**Analysis report**

# Spike 01:

## Description:

The main purpose of Spike 01 is to measures the duration of the communication that occurs between the server side and the client when using socket.io and bonescript libraries in javascript.

For this spike, socket.io (client communication) and bonescript (server/sensors communication) were implemented through JavaScript. The bonescript library is used to probe the PIR motion sensor for a signal, then a timestamp is recorded before using socket.io library to send the time stamp to the client side. Upon the client reception of the message the client records a time stamp and then calculates the difference between the recorded time stamp and the received time stamp. Then, the client presents the motion sensor status received in the message and the calculated timestamp difference on the webpage and sends a copy of the timestamp difference back to the server. Finally, upon receiving the response on the server side, the server records a time stamp and calculates the difference between the current time and the timestamp recorded when the message is sent and prints out the result and a copy of the timestamp received from the client to the console.

We used: <https://www.npmjs.com/package/bonescript> and <https://www.npmjs.com/package/socket.io> for reference.

## Alternatives

Calculating the timestamp on the client requires both client and server to have a synchronized clock. In the case of beagle, synchronizing the time may introduce having an internet connection as a dependency. Hence, it is more accurate to calculate the duration on the server side as doing so will guarantee accuracy despite unsynchronized clock.

## Risk:

1. Inaccuracy of time stamp calculation.
   1. Calculate the timestamp difference on the server side
2. Client listeners (ex. browser) keep increasing by one each iteration
   1. Ensure that syntax and logic of the code is correct when dealing with socket communication to prevent the overlapping of sockets which causes the connection to increase with each iteration
3. Hardware malfunction (sensor or beaglebone)
   1. Ensure the pin layout scheme is followed and the pins are connected to the correct pins.

**NOTE:** Since the time stamp difference calculated on the client side is highly dependable on the time synchronization between the client and the server, the result has a high probability of inaccuracy. Hence, the time will be calculated on the server side by recording the duration of the two-ways communication (server-to-client-to-server) and dividing the duration by two to get the duration of one-way communication.

## Console output example.

**Date –** 12/4/017 till 15/4/017

**Time stamp:**

ping from client: -405

ping in Millisecs: 8.5ms

ping from client: -406

ping in Millisecs: 15ms

ping from client: -406

ping in Millisecs: 7.5ms

ping from client: -411

ping in Millisecs: 5.5ms

ping from client: -414

ping in Millisecs: 5ms

ping from client: -414

ping in Millisecs: 9.5ms

ping from client: -415

ping in Millisecs: 8ms

ping from client: -419

ping in Millisecs: 9.5ms

ping from client: -423

ping in Millisecs: 5.5ms

ping from client: -425

ping in Millisecs: 8.5ms

ping from client: -427

ping in Millisecs: 6.5ms

## Average Duration:

The average duration is calculated by summing 11 results (ping in Millisecs) and computing the average.

Average ping in Millisecs = 12.2 ms

\*\*NOTE: the client time stamp is used to show the inaccuracy of the client time stamp record, and all the time stamps are in milliseconds.

# Spike 02:

## Description:

The main purpose of Spike 02 is to measures the duration of a 1-way communication that occurs between the server side and the client when using firebase and beaglebone-io libraries in javascript.

For this spike, firebase library is used for client side communication and beaglebone-io is used for server/sensors communication, both libraries are implemented in Javascript. The beaglebone-io library is used to probe the PIR motion sensor for a signal and records the starting time stamp, then the firebase library is used to send the sensor data to the Firebase database. Once the database receives the data it sends back a response, upon the server reception the response, it records a time stamp and then calculates the difference between the recorded time stamp and the starting time stamp which it recorded before sending the data to Firebase. Then the duration is computed by dividing the result by two to get how long it takes to get the duration of a one-way communication to occur. Finally, the server prints the motion sensor state, the timestamp when the server sent the data to client and the duration of a 1-way communication to the console.

We used : <https://www.npmjs.com/package/firebase-tools> and <https://www.npmjs.com/package/beaglebone-io> for reference.

## **Alternatives**

Using firebase library for client side can introduce a stable internet connection as a dependency for the system to work. Furthermore, the duration of communication is highly depending on the connection quality. Hence, it is recommended to use socket-io library for more stability.

Using beaglebone-io library for server side can be challenging as there is no sufficient troubleshooting support and examples online. Hence, bonescript library is recommended as there are many resources online that assists problems troubleshooting.

Risks **:**

1. Hardware malfunction(Beaglebone or sensors)
   1. Ensure to follow beaglebone-io pin layout.
2. Client side does not receive data
   1. Ensure the internet connection is stable
   2. Ensure nothing is blocking the communication eg. Proxy, firewall

## Console output example:

**Date -**15/04/017 till 20/04/017

**Time Stamp:**

State: LOW timestamp: 1492843948498

Ping: 147.5ms

State: HIGH timestamp: 1492844307579

Ping: 161.5ms

State: LOW timestamp: 1492844310500

Ping: 150.5ms

State: HIGH timestamp: 1492844312656

Ping: 213.5ms

State: LOW timestamp: 1492844315068

Ping: 149.5ms

State: HIGH timestamp: 1492844341291

Ping: 147ms

State: LOW timestamp: 1492844343325

Ping: 147.5ms

State: HIGH timestamp: 1492844344488

Ping: 605.5ms

State: LOW timestamp: 1492844349417

Ping: 149.5ms

State: HIGH timestamp: 1492844353276

Ping: 146.5ms

State: LOW timestamp: 1492844356605

Ping: 157ms

## Average Ping:

The average 1-way communication duration is calculated by taking the sum of the result of 11 duration entries and dividing it by 11 to get the result in millisecond.

Average 1-way communication duration: 197.77ms

Conclusion**:**

The preferred spike that we both preferred was spike 01, as the risks related to spike 01 is less than spike 02. For speed, we preferred socket.io and bonescript because its fast and easy to troubleshoot. Moreover, it does not require internet connection to be present.