Discovering Hue lights locally, monitoring and controlling them remotely

Authors: Victor Aymard, Alexandre Avakian, Daniel Ikka, Adrien Lebret & Adrien Veau

INTRODUCTION

Cloud Computing provides an extensible and flexible infrastructure for storing and processing IoT application data. This makes it possible to process the data of Things centrally and makes it reasonable to access and control different things remotely as part of the IoT.

Also, the detection and localization of nearby devices play an important role in improving the quality of IoT application services, which is becoming increasingly popular in the field.

For this project we will use both cloud computing and location recognition to design a complete system that uses both technologies with a new specific IoT protocol, the MQTT to monitor and control IoT devices, the Philips Hue.

In this work we are going to talk first about, the **system architecture**, how does the system work and what is his components. Then about the **evaluation** of the project; did it work or not and how can we tell. And finally, we are going to **discuss** about what we have learn during this project.

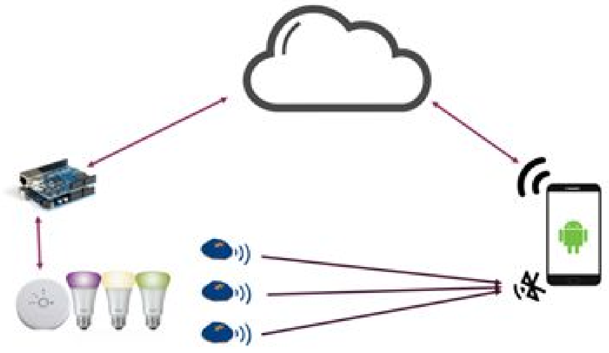
SYSTEM ARCHITECTURE

To allow the system to work, we must maintain and control smart objects in our environment. The smart objects are provided with a **Bluetooth beacon** that broadcast the existence of the object. Smart objects (the lamps) also send their status data by **MQTT** to a broker. After the object is find, we can control and monitor it via MQTT protocol.

In this project, we are controlling 3 Hue smart lights that we have already worked with previously. The beacons are broadcast to show existence of a lamp.

Then after being discovered by an Android application, we are able to receive the messages from the specific object discovered and also able to control it by the Android device.

The project overall communication model is illustrated below:



A specific lamp is discovered by a specific beacon. The Android application need to recognize three different lamps registered in a gateway by three beacons. The broker service receives attributes of lamps, the hue, the brightness and the status on/off every 20 seconds.

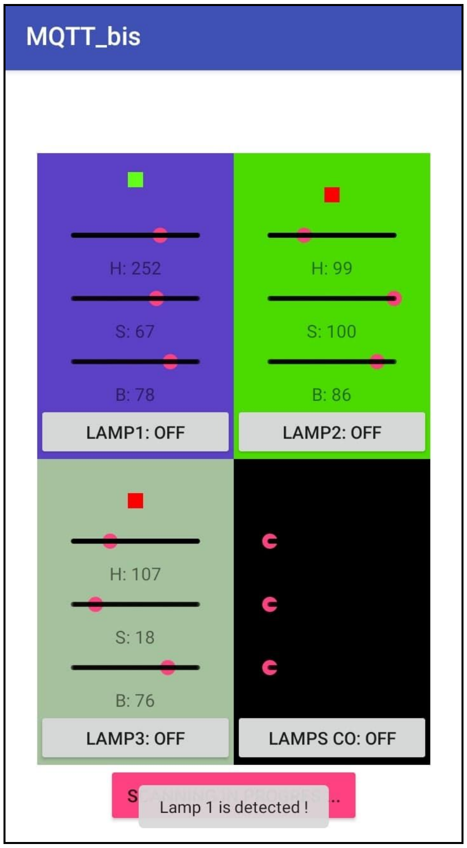
The Android device allow to the three different lamps to monitor their current status and attributes, to configure their attribute (hue, brightness and on/off) one by one, to set a configuration for all three lamps as a group configuration, to Keep and show their last published attributes.

The broker also preserves the last received messages from the lights. If no one is subscribed for receiving a message in a specific topic, the last published message is kept in the broker. When a mobile discovers a lamp, it receives its last status immediately, and it does not wait for the next publishing data from the lamp.

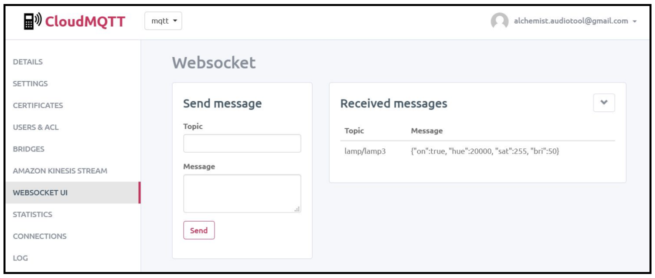
When Android phone lost receiving messages from a specific beacon for more than 10 seconds, then it stops receiving attribute messages from that lamp.

EVALUATION

For our application we have structured it like this:



We have 4 control menu with the hue, the saturation and the brightness. Three menu for the lamp 1, 2 and 3 and 1 menu to control all three lamps at the same time.



Topic	Message
lamp/lamp3	{"on":true, "hue":20000, "sat":255, "bri":50}

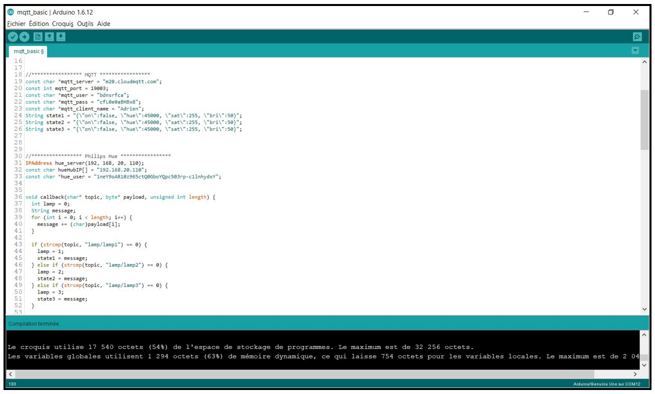
For the transmission between the application and the arduino, we used the MQTT transmission. For the transmission of information between the Android part and the MQTT we made 3 topics lamp/lamp1-2 and 3 for the sending **app** → **Arduino** and 1 Channel lamp/info from **Arduino** → **app**.

In addition, the message is sent when you remove the finger from a slider and click on a button (application)

It is important to notice that if the beacon of a lamp is not detected we do not send any message.

In addition, the menu that manages the 3 lamps at the same time, the latter sends three different messages: one on each Channel.

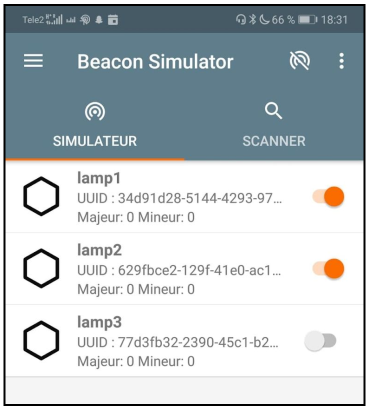
Here is an example of the arduino code:


mqtt_basic | Arduino 1.6.12
Fichier Édition Croquis Outils Aide
mqtt_basic §
Compilation terminée.
133
Arduino/Genuino Uno sur COM12

```


//***************** MQTT *****************
const char *mqtt_server = "m20.cloudmqtt.com";
const int mqtt_port = 19003;
const char *mqtt_user = "bdnsrfca";
const char *mqtt_pass = "cfL0e0aBHBx8";
const char *mqtt_client_name = "Adrien";
String state1 = "{\"on\":false, \"hue\":45000, \"sat\":255, \"bri\":50}";
String state2 = "{\"on\":false, \"hue\":45000, \"sat\":255, \"bri\":50}";
String state3 = "{\"on\":false, \"hue\":45000, \"sat\":255, \"bri\":50}";



//***************** Philips Hue *****************
IPAddress hue_server(192, 168, 20, 110);
const char hueHubIP[] = "192.168.20.110";
const char *hue_user = "1neY9oAR10z965ctQ0GboYQpc503rp-c1lnhydxY";


void callback(char* topic, byte* payload, unsigned int length) {
  int lamp = 0;
  String message;
  for (int i = 0; i < length; i++) {
    message += (char)payload[i];
  }

  if (strcmp(topic, "lamp/lamp1") == 0) {
    lamp = 1;
    state1 = message;
  } else if (strcmp(topic, "lamp/lamp2") == 0) {
    lamp = 2;
    state2 = message;
  } else if (strcmp(topic, "lamp/lamp3") == 0) {
    lamp = 3;
    state3 = message;
  }
```

```
Le croquis utilise 17 540 octets (54%) de l'espace de stockage de programmes. Le maximum est de 32 256 octets.
Les variables globales utilisent 1 294 octets (63%) de mémoire dynamique, ce qui laisse 754 octets pour les variables locales. Le maximum est de 2 04
```

For the Beacon, we use a Beacon emulator on a phone.



lamp1
UUID : 34d91d28-5144-4293-97...
Majeur: 0 Mineur: 0

lamp2
UUID : 629fbce2-129f-41e0-ac1...
Majeur: 0 Mineur: 0

lamp3
UUID : 77d3fb32-2390-45c1-b2...
Majeur: 0 Mineur: 0

On the **evaluation day**, the whole project was working as we would like it to. However, since we had not used a Beacon emulator, but only the Bluetooth of our devices with which we associate a lamp, we had to review this part. The approach remained the same for Bluetooth and Beacon, and we just had to set up the library in our Android code. We have achieved the results described above.

DISCUSSION

It was a great experience we gained during the whole project. We got a chance to build a way to maintain and control smart objects in your environment. For this project the smart object was Hue smart light.

To complete this project, we had to use all the knowledge acquired during the course in Internet of Things like in the assignment 2: “Controlling Hue light”, where we had to control the Hue lights via Gateway webpage then setup a connection between Arduino and Hue gateway and finally control the hue lamp via Arduino. And the knowledge acquired during the assignment 3 by developing our own Android MQTT client using CloudMQTT and turning a led On/Off in Arduino by MQTT.

During this project, we had to face multiples challenges like the one that we met during the evaluation day. It is together and concentrated that we have succeeded in meeting this challenge.

Indeed, we can also remember **teamwork** at the end of this IOT project. Team spirit is a very important skill that will serve us in the future, in our engineer job.
For this project, we very quickly, and naturally, divided into 3 groups that correspond to the 3 parts of this project: Arduino / MQTT / Android. Each group was able to carry out its above-mentioned task. Communication within the team was also very important, to know what everyone was sending and should receive (inputs / outputs).

CONCLUSION

Looking at the specifications and the result of our project we can say that we have met the expectations by managing to use all the tools we learned during our different laboratories.

We managed to control the hue light via arduino and via MQTT, we managed to transmit the information of the Android application to the arduino.

But we still encountered many challenges, mainly the challenge of the Beacon. But thanks to a good team spirit and a lot of determination we were able to finish the project completely. This one was very enriching by the subject he proposed.

The design of the application through android gave a really interesting aspect to this project with the possibility to personnalise our own user interface. Coming from a school where electronics plays an important role, the Arduino language was already known by some of the members of the group but the MQTT transmission and the Beacon was new to everyone.

At the end, we have gained better knowledge about the languages and protocols at the heart of cloud computing and location knowledge.