EURECOM INSTITUTE SOPHIA ANTIPOLIS
MULTIMEDIA

Semester Project:

Grab your favorite TV show and share it

Project responsible
Assistant Prof. TRONCY Raphaël
Project supervisor
Ph.D. Student José Luis Redondo García
Post Doc Giuseppe Rizzo
Student:
PHAN Thành Trung

Content

Introduction

Context of this project

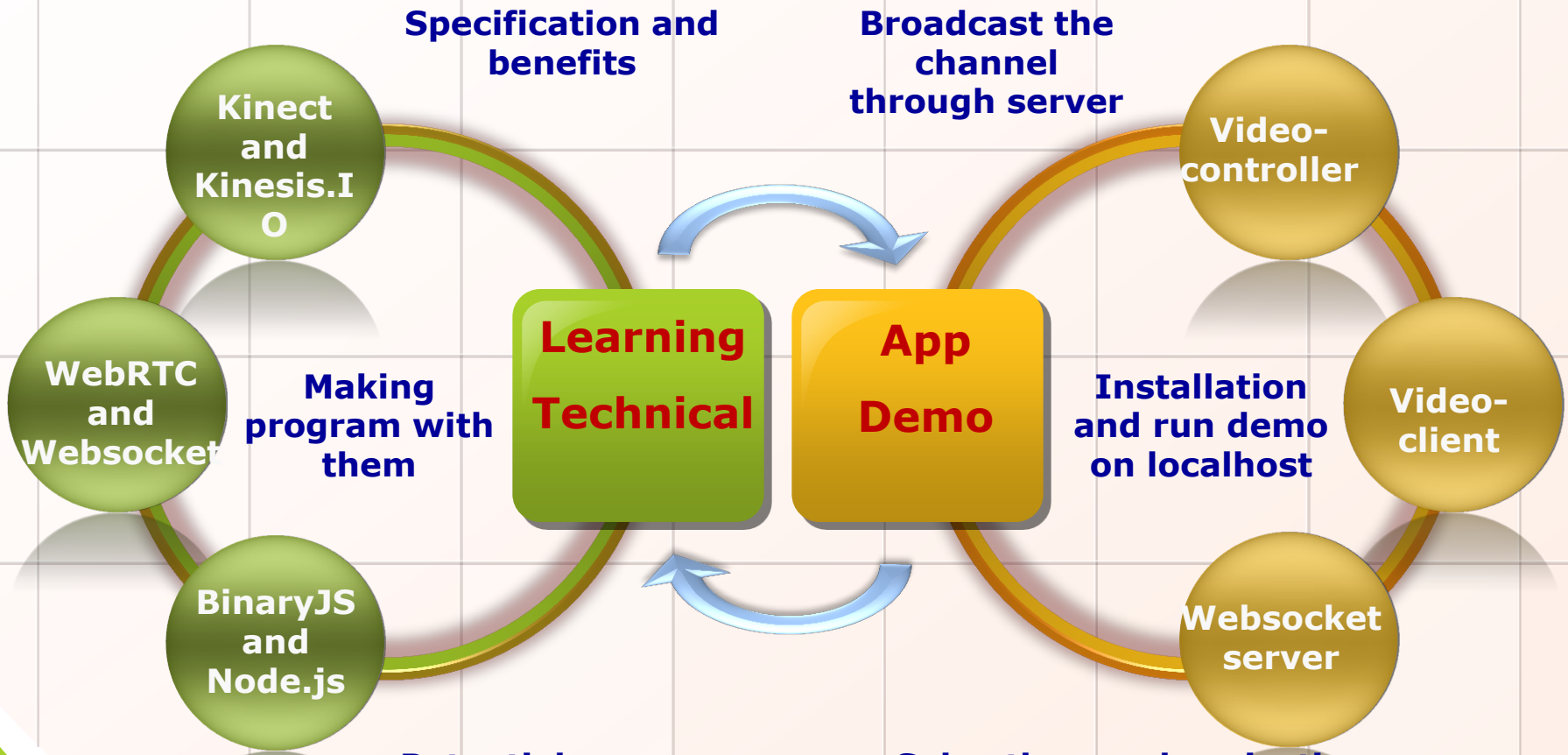
Introduction

Context of project

KINECT
for XBOX 360.

Introduction

Scope and objective

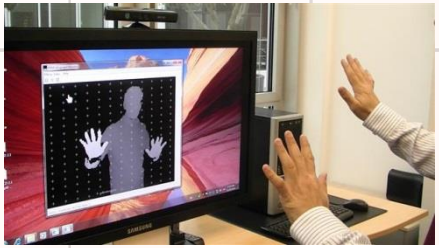


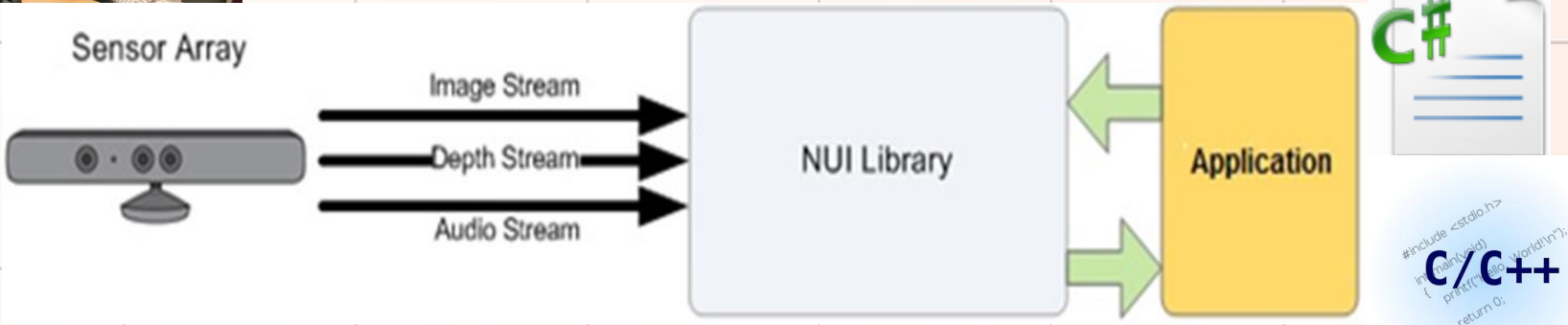
Content

Kinect Device and Kinesis.IO

What is Kinect Device?

"Motion sensing input device"





Kinect Device and Kinesis.IO

Developing with Kinect

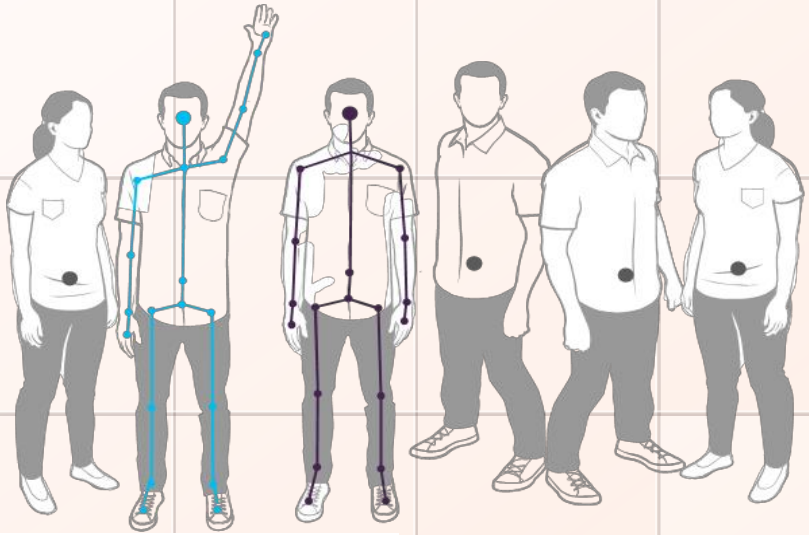
- Kinect for Windows SDK and Kinect Windows Developer Toolkit supports APIs and samples codes with:

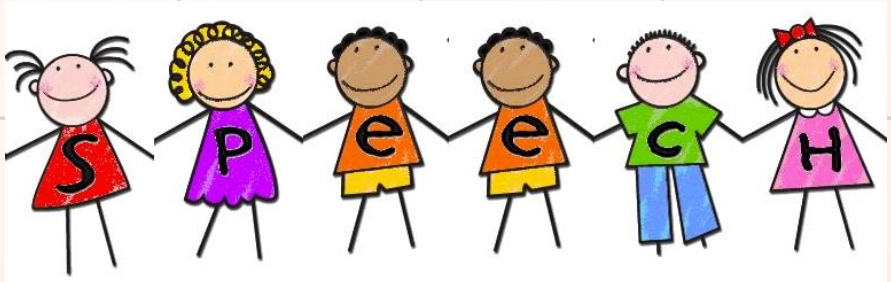
Audio Stream

Depth Stream

Color Stream

Infrared Stream

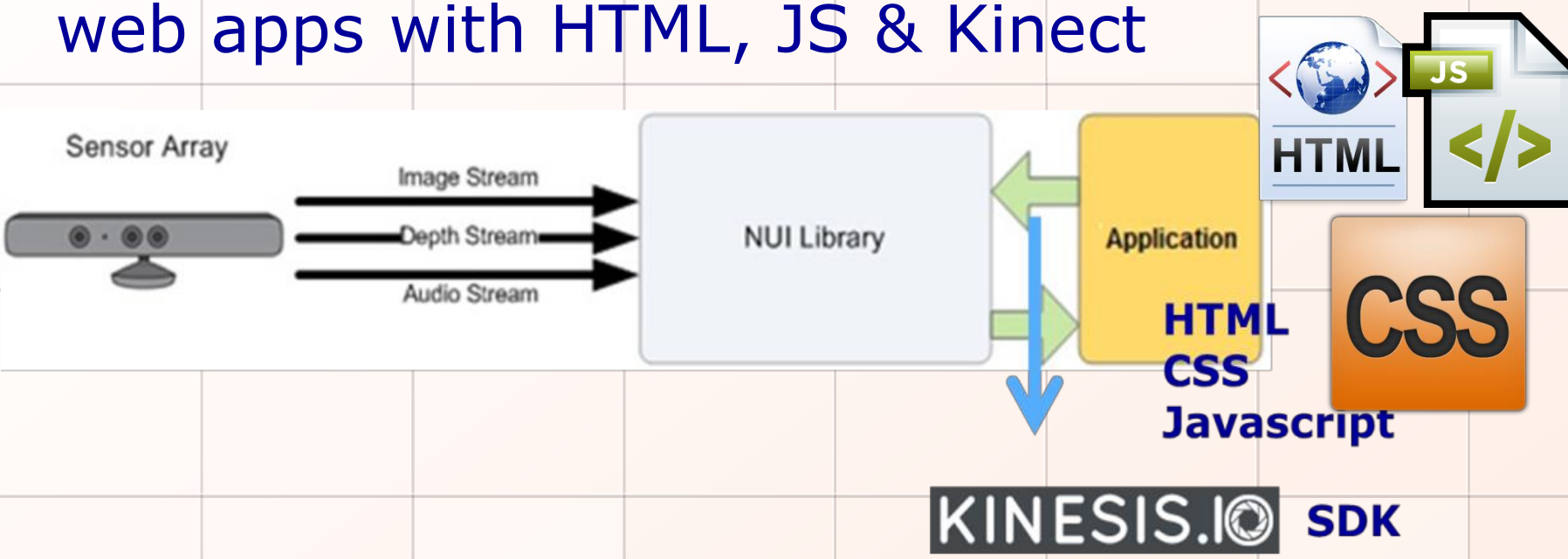




Kinect Device and Kinesis.IO

What is Kinesis.IO?

- Kinesis.IO helps us to build gesture driven web apps with HTML, JS & Kinect



- Kinesis.IO SDK includes Kinect SDK

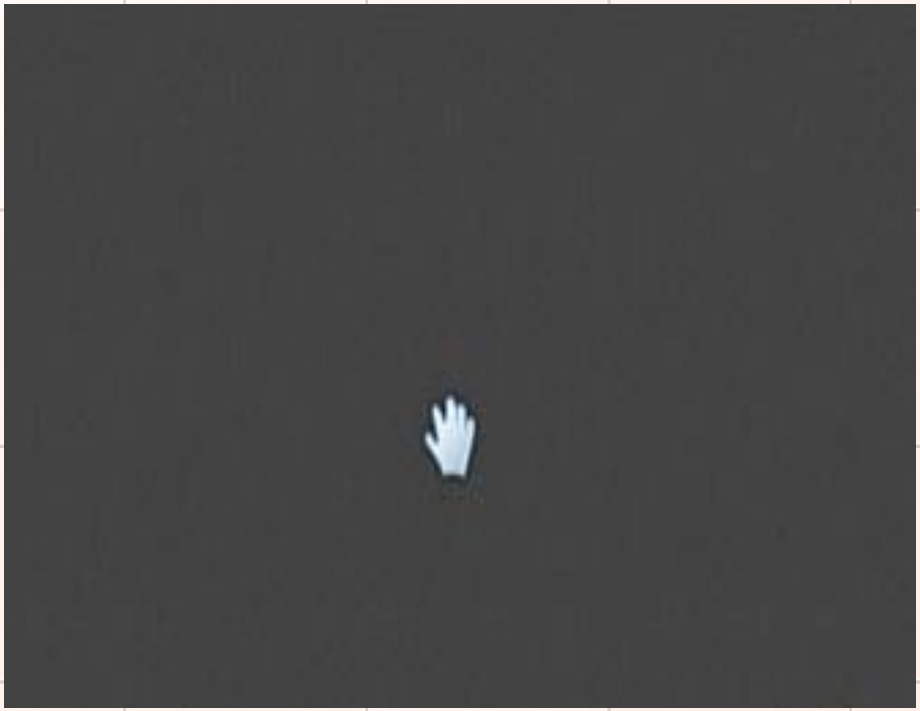
Kinect Device and Kinesis.IO

Developing with Kinesis.IO and Kinect device

- Hello world example code

Initialize! Few lines of code

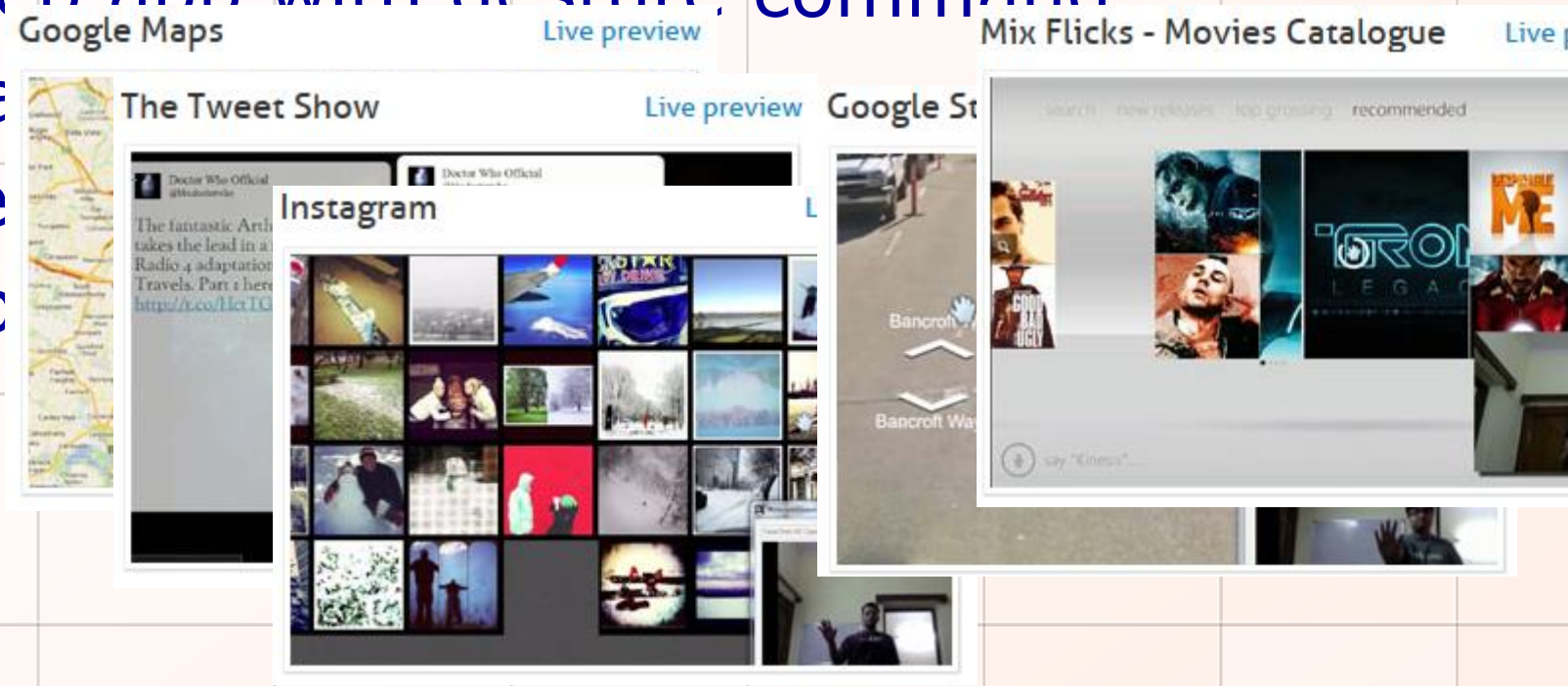
```
1  <head>
2    <!-- kinesis stylesheet -->
3    <link rel="stylesheet" type="text/css"
     href="/lib/css/kinesis.css">
4  </head>
5  <body>
6    <!-- kinesis js sdk -->
7    <script src="/lib/js/kinesis-js-sdk.min.js">
8    </script>
9    <script>
10     // initialize kinesis
11     var kinesis = new Kinesis;
12     // start adding gestures from here //
13   </script>
14 </body>
```



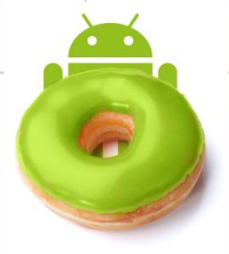
Kinect Device and Kinesis.IO

Potential with Kinesis domain

- Web app with gesture command
- Ga
- Te
- Sp



"Kinesis.IO is a solution for interacting between user and Kinect on laptop or smart TV through web browser"



Content

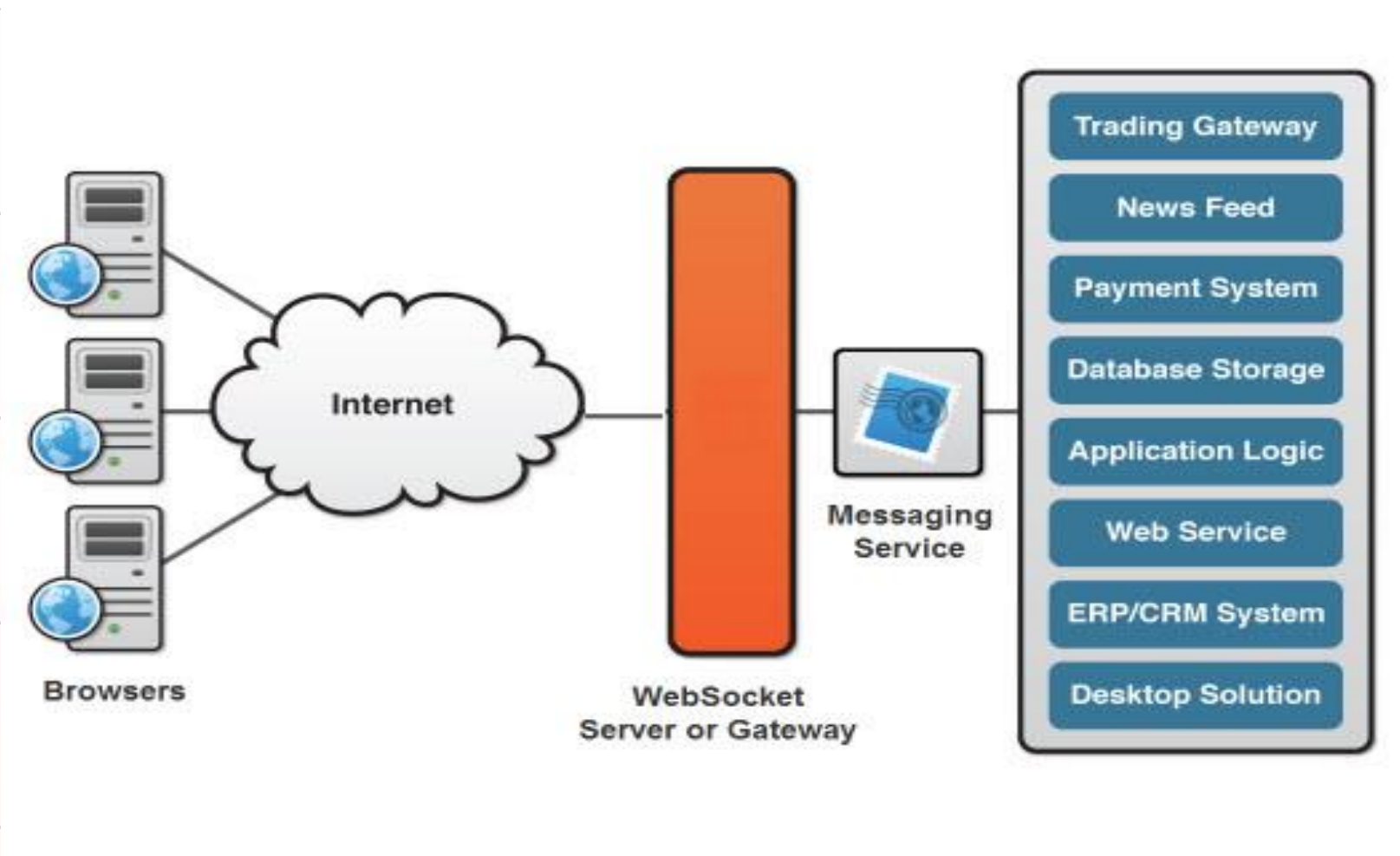
Websocket and Node.js

What is Websocket?

- JavaScript interface, defines a full-duplex single socket connection over which messages can be sent between client and server.
- Being used on the HTML5 environment.
- Reducing the complexity around bi-directional web communication and connection management

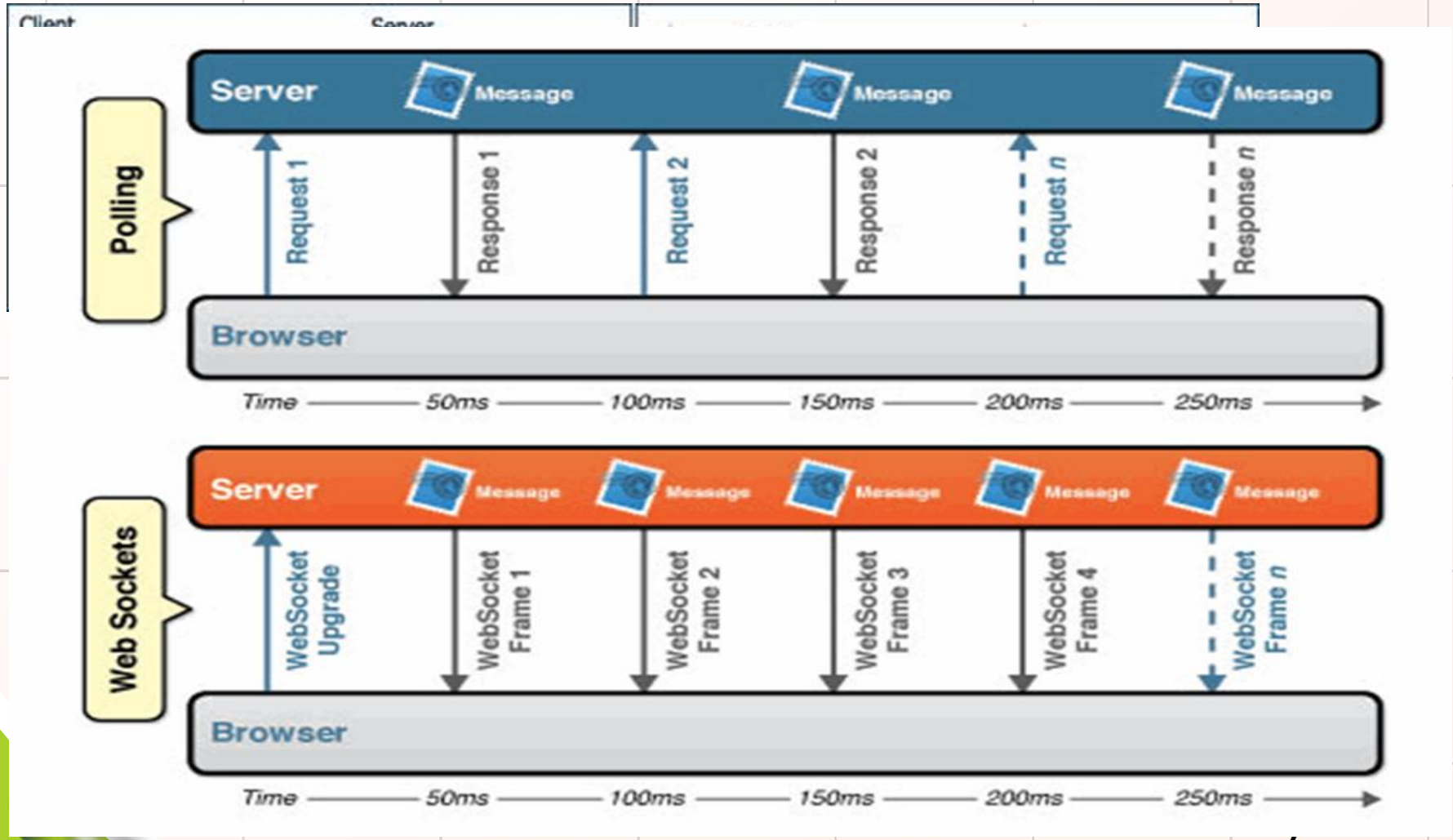
Websocket and Node.js

Websocket architecture



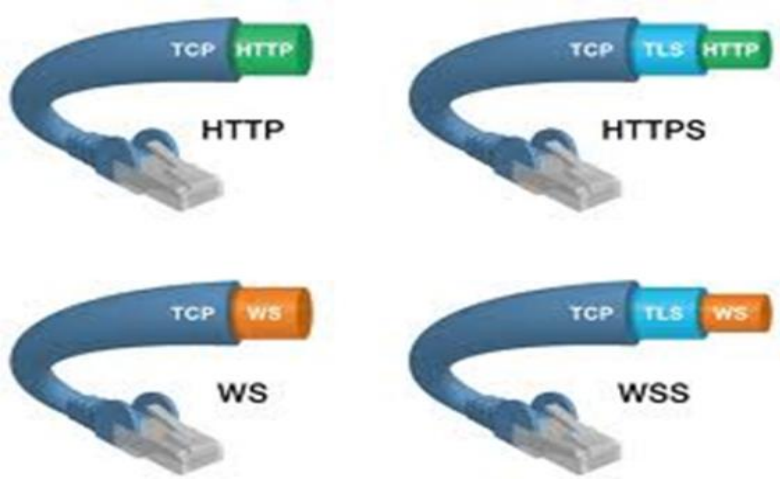
Websocket and Node.js

Comet VS Websocket



Websocket and Node.js

Websocket protocol and its potential



"It's good for the scope of this semester project → Update time stamp in real-time"

Websocket and Node.js

What is Node.js?

- The event server-side JavaScript which is good at handling lots of different kinds of I/O at the same time

Websocket and Node.js

Developing with Node.js

- Install Node.js package.
- Write the server side code.
- Start server command: Node servername.js

```
var http = require('http');
http.createServer(function (req, res) {
  res.writeHead(200, {'Content-Type': 'text/plain'});
  res.end('Hello World\n');
}).listen(1337, '127.0.0.1');
console.log('Server running at http://127.0.0.1:1337/');
```

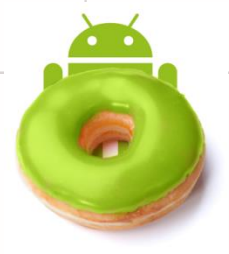
```
% node example.js
Server running at http://127.0.0.1:1337/
```

Websocket and Node.js

Potential of Node.js

- Building fast, scalable network applications.
- Using an event-driven, non-blocking I/O mode

“Lightweight, non-block IO → what we need for server in this project. Just transfer lightweight message ”



Content

1. Introduction
2. Kinect device and Kinesis.IO
3. WebSocket and Node.js
4. Application and demo
5. Conclusion

Application and demo

Application architecture

01

Server

- **Two servers: Websocket server (1) and HTML rendering source server (2).**
- Server 1: transfer message between controller and clients.
- **Server 2: shipping HTML source to browser**

02

Controller

- Getting command from user through Kinect.
- Send message to server (1).
- Get message from server (1)

03

Clients

- **Get messages from controller through server (1)**
- **Send message "canplay" to server (1) when it buffered enough video.**

All of them is running on the localhost

Application and demo

Application architecture

Keep hand stay on button
Swipe left
Open hand as a sign of pause

HTML Rendering Source Server

Send Hold gesture
Send swipeleft gesture
Send pause gesture

Send userCount
Send isAllReady

Send Timestamp
Send pause
Send canPlay
Send controlling video name

Send canPlay

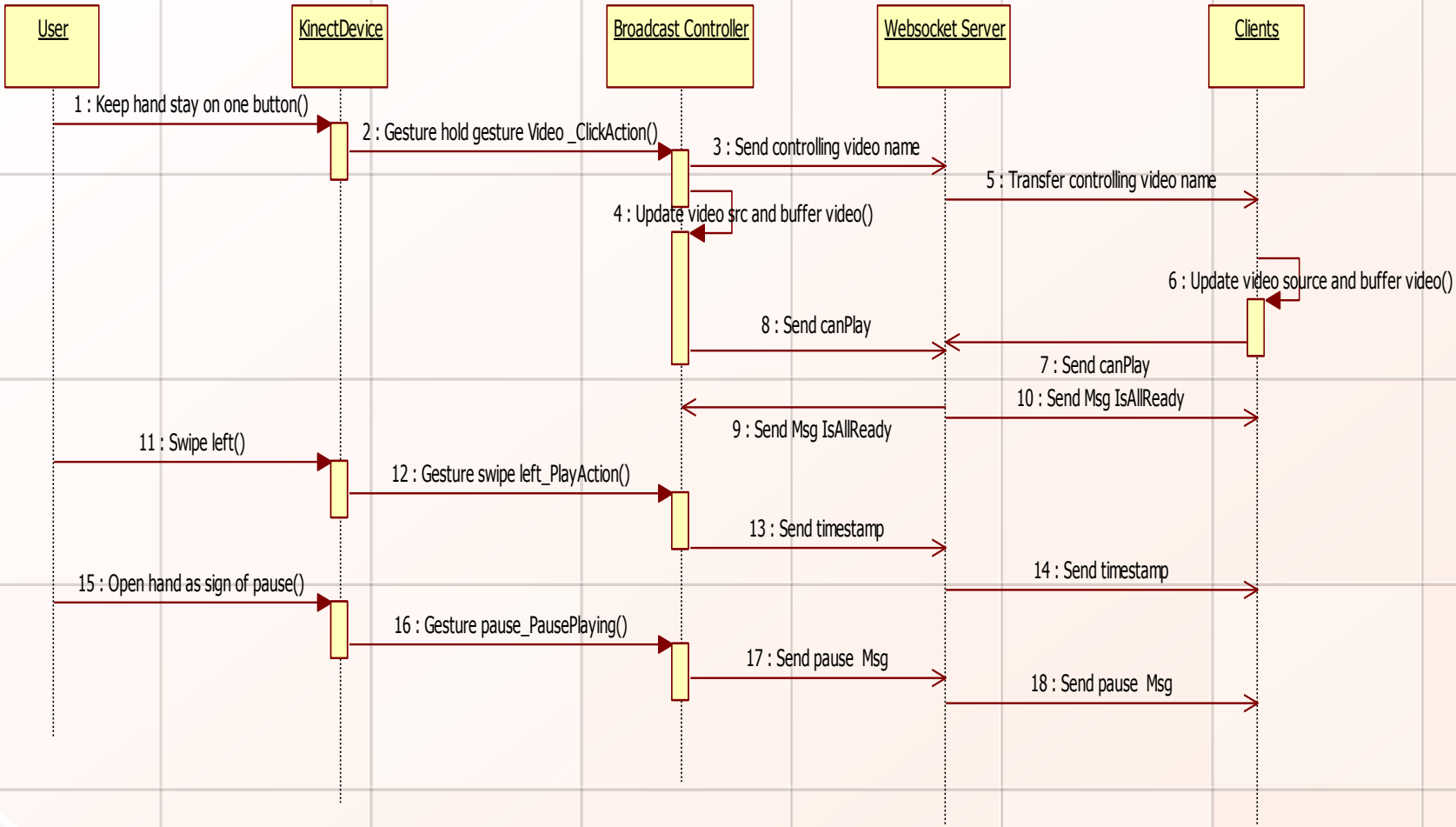
Send userCount
Send isAllReady
Send Timestamp
Send pause
Send controlling video name

Websocket Server

Process Msg

Application and demo

Sequence Diagram



Application and demo

Rendering HTML source server

- Playing as the source of keeping media source
- Rendering HTML source as well as media source for the Broadcast Controller and client(s).

Application and demo

Broadcast controller

- Sending the actions (play, pause, load video)
- Receiving number of users and the message of being ready all.

- **Time window**
- **Buffering the video**

Application and demo

Clients

- Receive messages from controller through websocket server
- Send the message of “canplay” to server to inform server that it buffers the video enough

- **Threshold to detect video sync or not**
- **Buffering the video enough**

Application and demo

Websocket server

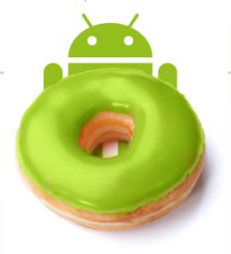
- Getting the message from controller.
- Processing it
- Transfering that message to all clients

- **How to detect all clients and controller are ready to broadcast or not**

Application and demo

Experiment and Demo

No.	Video	Latency between clients and controller (s)
1.	320x180 100kbit mp4	Time 1:0.018 - Time 2:0.002 - Time 3:0.007
2	640x360 800kbit mp4	Time 1:0.022 - Time 2:0.006 - Time 3:0.006
3	1280x720 1,4Mbit mp4	Time 1:0.553 - Time 2:0.056 - Time 3:0.138
4	1920x1080 2,7Mbit mp4	Time 1:2.367 - Time 2:1.942 - Time 3:1.932



Content

Conclusion

Obtained result

Code

Know how to develop JavaScript, HTML5, CSS on client side and server side.

Experience

Know the problem which is promoted in the process of playing video in synchronization.

Application

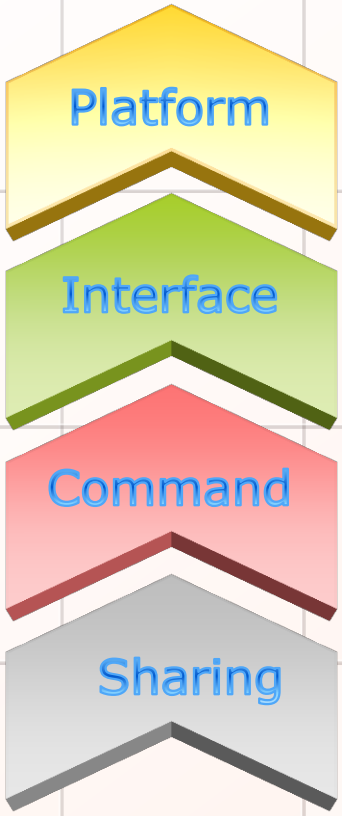
Building a small system of client, controller and server.

Trend

Realize the importance of second screen technology . It can be very popular in the future.

Conclusion

Development in future



- **Multi platform**
  Build controller or clients on other platforms: Linux, Mac
- **Customization**
  Change UI to adapt to each device: TV, tablet …
- **Speech**
  Add the voice command to command the controller.
- **Screenshot**
  Capture screenshots and share it on social network

Reference

- LinkedTV: http://www.linkedtv.eu/
- Microsoft Kinect: http://www.microsoft.com/en-us/kinectforwindows/
- Kinesis.IO : http://kinesis.io
- Node.js: http://nodejs.org/
- Websocket: http://www.websocket.org/
- Kinect on Wiki: http://en.wikipedia.org/wiki/Kinect
- Developing Kinect for Windows: http://www.microsoft.com/en-us/kinectforwindows/develop/new.aspx.
- Developer Download: . http://www.microsoft.com/en-us/kinectforwindows/develop/developer-downloads.aspx

Reference

- Kinect SDK Architecture: http://msdn.microsoft.com/en-us/library/jj131023.aspx
- MSDN- Skeletal Tracking: http://msdn.microsoft.com/en-us/library/hh973074.aspx
- HTML5 Web Sockets: *A Quantum Leap in Scalability for the Web* *http://www.websocket.org/quantum.html*
- BinaryJS: http://binaryjs.com/
- OpenKinect: http://openkinect.org/wiki/Main_Page

Thank you!