HYPER-LINKED COMMUNICATIONS

WebRTC enabled asynchronous collaboration

Sunday 29th May, 2016

Henrique Rocha

Instituto Superior Técnico
Universidade de Lisboa
henrique.rocha@tecnico.ulisboa.pt

Advisor: Ricardo Pereira
Co-Advisor: Paulo Chainho

OVERVIEW

1. Introduction
2. Related Work
3. Architecture
4. Implementation
5. Evaluation
6. Conclusions
7. Future Work

INTRODUCTION

CONTEXT

Written communication could never replace face to face communication.

> “No computer in our lifetimes will ever rival a human voice’s capacity to conveying rich and complex social and emotional meaning”
>
> — Geddes, Martin

Today, we can achieve more.

PROBLEM STATEMENT

Real-time communication applications can make a difference on business, education and health sectors.

An application that provides a collaborative environment and a way to remember our past communications would be a strong tool.

THESIS GOALS

Allow multi party conference calls.

Record and playback interactive video.

Create a collaborative environment

Use only standard technologies like JavaScript, WebRTC, HTML5 and CSS3.

RELATED WORK

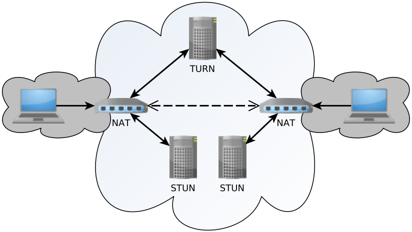
STATE OF THE ART OVERVIEW

1. Connection Establishment
2. Streaming audio and video
3. Overlay communications with content
4. Collaboration Environment

CONNECTION ESTABLISHMENT

○ Network Address Translation

○ STUN + TURN = ICE



STREAMING AUDIO AND VIDEO

WebRTC (Web Real-Time Communications)

- ○ Access to camera, microphone and screen*
- ○ Peer to Peer file and stream sharing
- ○ Standardized protocols
- ○ No plug-ins required

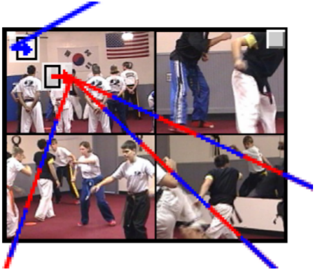
* requires installing a plug-in yet.

TÉCNICO LISBOA

OVERLAY COMMUNICATIONS WITH CONTENT

○ **Concepts:** HyperText & HyperMedia & HyperCommunications & Detail on Demand

○ **Implementations:** HyperCafe & HyperHitchcock



COLLABORATION ENVIRONMENT

Table: Comparision between Operational Transformation libraries

Library	Own Server	Own Storage	Operations
ShareJS	✓	✓	text+objects
TogetherJS	✓	✗	text+objects
Goodow	✓	✓	text+objects
Etherpad Lite	✓	✓	extendable
OT.js	✗	✗	text

OVERVIEW

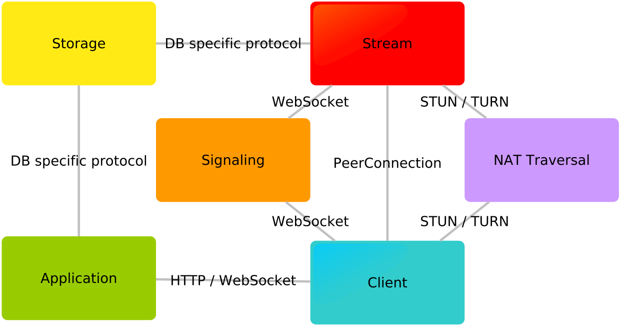
Name	Skype	Hangouts	Jitsi	Kurento	Our proposal
Technology	Proprietary	WebRTC[1]	WebRTC	WebRTC	WebRTC
Development	✗	✓[2]	✓	✓	✓
Audio/Video	✓	✓	✓	✓	✓
Text Messaging	✓	✓	✓	✗	✓
Collaborative Editor	✗	✓	✓	✗	✓
File sharing	✓	✓	✓	✗	✓
Recording & Playback	✗	✗	✗	✓	✓
Interactive Content	✗	✗	✗	✗	✓

[1] requires installing a plug-in on non chrome web browsers.

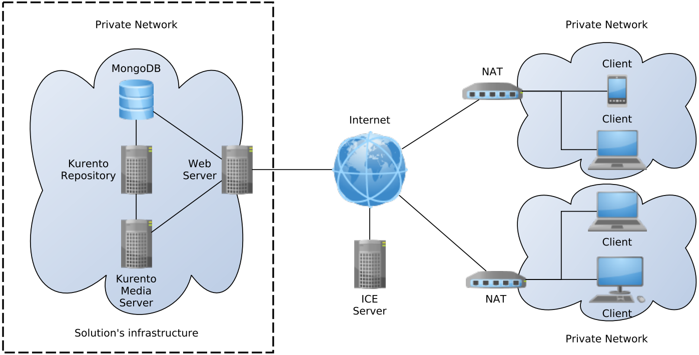
[2] allows the development of extensions.

ARCHITECTURE

MODULES

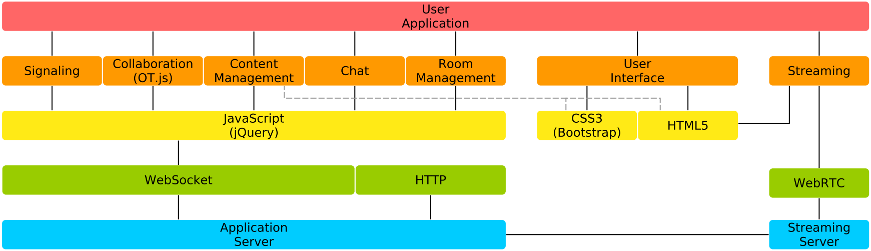

Storage
DB specific protocol
Stream
WebSocket
STUN / TURN
Signaling
DB specific protocol
PeerConnection
NAT Traversal
WebSocket
STUN / TURN
Application
HTTP / WebSocket
Client

SYSTEM INFRASTRUCTURE



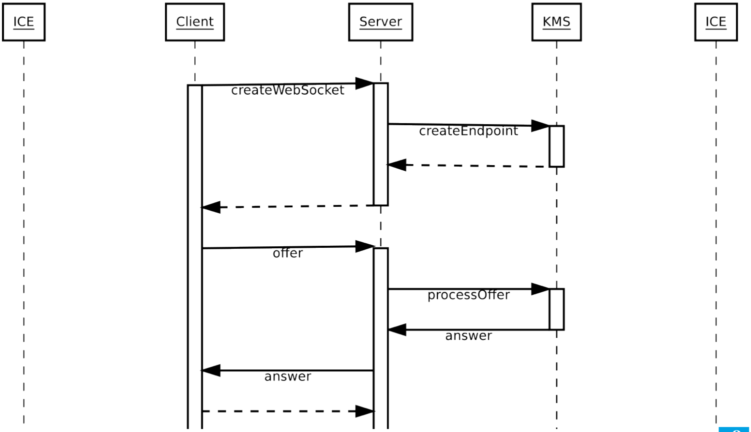
- **Signaling Server & Web Server**: Play Framework
- **Stream Server**: Kurento Media Server
- **Storage**: MongoDB & Kurento Repository
- **NAT Traversal**: Public STUN Servers

APPLICATION ARCHITECTURE

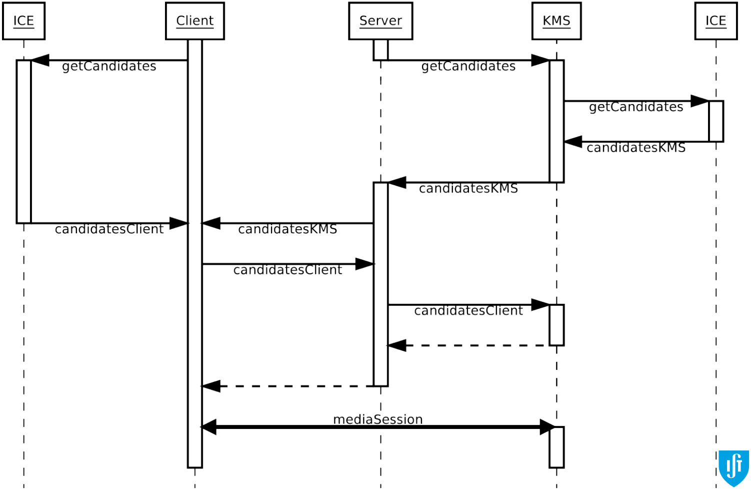


IMPLEMENTATION

SIGNALING PROTOCOL - PART 1



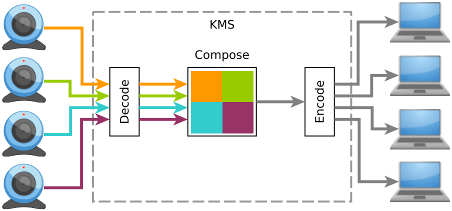
SIGNALING PROTOCOL - PART 2

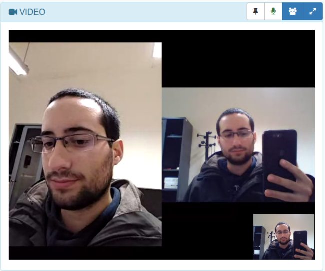


STREAM OPERATIONS

○ Server-side recording to database (Kurento Repository).

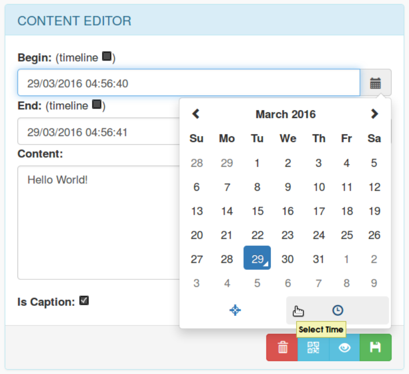
○ Server-side stream composition.





HYPER-CONTENT

- ○ Create & Search content
- ○ Scheduler
- ○ QR codes
- ○ Security concerns



CONTENT EDITOR

Begin: (timeline ■)

29/03/2016 04:56:40

End: (timeline ■)

29/03/2016 04:56:41

Content:

Hello World!

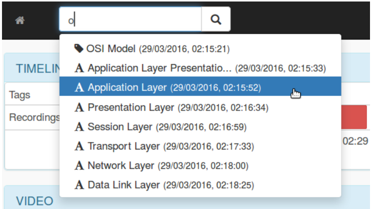
Is Caption: ☑

March 2016

Su	Mo	Tu	We	Th	Fr	Sa
28	29	1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	31	1	2
3	4	5	6	7	8	9

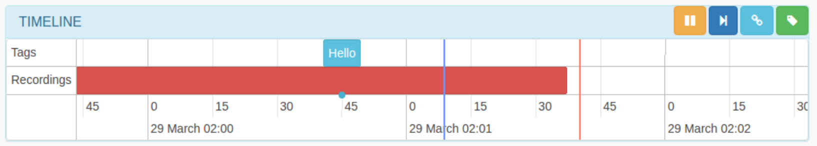
TIME MANIPULATION

- Playback recordings
- Create & Search annotations
- Time Hyper-links



o

- OSI Model (29/03/2016, 02:15:21)
- A Application Layer Presentatio... (29/03/2016, 02:15:33)
- A Application Layer (29/03/2016, 02:15:52)
- A Presentation Layer (29/03/2016, 02:16:34)
- A Session Layer (29/03/2016, 02:16:59)
- A Transport Layer (29/03/2016, 02:17:33)
- A Network Layer (29/03/2016, 02:18:00)
- A Data Link Layer (29/03/2016, 02:18:25)



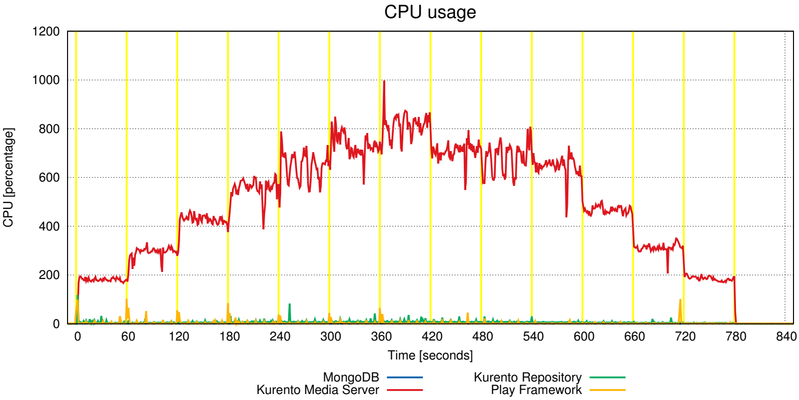
CHAT & COLLABORATIVE ENVIRONMENT

- Instant text messaging
  - WebSockets
- File sharing
  - HTTP file upload
  - stored in the database
- Collaborative text editor (OT.js)
  - retain
  - insert
  - delete

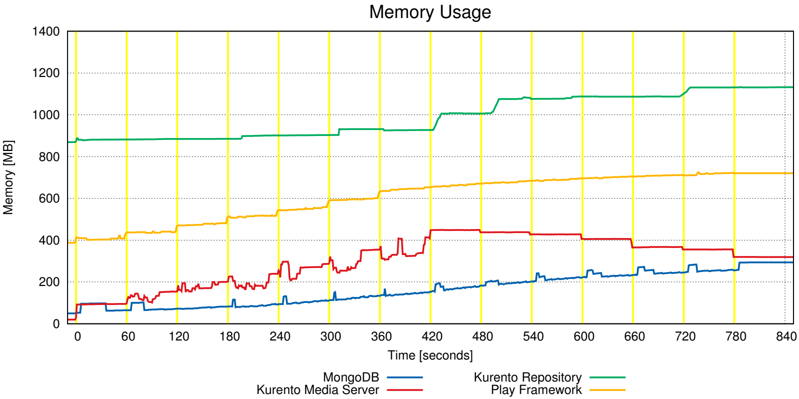
DEMONSTRATION

EVALUATION

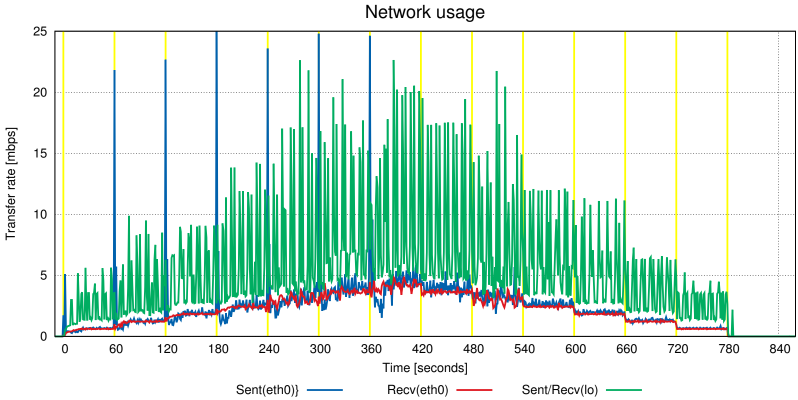
PERFORMANCE TESTS AT SERVER - CPU USAGE



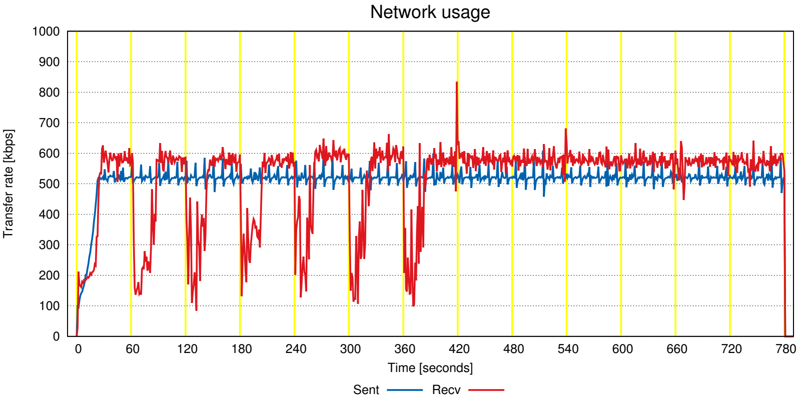
PERFORMANCE TESTS AT SERVER - MEMORY USAGE



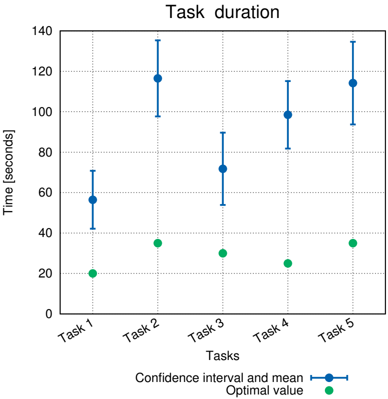
PERFORMANCE TESTS AT SERVER - NETWORK USAGE

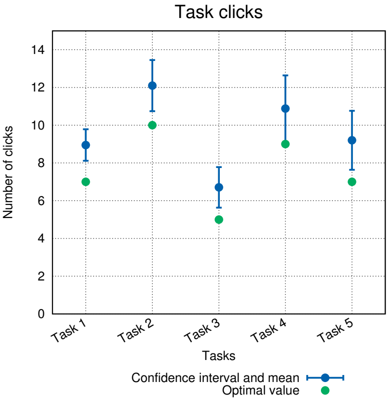


PERFORMANCE TESTS AT CLIENT - NETWORK USAGE



FIVE TASKS

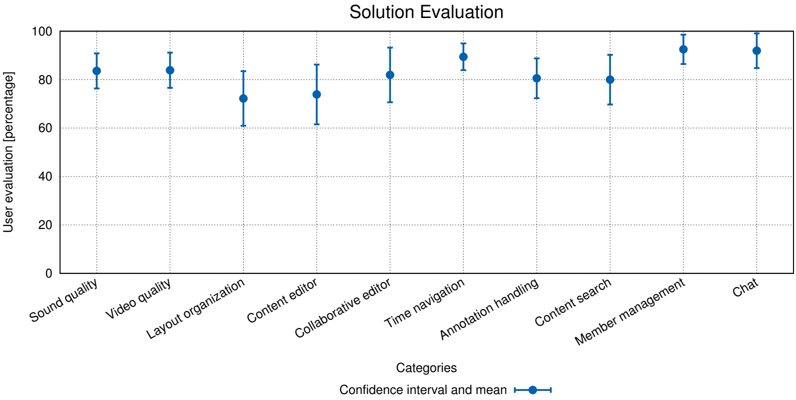




○ Difficulty per task.

○ Errors per task.

OVERALL EVALUATION



CONCLUSIONS

CONCLUSIONS

○ New usage scenarios for communication and collaboration applications.

○ Enrich communications using hypermedia concepts. Record, playback and collaboration features.

○ Prototype implementation and testing.

FUTURE WORK

FUTURE WORK

○ Implement fast-forward playback.

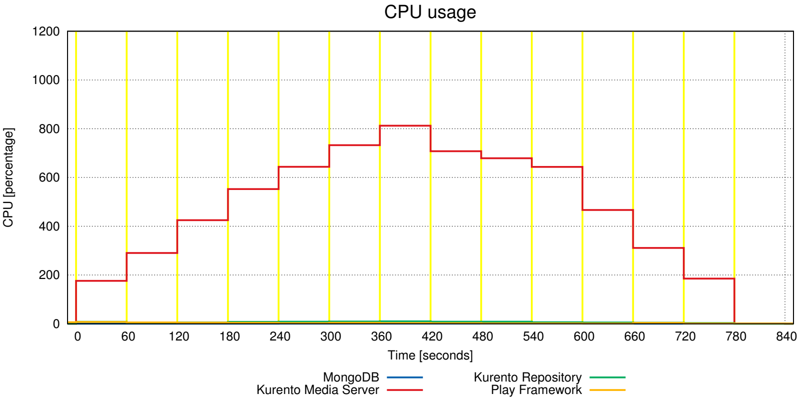
○ Improve solution’s security.

○ Scale our solution to multiple servers.

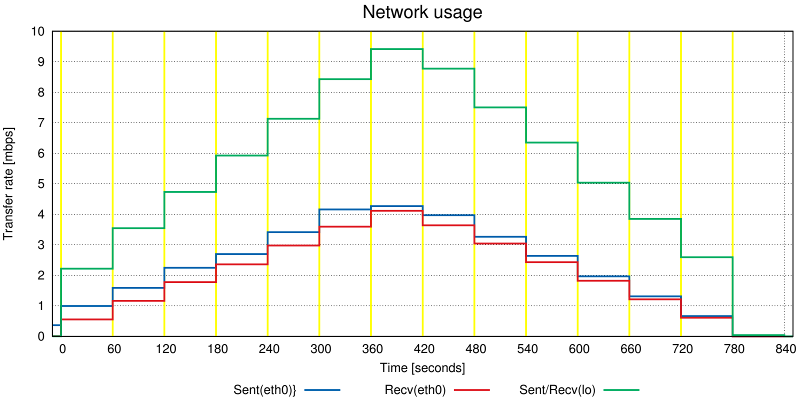
Questions?

TÉCNICO LISBOA

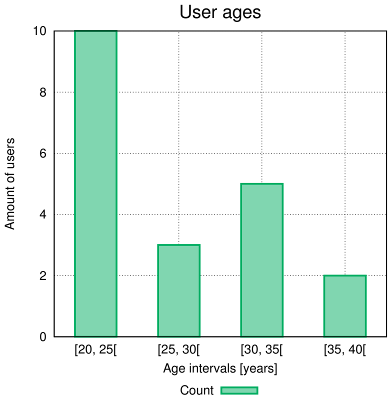
PERFORMANCE TESTS - CPU (AVERAGE)



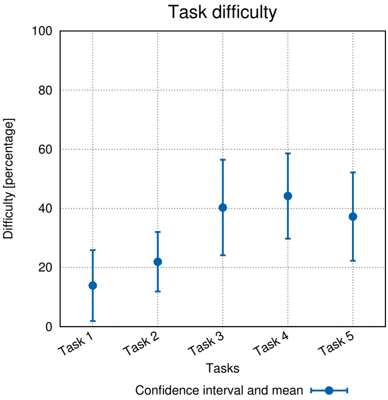
PERFORMANCE TESTS - NETWORK USAGE (AVERAGE)

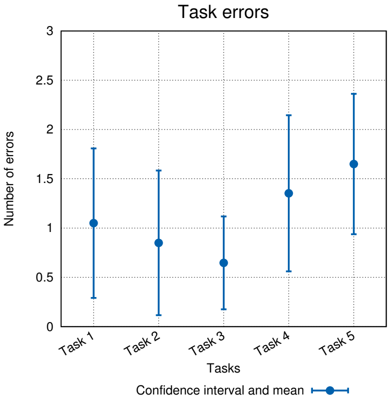


USER INTERFACE TESTS

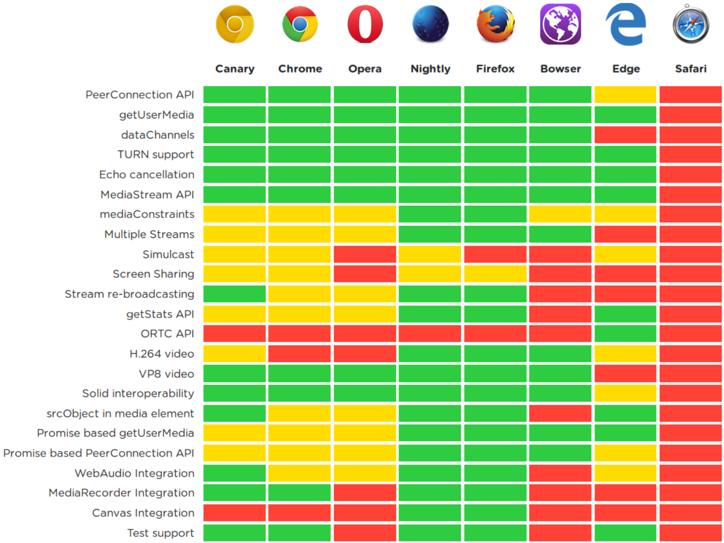


FIVE TASKS


Task difficulty
Difficulty [percentage]
100
80
60
40
20
0
Task 1
Task 2
Task 3
Task 4
Task 5
Tasks
Confidence interval and mean


Task errors
Number of errors
3
2.5
2
1.5
1
0.5
0
Task 1
Task 2
Task 3
Task 4
Task 5
Tasks
Confidence interval and mean

WEB BROWSER SUPPORT



	Canary	Chrome	Opera	Nightly	Firefox	Bowser	Edge	Safari
PeerConnection API	🟩	🟩	🟩	🟩	🟩	🟩	🟨	🟥
getUserMedia	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟥
dataChannels	🟩	🟩	🟩	🟩	🟩	🟩	🟥	🟥
TURN support	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟥
Echo cancellation	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟥
MediaStream API	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟥
mediaConstraints	🟨	🟨	🟨	🟩	🟩	🟨	🟨	🟥
Multiple Streams	🟨	🟨	🟨	🟩	🟩	🟩	🟥	🟥
Simulcast	🟨	🟨	🟥	🟨	🟥	🟥	🟨	🟥
Screen Sharing	🟨	🟨	🟥	🟨	🟨	🟥	🟥	🟥
Stream re-broadcasting	🟩	🟨	🟨	🟩	🟩	🟥	🟥	🟥
getStats API	🟨	🟨	🟨	🟩	🟩	🟥	🟩	🟥
ORTC API	🟥	🟥	🟥	🟥	🟥	🟥	🟩	🟥
H.264 video	🟨	🟥	🟥	🟩	🟩	🟩	🟨	🟥
VP8 video	🟩	🟩	🟩	🟩	🟩	🟩	🟥	🟥
Solid interoperability	🟩	🟩	🟩	🟩	🟩	🟩	🟨	🟥
srcObject in media element	🟩	🟨	🟨	🟩	🟩	🟥	🟩	🟥
Promise based getUserMedia	🟨	🟨	🟨	🟩	🟩	🟩	🟩	🟥
Promise based PeerConnection API	🟨	🟨	🟨	🟩	🟩	🟩	🟨	🟥
WebAudio Integration	🟩	🟨	🟨	🟩	🟩	🟥	🟨	🟥
MediaRecorder Integration	🟩	🟩	🟥	🟩	🟩	🟥	🟥	🟥
Canvas Integration	🟥	🟥	🟥	🟩	🟩	🟥	🟥	🟥
Test support	🟩	🟩	🟥	🟩	🟩	🟥	🟩	🟥

TÉCNICO LISBOA

SECURITY CONCERNS

- Subset of actions implemented and triggered by messages.
- Defined actions on demand, requires detecting malicious code.
  - EarlyBird - machine learning techniques.

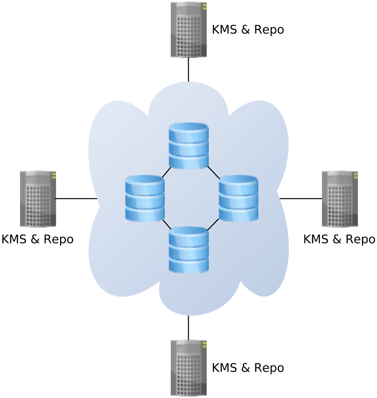
TÉCNICO LISBOA

SCALABILITY

○ Kurento Media Server and Kurento Repository in the same machine.

○ Fast data transfer on localhost.

○ Database bottleneck



FAST FORWARD

○ Implement fast forward on Kurento Media Server.

○ Convert videos in real time with a new playback velocity using *ffmpeg*.

○ Convert videos after recording with predefined playback velocities.

MEMORY USAGE

- MongoDB - Always receiving data, caches inserted data on RAM. Checkpoints data to disk every 60 seconds or when journal data exceeds 2GB.
- Play Framework - JVM uses memory recycling techniques instead of releasing to operating system.
- Kurento Repository - Caching videos.
- Kurento Media Server - Releases some memory and also recycles some of it.