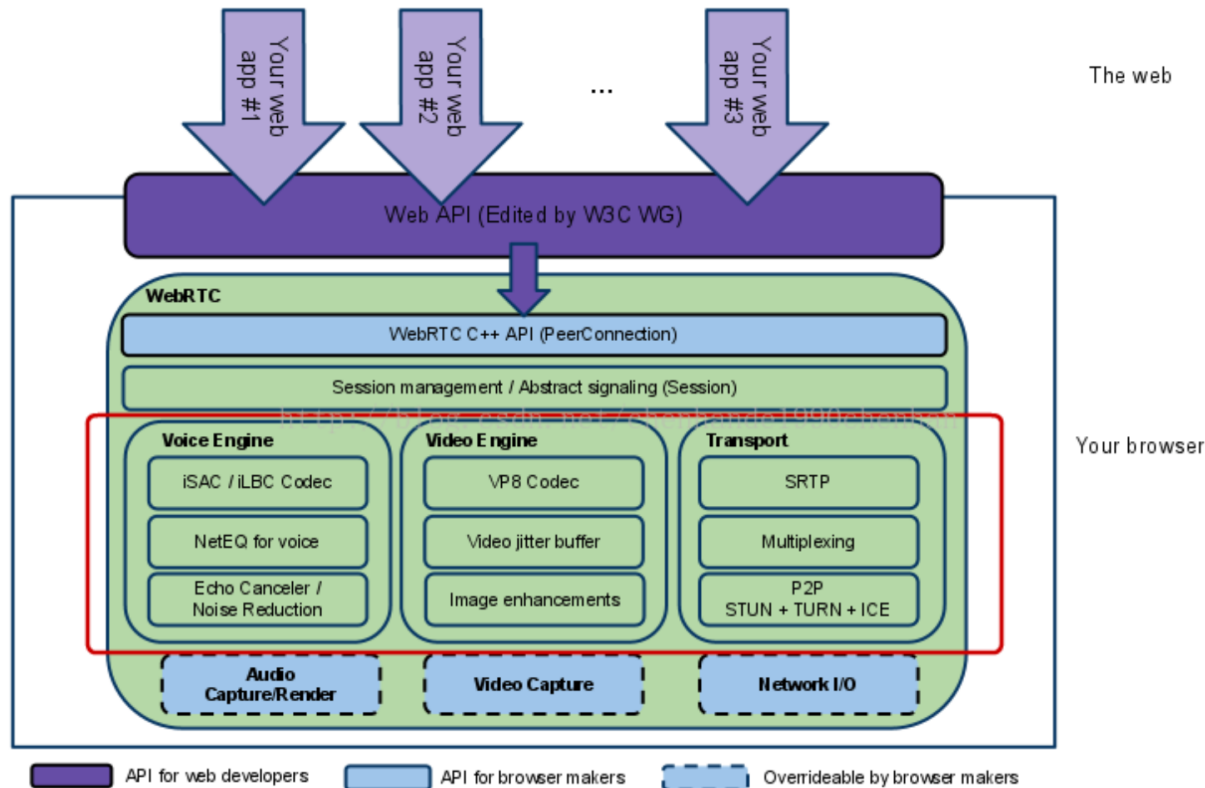




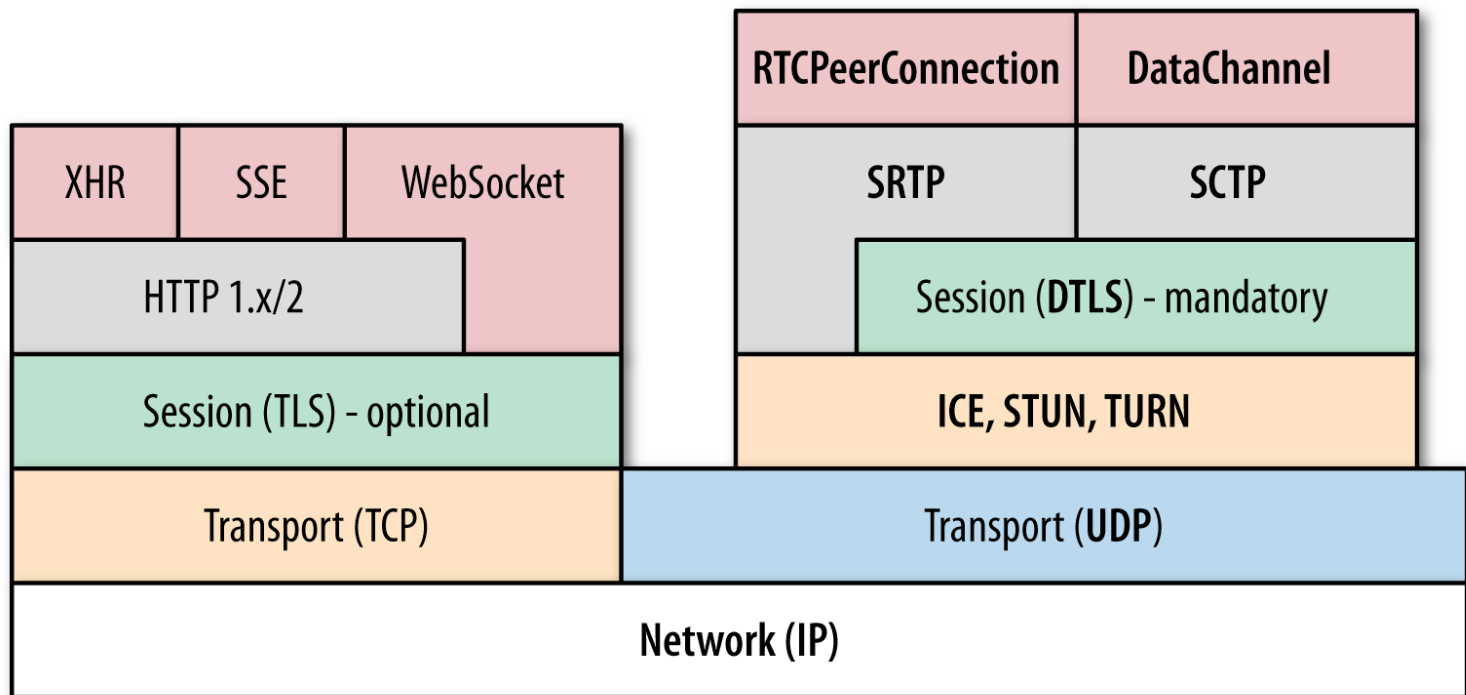
WebRTC Overview



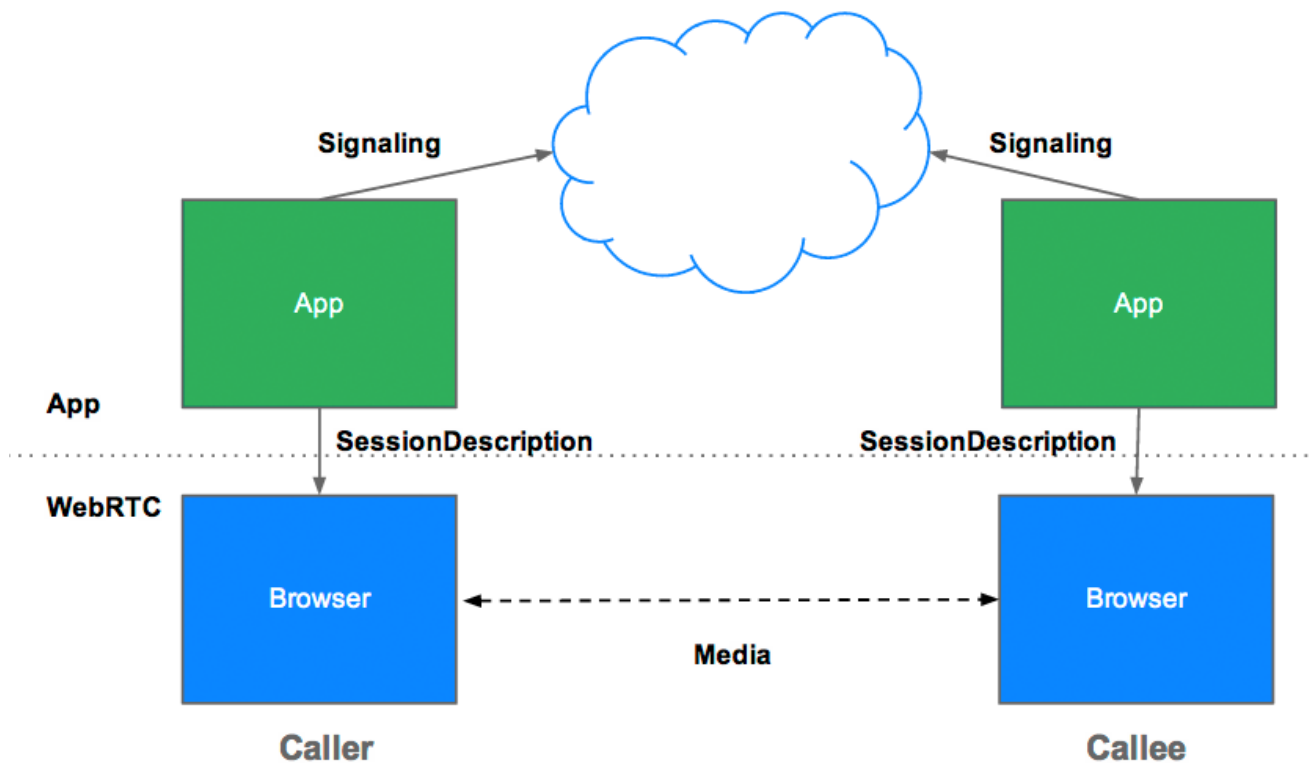
WebRTC



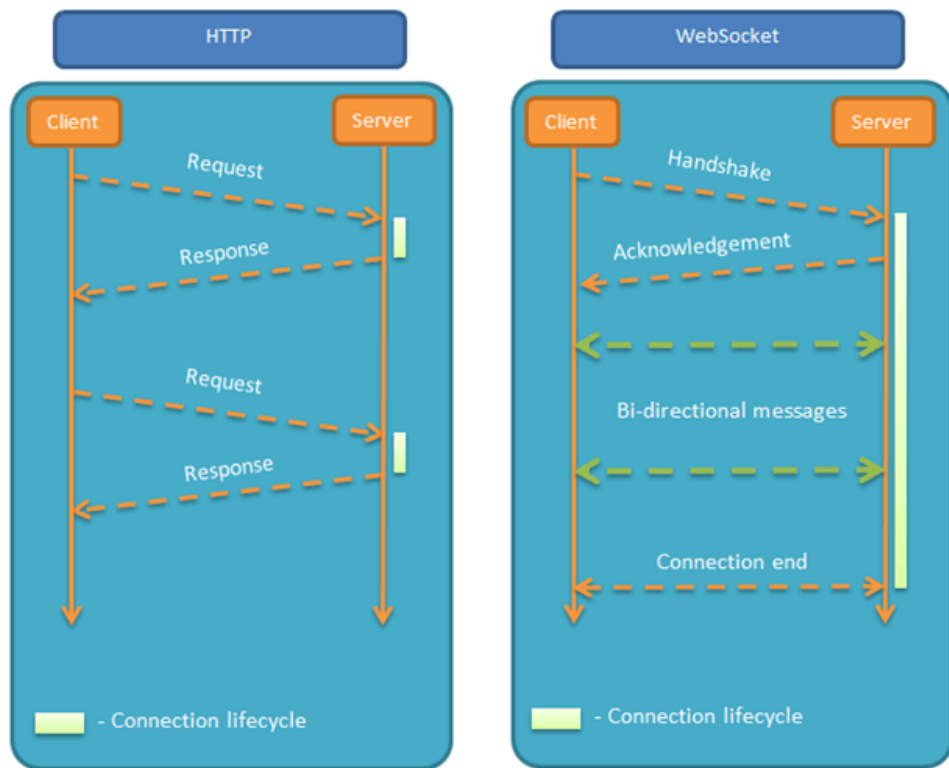
WebRTC Protocol Stack



WebRTC



WebSocket



WebSocket Example

```
Project
├── socket.io
│   └── /go/src/github.com/stmdo
│       ├── js
│       │   └── simpleGoClient_OK.js
│       ├── socket.io.js
│       ├── public
│       ├── index.html
│       ├── main.go
│       ├── External Libraries
│       └── Scratches and Consoles
└── 2. Structure

main.go x index.html x simpleGoClient_OK.js x
21
22 window.onload = function() {
23     setTimeout(handler: function () {
24         if (channel != "") {
25             console.log('Trying to create or join channel: ', channel);
26             // Send 'create or join' to the server
27             socket.emit('create or join', channel);
28         }
29     }, timeout: 100);
30 }
31
32 //Handle 'created' message
33 socket.on('created', function (channel){...});
34
35 //Handle 'full' message
36 socket.on('full', function (channel){...});
37
38 //Handle 'remotePeerJoining' message
39 socket.on('remotePeerJoining', function (channel){...});
40
41 //Handle 'joined' message
42 socket.on('joined', function (msg){...});
43
44 //Handle 'broadcast: joined' message
45 socket.on('broadcast: joined', function (msg){...});
46
47 //Handle remote logging message from server
48 socket.on('log', function (array){...});
49
50 //Handle 'message' message
51 socket.on('message', function (message){...});
52
53 //Handle 'response' message
54 socket.on('response', function (response){...});
55
56 //Handle 'Bye' message
57 socket.on('Bye', function () {...});
```

```
server, err := socketio.NewServer(transportNames: nil)
if err != nil {
    log.Fatal(err)
}

= server.On(event: "connection", func(so socketio.Socket) {
    //fmt.Println(" === connection === ")
    id := so.Request().FormValue(key: "id")
    channelID := so.Request().FormValue(key: "room_id")

    log.Println("on connection:" + id + "|room_id:" + channelID)

    // Handle 'message' messages
    = so.On(event: "message", func(message string) {
        log.Printf(format: "S --> Got message: %v", message)
        _ = so.BroadcastTo(channelID, event: "message", message, channelID)
    })

    // Handle 'create or join' messages
    err := so.On(event: "create or join", func(channel string) {
        numClients := findClientsSocket(id, namespaces)
        log.Printf(format: "numClients: %v", numClients)
        if numClients == 0 {
            so.Join(channel)
            err = so.Emit(event: "created", channel)
            if err != nil {
                log.Printf(format: "so.Emit(created) is failed! err: %v", err)
            }
            log.Printf(format: "so.Emit -> channel[%v] success!", channel)
        } else if numClients == 1 {
            err = so.Emit(event: "remotePeerJoining", channel)
            if err != nil {
                log.Printf(format: "so.Emit(remotePeerJoining) is failed! roo")
            }
            so.Join(channel)
            err = so.BroadcastTo(channel, event: "broadcast: joined", "S -->")
            if err != nil {
                log.Printf(format: "so.BroadcastTo(broadcast: joined) is fail")
            }
        } else {
            log.Println(v...: "Channel full!")
            so.Emit(event: "full", channel)
        }
    })
})

func(so socketio.Socket) func(channel string)
```



WebSocket Example

localhost:8181/public/?id=jack&room_id=1

Time: 0.393 --> Channel 1 has been created!

Time: 0.393 --> This peer is the initiator...

Time: 7.192 --> Broadcast message from server:

S --> broadcast(): client- timjoined channel-1

Time: 7.204 --> Got response from other peer:

```
{"channel":"1","message":{"channel":"1","message":{"S --> broadcast(): client- timjoined channel-1"}}
```

Time: 15.422 --> Got response from other peer:

server_response:

Time: 20.595 --> Got response from other peer:

```
{"channel":"1","message":null}
```

Time: 23.796 --> Sending "Bye" to server...

Time: 23.797 --> Going to disconnect...

The screenshot displays the browser's developer tools with the Network and Console panels open. The Network panel shows a list of messages exchanged over a WebSocket connection, including probes, broadcast messages, and responses. The Console panel shows the corresponding JavaScript logs.

Name	Headers	Messages	Cookies	Timing
Filter				
7id=jack&room_id=1&EIO=3&transport=websocket&sid=C_LDrwgUf7HE58IS8PH /socket.io				
Data				
2probe				6 20:15:12.295
3probe				6 20:15:12.297
42		[{"channel":"1","message":{"S --> broadcast(): client- timjoined channel-1"}}		81 20:15:19.090
42		[{"channel":"1","message":{"S --> broadcast(): client- timjoined channel-1"}}		98 20:15:19.092
42		[{"channel":"1","message":{"channel":"1","message":{"S --> broadcast(): client- timjoined chan..."}}		160 20:15:19.101
42		[{"channel":"1","message":{"Hello"}}		58 20:15:27.300
42		[{"channel":"1","message":{"Hello"}}		81 20:15:27.320
42		[{"channel":"1","message":null}],1]		57 20:15:32.488
42		[{"channel":"1","message":{"OK"}}		56 20:15:35.698
41		[{"channel":"1","message":{"OK"}}		13 20:15:35.698
42		[{"channel":"1","message":{"OK"}}		78 20:15:35.738
42		[{"channel":"1","message":{"OK"}}		13 20:15:35.738
41		[{"channel":"1","message":{"OK"}}		1 20:15:35.738

1 / 9 requests | 0 B / 69.0 KB transferred | 0 B / 67.7 KB resources | Finish: 23.83 s | DOMContentLoaded...

Console What's New

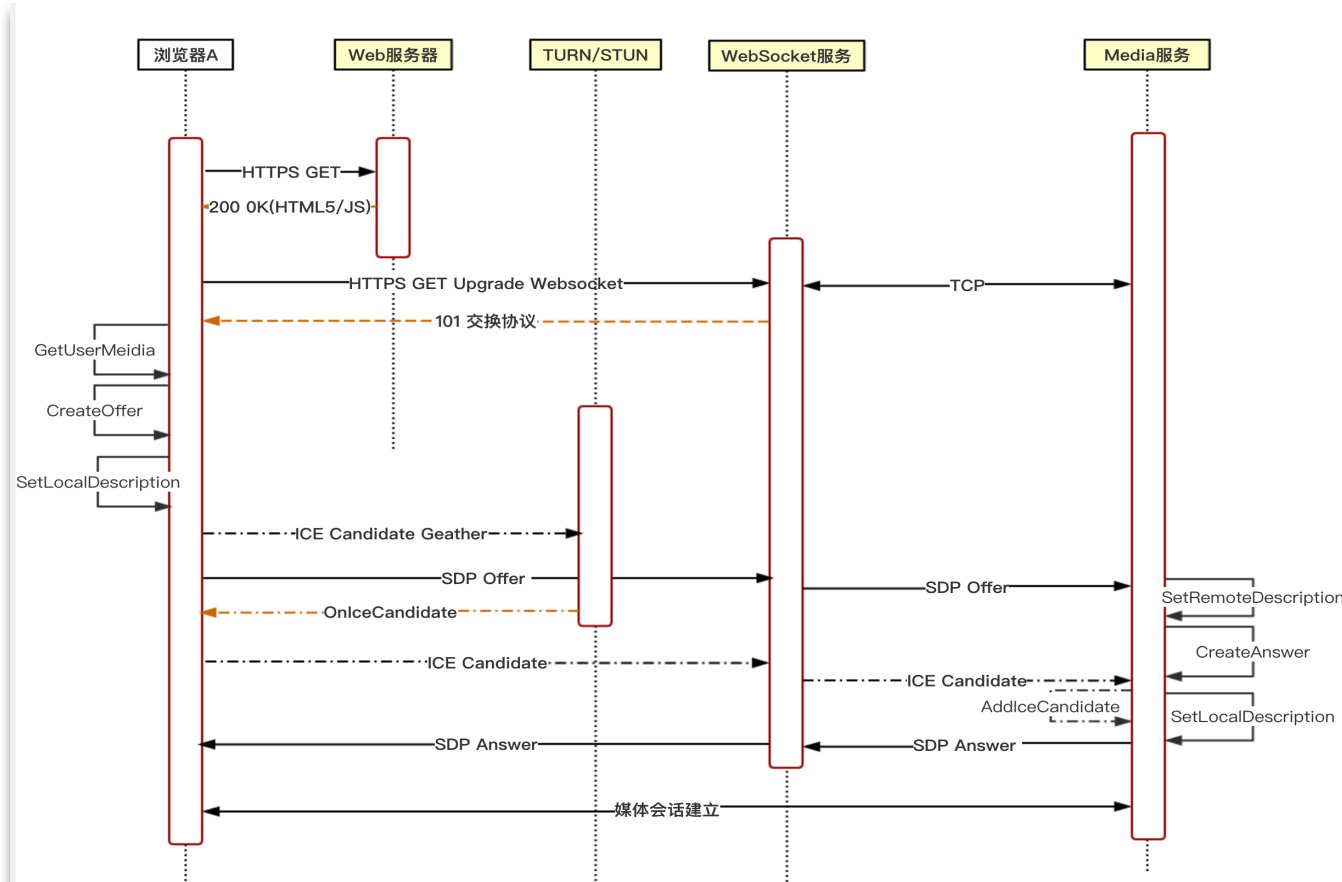
top Filter Default levels 3 hidden

```
Trying to create or join channel: 1
channel 1 has been created!
This peer is the initiator...
Broadcast message from server: 5 --> broadcast(): client- timjoined channel-1
Got response from other peer: {"channel":"1","message":{"channel":"1","message":{"S --> broadcast(): client- timjoined channel-1"}}}
Got response from other peer: server_response:
Got response from other peer: {"channel":"1","message":null}
Sending "Bye" to server
Going to disconnect...
```

simpleGoClient_OK.js:25
simpleGoClient_OK.js:36
simpleGoClient_OK.js:37
simpleGoClient_OK.js:37
simpleGoClient_OK.js:97
simpleGoClient_OK.js:142
simpleGoClient_OK.js:142
simpleGoClient_OK.js:142
simpleGoClient_OK.js:158
simpleGoClient_OK.js:165



WebRTC Flow



WebRTC Example:Front end

```
7 </head>
8
9 <body>
10
11 <div id='mainDiv'>
12
13   <table border="1" width="100%">
14     <tr>
15       <th>
16         Local video
17       </th>
18       <th>
19         'Remote' video
20       </th>
21     </tr>
22     <tr>
23       <td>
24         <video id="localVideo" autoplay></video>
25       </td>
26       <td>
27         <video id="remoteVideo" autoplay></video>
28       </td>
29     </tr>
30     <tr>
31       <td align="center">
32         <textarea rows="4" cols="60" id="dataChannelSend" disabled
33           placeholder="This will be enabled once the data channel is up.."></textarea>
34       </td>
35       <td align="center">
36         <textarea rows="4" cols="60" id="dataChannelReceive" disabled></textarea>
37       </td>
38     </tr>
39     <tr>
40       <td align="center">
41         <button id="sendButton" disabled>Send</button>
42       </td>
43     </tr>
44   </table>
45 </div>
46
47 <script src='/js/socket.io-1.3.7.js'></script>
48 <script src='js/adaptor.js'></script>
49 <script src='js/ClientWithDataChannel.js'></script>
50
51
```



WebRTC Example:Front end

```
123 // 1. Server-->Client...
124 ///////////////////////////////////////////////////
125
126 // Handle 'created' message coming back from server:
127 // this peer is the initiator
128 socket.on(event: 'created', fn: function (room){
129     console.log('Created room :: ' + room);
130     isInitiator = true;
131 });
132
133 // Handle 'full' message coming back from server:
134 // this peer arrived too late :- (
135 socket.on(event: 'full', fn: function (room){
136     console.log('Room ' + room + ' is full!');
137 });
138
139 // Handle 'join' message coming back from server:
140 // another peer is joining the channel
141 socket.on(event: 'join', fn: function (room){
142     console.log('Another peer made a request to join room ' + room);
143     console.log('This peer is the initiator of room ' + room + '!');
144     isChannelReady = true;
145 });
146
147 // Handle 'joined' message coming back from server:...
148
149 socket.on(event: 'joined', fn: function (room){
150     console.log('This peer has joined room ' + room);
151     isChannelReady = true;
152 });
153
154 // Server-sent log message...
155 socket.on(event: 'log', fn: function (array){
156     console.log.apply(console, array);
157 });
158
159 // Receive message from the other peer via the signalling server
160 socket.on(event: 'message', fn: function (message){...});
161 ///////////////////////////////////////////////////
162
163 // 2. Client-->Server
164 ///////////////////////////////////////////////////
```

```
// Receive message from the other peer via the signalling server
socket.on('message', function (message){
    console.log('Received message:', message);

    if (isJsonString(message)){
        message = JSON.parse(message)
    }else{
        message = message.replace(/\"|\"$/g, "");
    }

    if (message === 'got user media') {
        checkAndStart();
    } else if (message.type === 'offer') {
        if (!isInitiator && !isStarted) {
            checkAndStart();
        }
        pc.setRemoteDescription(new RTCSessionDescription(message));
        doAnswer();
    } else if (message.type === 'answer' && isStarted) {
        pc.setRemoteDescription(new RTCSessionDescription(message));
    } else if (message.type === 'candidate' && isStarted) {
        var candidate = new RTCIceCandidate({sdpMLineIndex:message.label,
            candidate:message.candidate});
        pc.addIceCandidate(candidate);
    } else if (message === 'bye' && isStarted) {
        handleRemoteHangup();
    }
}
});
```



WebRTC Example:Back end

```
1 package main
2
3 import (
4     "github.com/goollee/go-socket.io"
5     "log"
6     "net/http"
7 )
8
9 func main() {
10     namespaces := make(map[string]int)
11     server, err := socketio.NewServer(transportNames: nil)
12     if err != nil {
13         log.Fatal(err)
14     }
15     _ = server.On(event: "connection", func(so socketio.Socket) {...})
16
17     //http.Handle("/socket.io/", server)
18     http.HandleFunc(pattern: "/socket.io/", func(w http.ResponseWriter, r *http.Request) {
19         server.ServeHTTP(w, r)
20     })
21     http.Handle(pattern: "/", http.FileServer(http.Dir("./")))
22     http.Handle(pattern: "/js", http.FileServer(http.Dir("./js")))
23     log.Println(v...: "Serving at localhost:8282...")
24     log.Fatal(http.ListenAndServe(addr: ":8282", handler: nil))
25 }
26
27 func findClientsSocket(cID string, namespace map[string]int) int {...}
```

```
15     _ = server.On(event: "connection", func(so socketio.Socket) {
16         id := so.Request().FormValue(key: "id")
17         // Handle 'message' messages
18         _ = so.On(event: "message", func(message string) {
19             log.Printf("S --> Got message: #{message}")
20             so.BroadcastTo(room: "1", event: "message", message)
21         })
22         // Handle 'create or join' messages
23         err := so.On(event: "create or join", func(room string) {
24             numClients := findClientsSocket(id, namespaces)
25             log.Printf(format: "numClients: %v", numClients)
26             log.Printf(format: "S --> Room : %v has %v client(s)", room, numClients)
27             log.Printf(format: "S --> Request to create or join room: %v ", room)
28             // First client joining...
29             if numClients == 0 {
30                 log.Printf(format: "First client joining...: %v", numClients)
31                 err = so.Join(room)
32                 if err != nil {
33                     log.Printf(format: "so.Join[0] failed! err: %v", err)
34                 }
35                 log.Println(v...: "First client joined")
36
37                 err = so.Emit(event: "created", room)
38                 if err != nil {
39                     log.Printf(format: "so.Emit[0] created failed. err: %v", err)
40                 }
41
42                 log.Println(v...: "First client :Created")
43                 // Second client joining...
44             } else if numClients == 1 {
45                 err = so.Emit(event: "join", room)
46                 if err != nil {
47                     log.Printf(format: "so.Emit[1] send 'join' failed! err:%v", err)
48                 }
49                 err = so.Join(room)
50                 if err != nil {
51                     log.Printf(format: "so.Join[1] join failed! err: %v", err)
52                 }
53                 err = so.BroadcastTo(room, event: "join", room)
54                 if err != nil {
55                     log.Println(v...: "so.BroadcastTo[1] room join failed!")
56                 }
57                 err = so.Emit(event: "joined", room)
58             }
59         })
60     })
```

main() func(so socketio.Socket) func(message string)



Debug Tools

► Create Dump

[http://localhost:8282/?id=tim&room_id=1.\[44445-1\]](http://localhost:8282/?id=tim&room_id=1.[44445-1])

[http://localhost:8282/?id=jack&room_id=1.\[44445-2\]](http://localhost:8282/?id=jack&room_id=1.[44445-2])

GetUserMedia_Requests

http://localhost:8282/?id=tim&room_id=1, { iceServers: [stun:stun.l.google.com:19302], iceTransportPolicy: all, bundlePolicy: balanced, rtcMuxPolicy: require, iceCandidatePoolSize: 0, sdpSemantics: "unified-plan" }, [advanced: {[enableDtlsSrtp: {exact: true}], [enableRtpDataChannels: {exact: true}]]

Time	Event	Stats Tables
2019/4/11 下午8:28:04	► createLocalDataChannel	► googTrack_c1c1b857-dcc5-4a6e-8c59-d9ceb22412b8 (googTrack)
2019/4/11 下午8:28:04	► transceiverAdded	► googLibjingleSession_9203664938407920264 (googLibjingleSession)
2019/4/11 下午8:28:04	► createOffer	► bweforvideo (VideoBwe)
2019/4/11 下午8:28:04	negotiationneeded	► googCertificate_1C:D5:18:C4:6B:96:63:E1:88:AD:3C:2A:D9:3F:7B:3B:30:68:5A:1E:71:EF:31:A7:75:50:A (googCertificate)
2019/4/11 下午8:28:04	negotiationneeded	► Channel-0-1 (googComponent)
2019/4/11 下午8:28:04	► createOfferOnSuccess	► Cand-Kv+2/Me4 (localcandidate)
2019/4/11 下午8:28:04	► setLocalDescription	► Channel-1-1 (googComponent)
2019/4/11 下午8:28:04	► signalingstatechange	► Cand-JHZ/qaAz (localcandidate)
2019/4/11 下午8:28:04	► transceiverModified	► Channel-2-1 (googComponent)
2019/4/11 下午8:28:04	► transceiverModified	► Cand-mdMHHWHG (localcandidate)
2019/4/11 下午8:28:04	► transceiverModified	► ssrc_4263543805_send (ssrc)
2019/4/11 下午8:28:04	setLocalDescriptionOnSuccess	► googTrack_a039b6ca-424f-4f5e-bd80-c2ac32e5c15d (googTrack)
2019/4/11 下午8:28:04	► icegatheringstatechange	► googCertificate_39:DF:D3:CD:7D:C5:52:F3:82:2F:49:8C:B5:26:59:A8:2E:65:52:E2:BC:1C:D8:48:1F:EI (googCertificate)
2019/4/11 下午8:28:04	► setRemoteDescription	► Cand-tPLCSVNH (localcandidate)
2019/4/11 下午8:28:04	► iceconnectionstatechange	► Conn-0-1-0 (googCandidatePair)
2019/4/11 下午8:28:04	► signalingstatechange	► Cand-KLTqGf8W (remotecandidate)
2019/4/11 下午8:28:04	► transceiverModified	► ssrc_3100507769_recv (ssrc)
2019/4/11 下午8:28:04	► transceiverModified	
2019/4/11 下午8:28:04	setRemoteDescriptionOnSuccess	
2019/4/11 下午8:28:04	► icecandidate (host)	
2019/4/11 下午8:28:04	► icecandidate (host)	
2019/4/11 下午8:28:04	► icecandidate (host)	
2019/4/11 下午8:28:04	► icecandidate (host)	
2019/4/11 下午8:28:04	► addIceCandidate (host)	
2019/4/11 下午8:28:04	► icegatheringstatechange	
2019/4/11 下午8:28:04	► iceconnectionstatechange	
2019/4/11 下午8:28:04	► iceconnectionstatechange	
2019/4/11 下午8:28:04	complete	

- Stats graphs for bweforvideo (VideoBwe)
- Stats graphs for Cand-Kv+2/Me4 (localcandidate)
- Stats graphs for Cand-JHZ/qaAz (localcandidate)
- Stats graphs for Cand-mdMHHWHG (localcandidate)
- Stats graphs for ssrc_4263543805_send (ssrc) (video)
cname: E6TmC.88Nj2i9j9HM
- Stats graphs for Cand-tPLCSVNH (localcandidate)
msid: FRH99WHwAmObkNZD59s10ErNAwTuZ5Pwe4h1 c1c1b857-dcc5-4a6e-8c59-d9ceb22412b8
mslabel: FRH99WHwAmObkNZD59s10ErNAwTuZ5Pwe4h1
label: c1c1b857-dcc5-4a6e-8c59-d9ceb22412b8
- Stats graphs for Cand-tPLCSVNH (localcandidate)
- Stats graphs for Conn-0-1-0 (googCandidatePair)
- Stats graphs for Cand-KLTqGf8W (remotecandidate)
- Stats graphs for ssrc_3100507769_recv (ssrc) (video)
cname: YWzHTSGpNwC.7Y7T

<chrome://webrtc-internals/>



Demo Result



Network tab showing a table of connections:

Name	Status	Type	Initiator	Size	Time	Waterfall
7id=jack&room_id=1&EO=3&transport=websocket&id=VhSpqU... /socket.io	101 Switching Protocols	websocket	socket.io-1.3.7.js#3 Script		0 B 0 B	Pending

1 / 8 requests | 0 B / 108 KB transferred | 0 B / 107 KB resources | Finish: 372 ms | DOMContentLoaded: 322 ms | Load: 295 ms

Console: What's New

Expression not available

```
urn:ietf:params:rtp-hdrext:ssrc-audio-level\r\nname:tpmap:13 urn:ietf:params:rtp-hdrext:sdes:mid\r\nname:tpmap:14 urn:ietf:params:rtp-hdrext:sdes:cpair:red-rtp-stream-id\r\nname:recv\r\nname:tcp-mux\r\nname:tpmap:111 opus/48000/2\r\nname:tcp-frag:121 transport-cc\r\nname:frags:121 m1p:line=10;useinbandfec=1\r\nname:tpmap:103 ISAC/16000\r\nname:tpmap:104 ISAC/32000\r\nname:tpmap:5 722/8000\r\nname:tpmap:8 PCM/8000\r\nname:tpmap:106 QM/32000\r\nname:tpmap:105 QM/16000\r\nname:tpmap:13 QM/8000\r\nname:tpmap:111 telephone-event/48000\r\nname:tpmap:112 telephone-event/32000\r\nname:tpmap:113 telephone-event/16000\r\nname:tpmap:126 telephone-event/8000\r\nname:application 9 UDP/TLV/8379/SAVPF 118\r\nname:IP4 0.0.0.0\r\nname:tcp:9 IN IP4 0.0.0.0\r\nname:ice-ufrag:Zur\r\nname:ice-pwd:27UkDXnFTZv4TFS9eW0rhq\r\nname:ice-options:trickle\r\nname:fingerprint:sha-256 49:FF:66:45:0D:92:EC:07:08:1E:0A:3B:BC:CF:41:10:10:75:26:19:00:2A:5E:76:4F:4C:63:4E:5E:73:24:49:\r\nname:setuptcpactsps\r\nname:tcp:2\r\nname:sendrecv\r\nname:msid:sendDataChannel sendDataChannel\r\nname:tcp-mux\r\nname:tpmap:118 google-data/90000\r\nname:ssrc:2980273988 cname:g1jD2RHkcdAEZ\r\nname:ssrc:2980273988 msid:sendDataChannel sendDataChannel\r\nname:ssrc:2980273988 msLabel:sendDataChannel\r\nname:ssrc:2980273988 label:sendDataChannel\r\n"}  
Created RTCPeerConnection with:  
  config: {"urls":["stun:stun.l.google.com:19302"]};  
  constraints: {"optional":[{"urls":[{"DtlsSrtpKeyAgreement":true}],"RtpDataChannels":true}}.  
done:handleRemoteStreamAdded  
Sending answer to peer.  
Sending message:  
  RTCSessionDescription (type: "answer", sdp: "v=0 o=- 452278985887196246 2 IN IP4 127.0.0.1 s=90000 a=ssrc:1688559292 cname:cn/EUE/ZP19YnTm")  
0.568: Receive Channel Callback  
Remote stream added.  
handleIceCandidate event:  
  RTCPeerConnectionIceEvent (isTrusted: true, candidate: RTCIceCandidate, type: "icecandidate", target: RTCPeerConnection, currentTarget: RTCPeerConnection, ...)  
Sending message:  
  type: "candidate", label: "", id: "", candidate: "candidate:395635446 1 udp 2122260223 10.0.0.66 526.0.0.0 ufrag 60y2 network-id 1 network-cost 10"  
End of candidates.  
handleIceCandidate event:  
  RTCPeerConnectionIceEvent (isTrusted: true, candidate: null, type: "icecandidate", target: RTCPeerConnection, currentTarget: RTCPeerConnection, ...)  
End of candidates.  
Received message: {"type":"candidate","label":"","id":"","candidate":"candidate:395635446 1 udp 2122260223 10.0.0.66 54265 typ host generation 0 ufrag uZur network-id 1 network-cost 10"}  
Received message: {"type":"candidate","label":"1","id":"1","candidate":"candidate:395635446 1 udp 2122260223 10.0.0.66 53843 typ host generation 0 ufrag uZur network-id 1 network-cost 10"}  
Received message: {"type":"candidate","label":"2","id":"2","candidate":"candidate:395635446 1 udp 2122260223 10.0.0.66 59819 typ host generation 0 ufrag uZur network-id 1 network-cost 10"}  
Received message: {"type":"candidate","label":"","id":"","candidate":"candidate:1494685190 1 tcp 1518280447 10.0.0.66 9 typ host tcpType active generation 0 ufrag uZur network-id 1 network-cost 10"}  
}
```



SDP

```
....  
o=- 20518 0 IN IP4 0.0.0.0  
.....  
a=ice-options:trickle  
a=group:BUNDLE audio video  
  
m=audio 54609 UDP/TLS/RTP/SAVPF 109 0 8  
a=mid:audio  
a=sendrecv  
a=rtpmap:109 opus/48000/2  
.....  
a=candidate:0 1 UDP 2122194687 192.168.1.4 61665 typ host  
a=candidate:1 1 UDP 1685987071 24.23.204.141 54609 typ srflx raddr  
192.168.1.4 rport 61665  
.....
```

来源信息 用户 会话id 版本
id 网络类型 ip类型 ip

采用trickle方式
媒体数据包含音频和视频

媒体信息描述

[\[RFC3264\]](#)can send and
recv audio
[\[I-D.ietf-payload-rtp-opus\]](#)
- Opus Codec 48khz, 2
channels

[\[RFC5245\]](#) - ICE Host
Candidate
[\[RFC5245\]](#) - ICE Server
Reflexive Candidate

会话描述

```
1 v = (协议版本)  
2 o = (发起者和会话标识符)  
3 s = (会话名称)  
4 i = * (会话信息)  
5 u = * (描述的URI)  
6 e = * (电子邮件地址)  
7 p = * (电话号码)  
8 c = * (连接信息 - 如果包含在内, 则不需要所有媒体)  
9 b = * (零个或多个带宽信息行)  
10  
11 一个或多个时间描述 ("t ="和"r ="行; 见下文)  
12 z = * (时区调整)  
13 k = * (加密密钥)  
14 a = * (零个或多个会话属性行) 零个或多个媒体描述
```

时间描述

```
1 t = (会话活动时间)  
2 r = * (零个或多个重复次数)
```

媒体描述, 如果存在

```
1 m = (媒体名称和传输地址)  
2 i = * (媒体标题)  
3 c = * (连接信息 - 如果包含在内, 则为可选项会话级别)  
4 b = * (零个或多个带宽信息行)  
5 k = * (加密密钥)  
6 a = * (零个或多个媒体属性行)
```



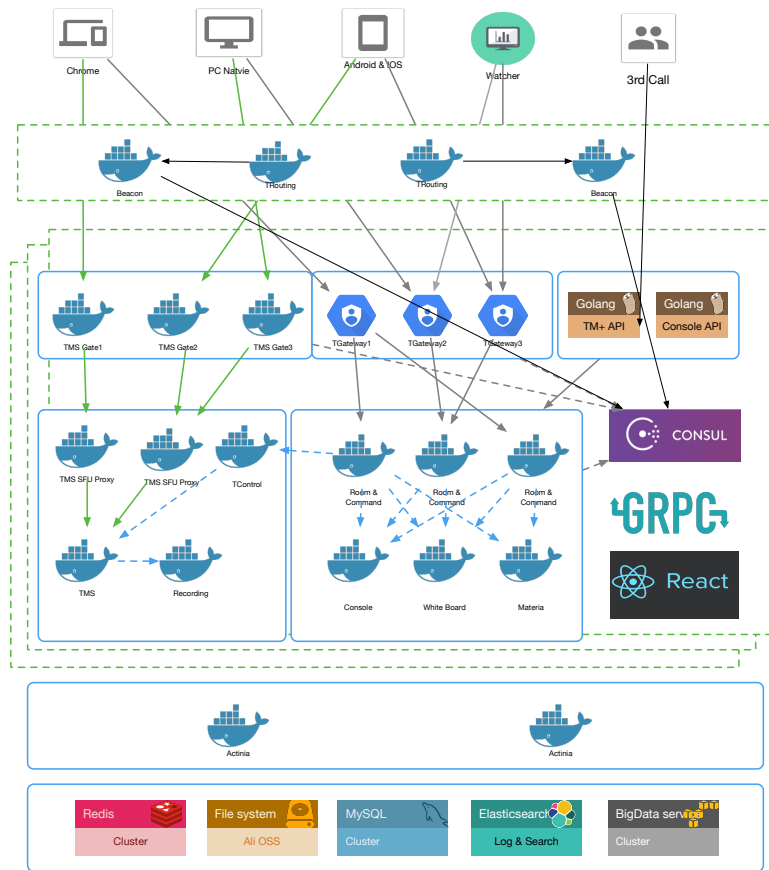
WebRTC Server

Open Source Media Server :

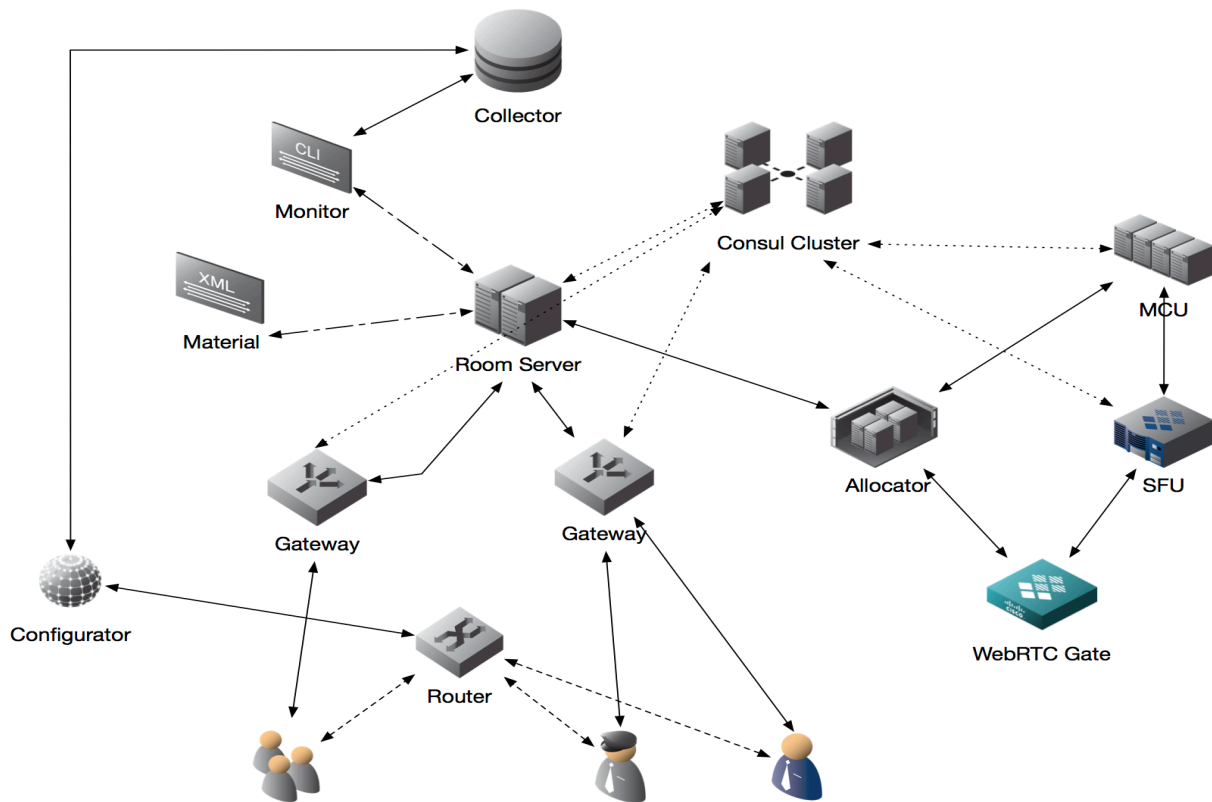
- lynckia/licode : github.com/lynckia/licode/ 【C++】
- meetecho (janus) : github.com/meetecho/janus-gateway 【C】
- jitsi (meetme) : github.com/jitsi/jitsi 【Java】
- Kurento : github.com/Kurento/kurento-media-server 【C++】



TM+ Architecture I

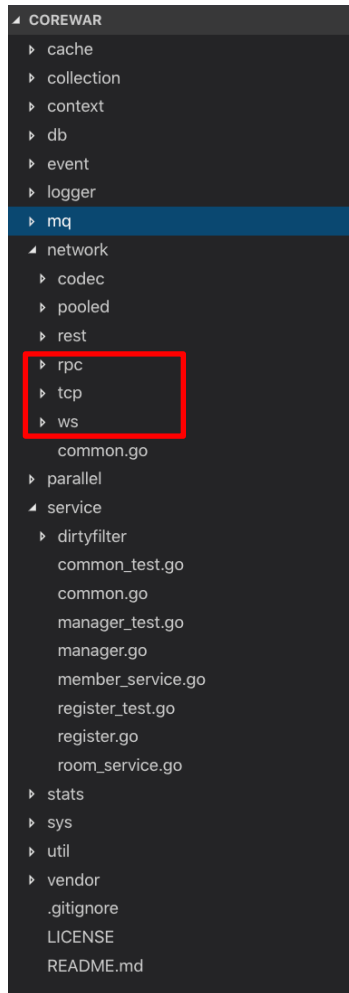


TM+ Architecture II



Go Lib

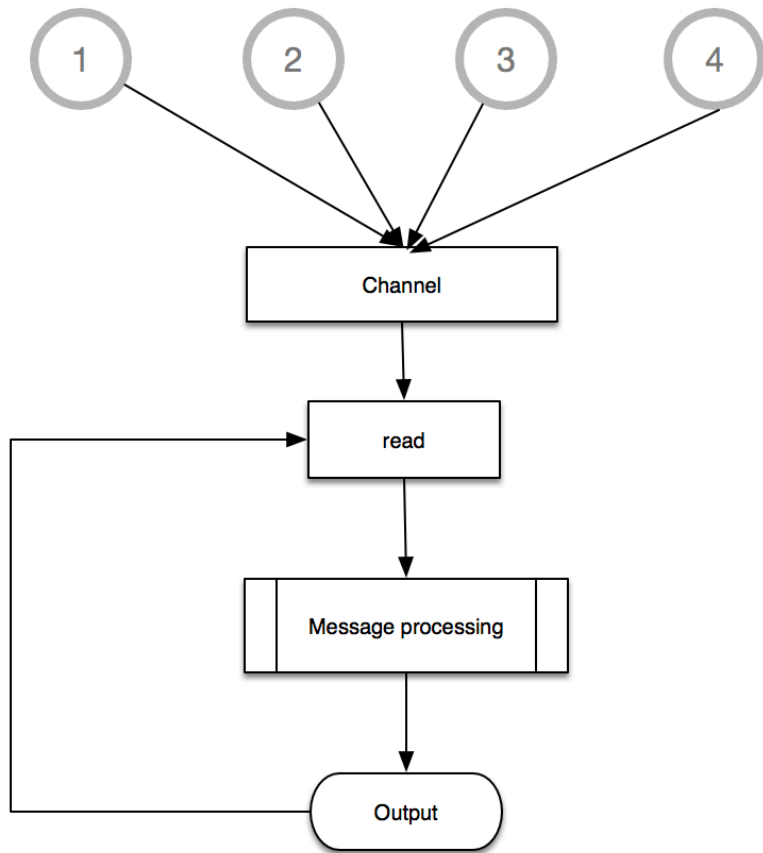
- gRPC 、 TCP
- Websocket 、 Socket.IO
- Gin
- Redigo(Redis)
- Mgo(MongoDB)
- Consul
-



Problem I

loop issue

```
for {  
  select {  
    case msg := <- stream:  
      handle(msg) // message处理  
  }  
}
```



Problem I

loop issue

Solution A :

```
for {
    select {
        case msg := <-stream:
            messages := []Message{msg}
            for i := 0; i < len(stream); i++ {
                messages = append(messages, <-
stream)
            }
            handles(messages) // 批处理
    }
}
```



Problem I

loop issue

Solution B :

```
sema := make(chan *Message, 16) //  
for {  
    select {  
    case msg := <-stream:  
        sema <- msg  
        go func(sem <-chan *Message) {  
            handle(<-sem)  
        }(sema)  
    }  
}
```



Problem II

- **Go Time :**
- ```
p := fmt.Println
thisday,_ := time.Parse(time.RFC3339,"2018-11-30T14:00:00+08:00")
tomorrow,_ := time.Parse(time.RFC3339,"2018-12-01T14:00:00+08:00")
time2 := thisday.Format("2018-01-01 00:00:00")
p(time2)
time3 := tomorrow.Format("2018-01-01 00:00:00")
p(time3)
```
- **Result:**
- `time2 =30118-11-11 00:00:001128-12-12 00:00:00`
- `time3 =1128-12-12 00:00:00`



# Problem II

Go Time :

Why “**2006-01-02 15:04:05**”

2006年1月2日3点4分5秒~

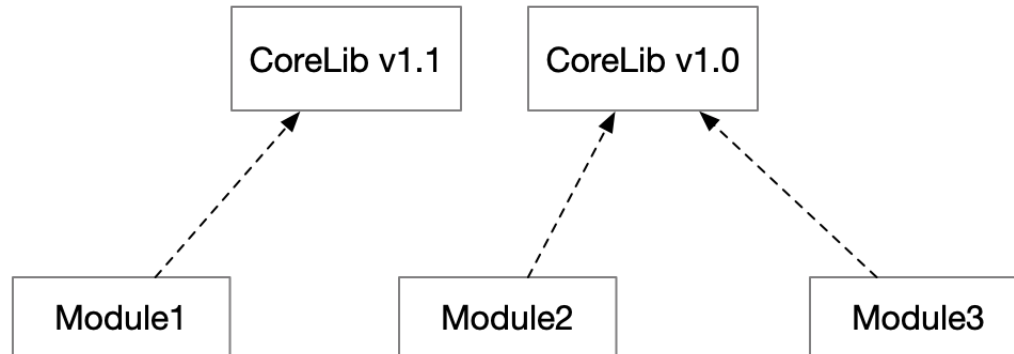
Format:

**2006-01-02 15:04:05.999999999 -0700 MST**

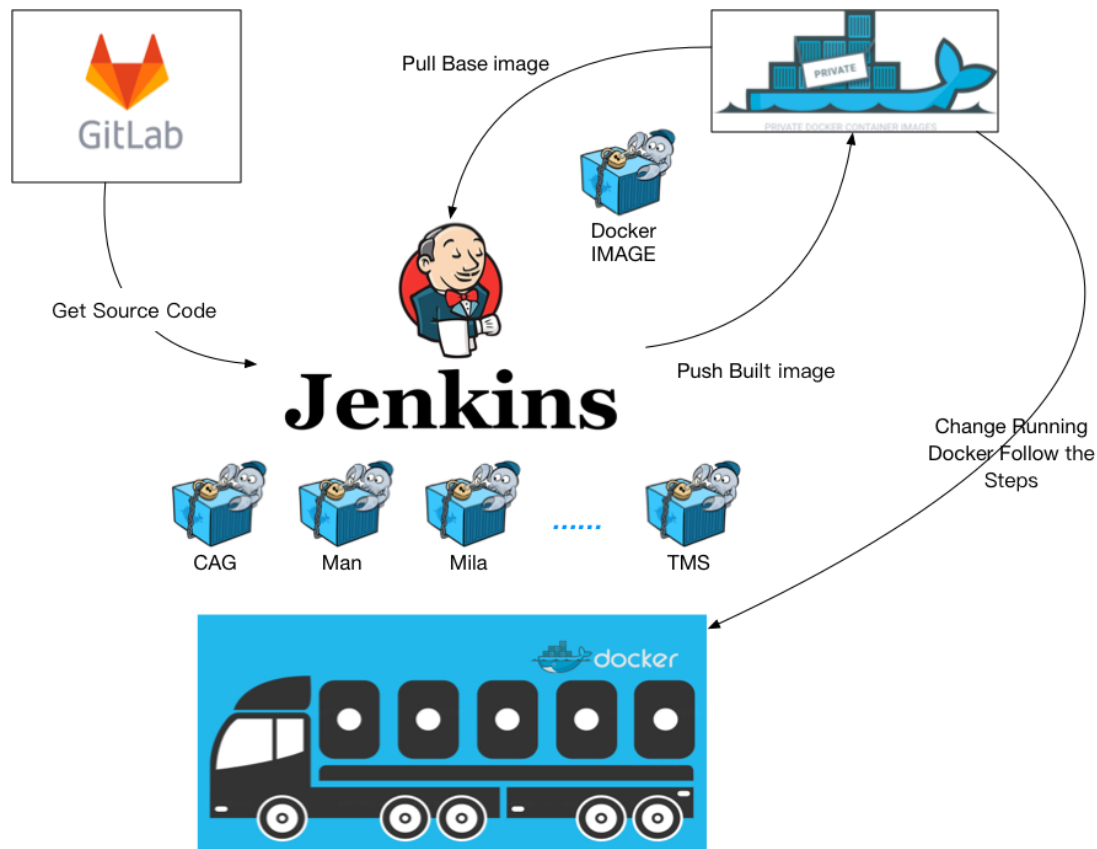


# Problem III

- Go module:

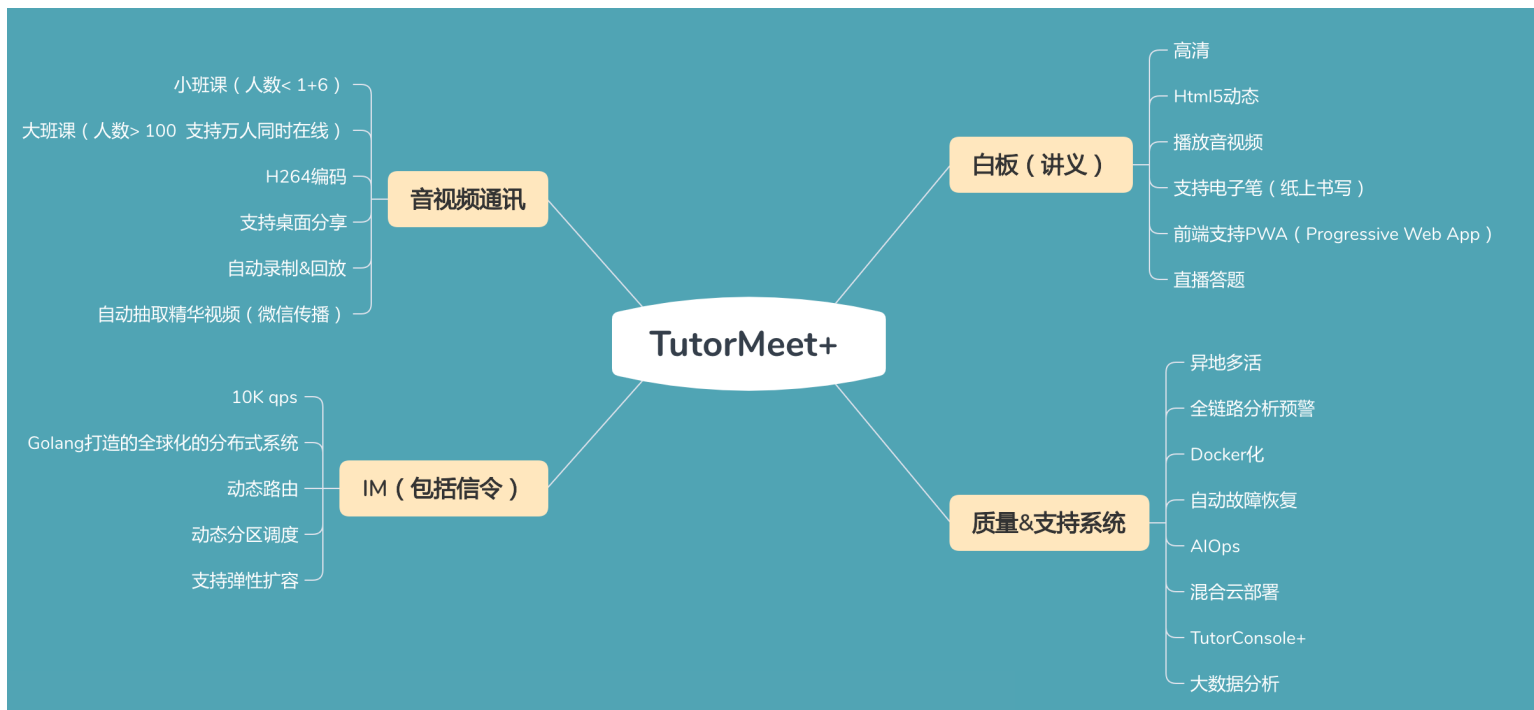


# TM+ Deployment





# TM+ Module

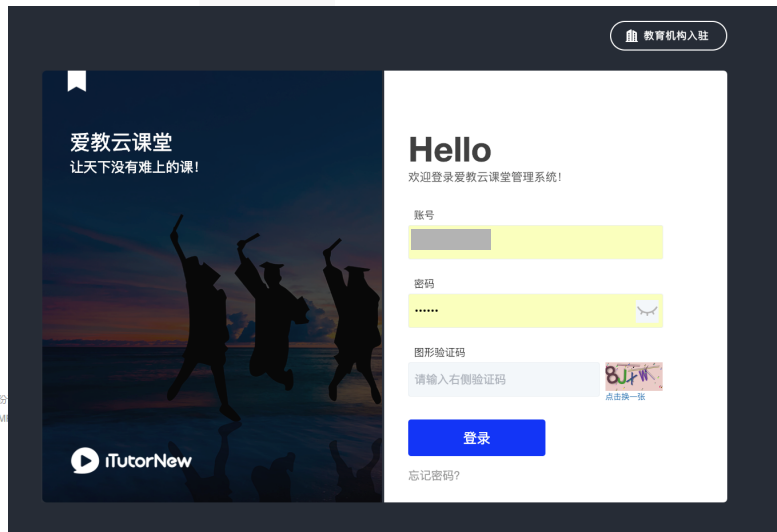
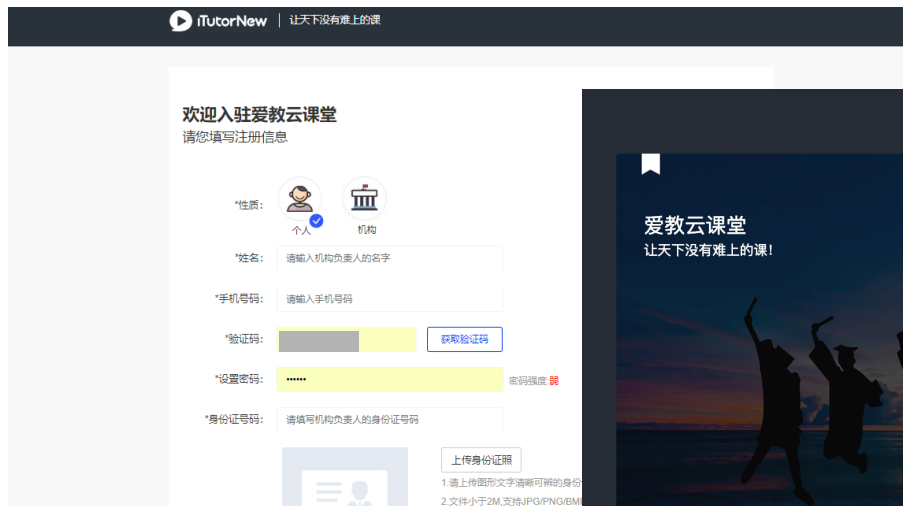


# SaaS

# Real-Time Interaction



# SaaS



机构端：<https://open.cntutormeet.com/hlogin.html>  
教师端：<https://open.cntutormeet.com/tlogin.html>  
学生端：微信小程序，在微信中搜索小程序“爱教云课堂”  
或微信扫描下方二维码：



O'REILLY®



# Real-Time Communication with WebRTC

PEER-TO-PEER IN THE BROWSER

Salvatore Loreto &  
Simon Pietro Romano

# Thanks !

# Q&A

