

L3 Miage

**Rapport de Stage**

**PROGRAMMATION DES ENVIRONNEMENTS**

**DITES INTELLIGENTE**

Aly FALL

LABORATOIRE TIMC/IMAG GMCAO

Tuteur Entreprise : FOUARD Céline(TIMC) et Alexandre Demeure(IIHM)

Tuteur Université : PIERRE Laurence

**Résumé**

Né au milieu des années 1980, le développement de la domotique est, entre autre, la conséquence de la miniaturisation des systèmes électroniques et informatiques.

Le stage réalisé au **Laboratoire TIMC/IMAG** a justement pour but le développement de services et d’appareils dans le cadre de l’internet des objets et de la maison intelligente. Ces appareils services devront être intégrés à une plateforme de domotique qui est openhab. La conception des appareils est basée sur la plateforme ESP8266, le développement est réalisé via l’IDE arduino. L’objectif est d’arriver à piloter des petits moteurs (moteur DC 12V) à distance à travers une interface humain-machine.

**Mots-clés**

Domotique, ESP8266, internet des objets, maison intelligente, arduino

**Abstract**

Born in the mid-1980s, the development of home automation is, among other things, the consequence of the miniaturization of electronic and computer systems.

The internship at the TIMC / IMAG Laboratory aims to develop services and devices in the context of the Internet of Things and the smart home. These devices services will have to be integrated into a home automation platform that is openhab. The device design is based on the ESP8266 platform, the development is done via the arduino IDE. The goal is to manage small motors (DC 12V motor) remotely through a human-machine interface.

**Keywords**

Home automation, ESP8266, internet of things, smart home, arduino.

# Contexte du stage

* 1. **Présentation de l’entreprise**
     1. Caractéristiques

Mon stage devait se dérouler au laboratoire informatique de Grenoble (LIG) mais par défaut de place je me suis installé au laboratoire TIMC/IMAG situé au rond point de la chantourne Pavillon Taillefer 38700 la tronche France. Le laboratoire informatique de Grenoble (LIG) a était créé le 1er janvier 2007, c’est un laboratoire en recherche français situé au 700 avenue Centrale dans le Domaine Universitaire à Saint-Martin d’Hères.

* + 1. DESCRIPTION DES ACTIVITÉS

Le laboratoire d’informatique de Grenoble est composé de 24 équipes qui représentent chacune un domaine de recherche spécifique.

L’ambition est de s’appuyer sur la complémentarité et la qualité reconnue des 24 équipes de recherche du LIG pour contribuer au développement des aspects fondamentaux de l'informatique (modèles, langages, méthodes, algorithmes) et pour développer une synergie entre les défis conceptuels, technologiques et sociétaux associés à cette discipline. Ici je me focalise sur **l’axe Systèmes Interactifs et Cognitifs**

Constitué des équipes IIHM dont je suis rattaché, MAGMA, METAH et PERVASIVE Interaction. Elles ont comme préoccupation commune de se focaliser sur l'utilisateur au cœur de son environnement physique et numérique, considéré de sa conception jusqu’à sa coadaptation dynamique à l'exécution.

* 1. **Présentation du service d’accueil**

Mon stage s’étant déroulé la majorité du temps au laboratoire TIMC je présente donc aussi ce laboratoire.

TIMC-IMAG compte environ 300 membres répartis sur 12 équipes de recherche, dont environ 160 personnels permanents : chercheurs, enseignants chercheurs, ingénieurs, techniciens et personnels administratifs. Il accueille chaque année près de 70 doctorants et post-doctorants.

J’ai été affecté à l’équipe GMCAO du laboratoire. Créée par le Pr Philippe Cinquin il y a plus de 3 décennies, l'équipe GMCAO a été pionnière dans le développement de dispositifs d'assistance aux gestes médico-chirurgicaux. J’ai donc intégré l’équipe en tant que membre non permanent avec le statut de stagiaire, travaillant avec Monsieur Alexandre DEMEURE - Maître de conférences (Université Grenoble Alpes) qui est mon tuteur de stage.

* 1. **Problématique du projet**

Dans le cadre de l’internet des objets, l’équipe IIHM cherche à concevoir des plateformes des objets communicants qui puissent facilement s’intégrer aux plateformes existantes comme openHab [[1]](#footnote-1)qui est utilisé entre autre dans la plateforme Amiqual4Home[[2]](#footnote-2). Mon encadrant m’a demandé d’explorer l’utilisation de la plateforme ESP8266 qui est très usitée par les concepteurs de solutions Do It Yourself (DIY). L’objectif est d’appréhender la faciliter de développement avec cette plateforme, notamment en ce qui concerne son utilisation pour piloter de petit moteurs électriques.

Cette plateforme pourra ensuite être utilisée pour concevoir des appareils intégrables dans un habitat dit intelligent.

1. **DEROULEMENT DU PROJET**

Durant les toutes premières heures, nous avons pris connaissance des modalités de déroulement du stage tel que le respect de la hiérarchie, la ponctualité, l’assiduité et toutes autres conditions relatives au fonctionnement du laboratoire. Ensuite le projet s’est déroulé selon le planning suivant :

L’étape après avoir pris connaissances de toutes les modalités du stage consistait à la prise en main en main de environnement de travail du coup

je me suis rendu au FabMSTIC qui est un fablab universitaire pour faire mes premiers pas dans mon projet vu qu’ils ont déjà travaillé avec la plateforme ESP8266.

Ensuite, c’était le moment de me documenter sur l’ESP8266. Celui-ci est une plate-forme open source IoT (Internet of Things), matérielle et logicielle, basée sur un SoC Wi-Fi ESP8266 ESP-12 fabriqué par fabricant chinois  Espressif Systems.

L’ESP8266 peut se programmer de plusieurs façons soit avec des scripts [**Lua**](https://fr.wikipedia.org/wiki/Lua) interprétés ou compilés, du [**JavaScript**](https://fr.wikipedia.org/wiki/JavaScript)**,** [**MicroPython**](https://fr.wikipedia.org/wiki/MicroPython)**, du C, En** Go, ou bien en C++ avec l’IDE arduino et c’est cette dernière façon de programmer que j’ai opté pour vu sa faciliter à programmer.

Le mécanisme pour programmer sous cette plateforme consiste à coder via l’IDE arduino et après faire le televersement du code vers L’ESP8266 à travers un câble et celui-ci exécute le code écris en C++.

Pendant cette période au fablab j’ai mis en place la première version de mon serveur web et la mise en place de la connexion via un réseau WIFI.

La prochaine étape consistait à mettre en place le circuit électrique celui-ci est constitué de :

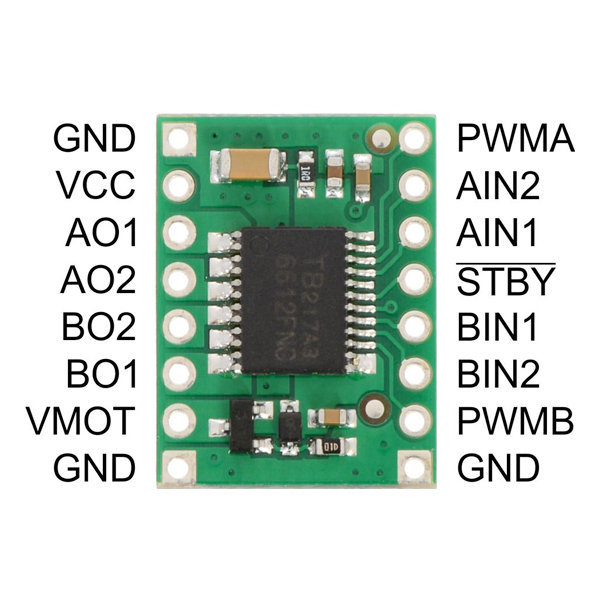
* **Moteur DC 12V** (voir figure 1) qui est un petit moteur a courant continue.



**Figure 1** : moteur DC 12V

La figure ci-dessus nous montres un moteur DC le moteur à courant continu est un moteur électrique 12v qui s'utilise dans de nombreux domaines. D'un fonctionnement simple, le moteur 12v DC offre une pluralité de possibilités. Transformant une énergie électrique en énergie mécanique, on le retrouve dans de multiples installations, bricolages, jouets, etc.

Le circuit électrique est aussi constitué par une carte de pilotage. La carte de pilotage utiliser dans le projet est le pololu (voir figure 2) qui permet de piloter jusqu’à deux moteurs de ce genre.



**Figure 2** : carte pololu

La figure ci-dessus montre une carte de pilotage pololu permettant de piloter des moteurs. Elle possède plusieurs broches de sortie (16 bronches), chaque broche de sortie a un rôle bien défini et doit être connecté à la bonne bronche du NodeMCU à l’aide de câble de liaison.

Et après la mise en œuvre du circuit électrique l’étape suivante consiste a développé le programme et celui-ci s’est déroulé  suivant trois étapes :

Premièrement, la mise en place d’un serveur WIFI. Cette étape consistait à crée un programme (serveur HTTP) qui va se connecter à un réseau WIFI existant. Pour cela il faut importer la bibliothèque **WiFiClient.h** qui va nous permettre d’utiliser la fonction **WiFi.begin**(nom du wifi, password) qui prend en paramètre le nom du réseau WIFI et le mot de passe pour se connecter à se réseau. Une fois connecter à se réseau tout utilisateur connecter à se réseau peut prendre le contrôle du moteur et le piloter.

Ensuite il fallait étudier le comportement du moteur. Après la mise en place de la connexion WIFI il était temps de mettre en place un programme qui permet à la carte pilotage de commander le moteur indépendamment du service WIFI. Ce programme nous a permis d’observer le comportement du moteur DC pour mieux pouvoir le programmer.

Et à la fin, après avoir terminées les deux précèdent étapes et s’être assurer que tout marche indépendamment, cette phase consistait à mettre en place le programme principal pour que le moteur réponde aux instructions qui viennent de l’utilisateur sous forme de requêtes http via une page web.

1. **PRINCIPALES RÉALISATIONS**

Il a était primordiale de comprendre les concepts fondamentales qui tournent autour de ce projet avant de se lancer dans le cœur du travail. Le projet possède deux axes :

1. **Pilotage du moteur**

La programmation de la carte de pilotage moteur nécessitait la lecture de la documentation de cette carte car chaque carte de pilotage moteur à sa façon d’être codé.

Un moteur pas à pas (moteur DC 12V) est un type de dispositif électromagnétique dont la rotation peut être contrôlée de façon précise. Ce type de moteur possède deux sens de rotation qui sont gérés par les Les broches d’entrée AIN1 et AIN2 de la carte de pilotage du moteur A spécifiant le sens de rotation du moteur. Il suffit de mettre l’une des bronches (AIN1 ou AIN2) sur HIGHT et l’autre sur LOW pour avoir un sens et d’inverser pour avoir l’autre sens de rotation. Pour la gestion de la vitesse il suffit de jouer sur la bronche PWD et lui mettre une vitesse entre 0-1200.

En se qui concerne l’alimentation du moteur il suffit de jouer sur la bronche STBY c’est-à-dire la mettre sur HIGHT pour l’allumer et sur LOW pour éteindre celui-ci.

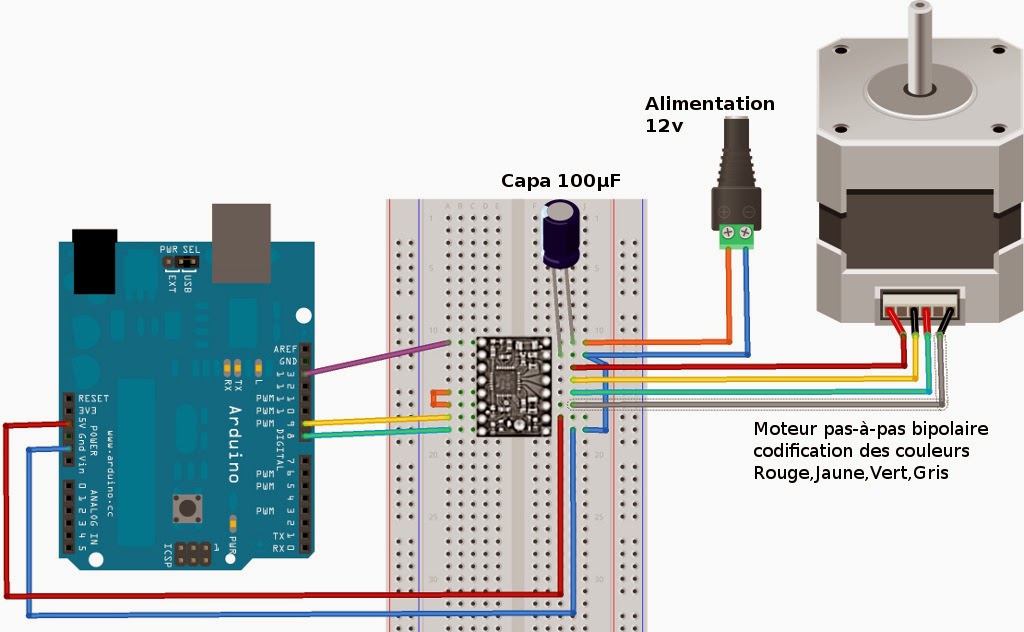
1. **Interface HTTP**

L’interface  HTTP fournit une fonctionnalité permettant d’émettre des requêtes HTTP correctement structurées et de traiter leurs réponses.

Dans notre projet le serveur Web s’attend à six types de requêtes qui représentent chacune une fonction au quelle le serveur devra faire appelle comme réponse a cette requête http qui sont :

A noter ici que le mot IP fait allusion à l’adresse IP du serveur Web de l’ESP8266.

* **Allumer** : permet à la personne d’allumer le moteur à distance. Cette ressource est accessible via URI IP /startUp
* **Eteindre** : permet à l’utilisateur d’éteindre le moteur à distance. Cette ressource est accessible via URI IP/shutDown
* **Avancer** : permet à l’utilisateur de faire marcher le moteur vers de l’avant. Cette ressource est accessible via URI IP/goForward
* **Reculer** : permet à l’utilisateur de faire marche arrière du moteur. Cette ressource est accessible via URI IP/goBackward
* **Accélérer**: permet à l’utilisateur d’augmenter la vitesse du moteur. Cette ressource est accessible via URI IP/accelerer
* **Décéléré** : permet au moteur de diminuer la vitesse du moteur. Cette ressource est accessible via URI IP/ralentir



**Figure 3** : circuit complet

La figure 3 illustre le circuit complet du projet. Elle contient tous les éléments qui rentrent en jeux dans le projet pour le bon fonctionnement de la plateforme ESP8266.

1. **Conclusions**

4.1. **Intérêt du travail pour l’entreprise**

A la fin de ce stage comme je l’ai annoncé tout au début, le laboratoire pourra bénéficier d’un programme conçu pour être intégré dans des environnements intelligents qui sont des environnements dans lesquels des détecteurs et actionneurs ont été intégrés, pour réagir à des évènements et s'adapter aux personnes présentes.

Le laboratoire disposera d’un serveur programmé pour recevoir des requêtes http et répondre à celles-ci. Ce NodeMCU va aussi va servir de microcontrôleur pour le pilotage du moteur DC 12V.

* 1. **Intérêt du travail pour le stagiaire**

Ce stage m’a permis de découvrir un autre domaine de l’informatique qui est celui lié à la domotique. Au début de ce stage j’avais aucune idée comment et même pas le minimum de bagage pour abordé ce stage et comment pourrais-je arriver à réaliser ce projet ?

Ce qui fut un chalenge pour moi.

Durant ce stage j’ai développé mes capacités de développement en C++ car c’est le langage de programmation que j’ai utilisé pour le développement du serveur du NodeMCU.

Et en plus j’ai exploré l’utilisation de la plateforme ESP8266 qui est une plate-forme open source et de son environnement de développement arduino, celui-ci peut être utilisé pour construire des objets interactifs indépendants (prototypage rapide), ou bien peut être connecté à un ordinateur pour communiquer avec ses logiciels.

* 1. **Axes d’améliorations**

Après avoir fini le développement et la mise en place de tous les services la prochaine étape consistera à l’intégration de l’ESP8266 dans la plateforme openHab.

* 1. **Conclusion libre**

Ce stage a été très enrichissant car il m’a permis d’appréhender la faciliter de développement avec la plateforme ESP8266, notamment en ce qui concerne son utilisation pour piloter de petit moteurs électriques.

En plus des compétences techniques j’ai développé des compétences personnel car évoluait au sein d’une entreprise implique beaucoup de discipline et de respect de l’autrui.

Et aussi sa m’a permis de développé mon sens du relationnel à travers le contact avec les stagiaires présent et les membres du laboratoire ceux qui a était[[3]](#endnote-1) une expérience vraiment enrichissante.

1. **Références bibliographiques**
2. **BIBLIOGRAPHIE**

pololu tb6612fng dual motor driver with an Arduino to control two low powered DC motors consulté le 08 juillet

<https://www.robotshop.com/community/forum/t/using-the-pololu-tb6612fng-dual-motor-driver-with-an-arduino-to-control-two-low-powered-dc-motors/13158>

Mise en place serveur Web consulté le 04 juillet

[https://randomnerdtutorials.com/esp8266-web-server/](https://randomnerdtutorials.com/esp8266-web-server/" \t "_blank)

1. **WEBOGRAPHIE**

tutoriel arduino consulté le 05 juin 2019

<https://www.youtube.com/watch?v=9f9KYO8Hzzg&list=PL5sYQxMUpgkO_FGrsMmfrqhBuZphBJ6uD>

Site web de openhab consulté le 16 juin 2019

<https://www.openhab.org/>

site web plateforme Amiqual4Home consulté 23 juin 2019

<https://amiqual4home.inria.fr/fr/>

1. **Remerciements**

Je remercie mon tuteur et responsable Monsieur

DEMEURE Alexandre pour l’aide et les conseils concernant les missions évoquées dans ce rapport et pour m’avoir proposé ce sujet de

stage. Ce fut un honneur.

Remerciements à l’ensemble du personnel de la TIMC pour leur accueil chaleureux et leur coopération professionnelle tout au long de ce stage.

Annexes

[Annexe 1 : Code 10](#_Toc12969096)

[Annexe 2 : code Page web 12](#_Toc12969097)

Annexe 1 : Code

#include <ESP8266WiFi.h>

#include <WiFiClient.h>

#include <ESP8266WebServer.h>

#include "index.h"

#include <SPI.h>

#include <DNSServer.h>

// Include the Wi-Fi-Multi library

#include <ESP8266WiFiMulti.h>

// Include the mDNS library

#include <ESP8266mDNS.h>

/\* cablage des broches de la carte de puissance TB6612FNG vers la carte arduino \*/

#define PWMA D7

#define AIN1 D1

#define AIN2 D0

#define STBY D3

#ifndef APSSID

#define APSSID "iPhone de Aly"

#define APPSK "af123456789"

#endif

MDNSResponder mdns;

/\* Set these to your desired credentials. \*/

const char \*ssid = APSSID;

const char \*password = APPSK;

const byte DNS\_PORT = 53;

ESP8266WebServer server(80);

void setup()

{

// definition des broches de la carte de puissance en tant que sortie

pinMode(PWMA, OUTPUT);

pinMode(AIN1, OUTPUT);

pinMode(AIN2, OUTPUT);

pinMode(STBY, OUTPUT);

// Ouvre la voie série avec l'ordinateur

Serial.begin(115200);

// Connect to Wi-Fi network with SSID and //password

Serial.print("Connecting to ");

Serial.println(ssid);

WiFi.begin(ssid, password);

// attente de connexion

while (WiFi.status() != WL\_CONNECTED) {

delay(500);

Serial.print(".");

}

// on affiche l'adresse IP qui nous a ete attribuee

Serial.println("");

Serial.println("WiFi connected.");

Serial.println("IP address: ");

Serial.println(WiFi.localIP());

// DNS

if(!MDNS.begin("esp8266")){

// Start the mDNS responder for esp8266.local

Serial.println("Error setting up MDNS responder!");

}

Serial.println("mDNS responder started");

server.on("/", handleRoot);

server.on("/startUp", startUp);

server.on("/shutDown", shutDown);

server.on("/goForward", goForward);

server.on("/goBackward", goBackward);

server.on("/accelerer", accelerer);

server.on("/ralentir", ralentir);

server.begin();

// Add service to MDNS-SD

MDNS.addService("http", "tcp", 80);

}

void loop()

{

// dnsServer.processNextRequest();

server.handleClient();

}

void handleRoot() {

server.send(200, "text/html", MAIN\_page);

}

void startUp ()

{

if (digitalRead(STBY) == LOW)

{

server.send(200, "text/html", "<h1> ON</h1>");

Serial.println("Debut");

digitalWrite(STBY, HIGH);

Serial.println("STBY HIGH");

delay(1000);

digitalWrite (AIN1, LOW);

digitalWrite (AIN2, LOW);

//ce code est juste pour un test

} else{

digitalWrite(STBY, HIGH);

Serial.println("STBY HIGH");

delay(1000);

}

}

void shutDown ()

{

server.send(200, "text/html", "<h1>off</h1>");

digitalWrite(STBY, LOW);

Serial.println("STBY LOW");

digitalWrite (AIN1, LOW);

digitalWrite (AIN2, LOW);

delay(1000);

}

void goForward ()

{

if (digitalRead(STBY) == HIGH)

{

analogWrite(PWMA, 500); //0-1023

Serial.println("PWMA");

delay(1000);

digitalWrite (AIN1, HIGH);

digitalWrite (AIN2, LOW);

Serial.println("avancer");

delay(1000);

}

}

void accelerer ()

{

if (digitalRead(STBY) == HIGH)

{

analogWrite(PWMA, 2000); //0-1023

Serial.println("PWMA");

delay(1000);

}

}

void ralentir ()

{

if (digitalRead(STBY) == HIGH)

{

analogWrite(PWMA, 500); //0-1023

Serial.println("PWMA");

delay(1000);

}

}

void goBackward ()

{

if (digitalRead(STBY) == HIGH)

{

analogWrite(PWMA, 500); //0-1023

Serial.println("PWMA");

digitalWrite (AIN1, LOW);

digitalWrite (AIN2, HIGH);

Serial.println("marche arriere");

delay(1000);

}

}

Annexe 2 : code Page web

**const char** MAIN\_page[] **PROGMEM** = R"=====(

<html>

<head>

<title>Moteur</title>

<script>

function getRandomInt(max) {

return Math.**floor**(Math**.random**() \* Math.**floor**(**max**));

}

function s(u){var x = new XMLHttpRequest();x.**open**("**GET**", u+**"?x="+**getRandomInt(10000), **true**);x.**send**(**null**);}

function ret(u){**return** function(e){

s(u);

e.returnValue **= false**;

**return** false;

}

}

window.addEventListener('load', function(){

var lf = document.getElementById('lf');

var rf = document.getElementById('rf');

var av = document.getElementById('av');

var rc = document.getElementById('rc');

var acc = document.getElementById('acc');

var re = document.getElementById('re');

lf.addEventListener('click', ret("**startUp**"), **false**);

rf.addEventListener('click', ret("**shutDown**"), **false**);

av.addEventListener('click', ret("**goForward**"), **false**);

rc.addEventListener('click', ret("**goBackward**"), **false**);

acc.addEventListener('click', ret("**accelerer**"), **false**);

re.addEventListener('click', ret("**ralentir**"), **false**);

}, **false**);

</script>

</head>

<body>

<**div**>

<div style="width:100%;heigth:100%">

<button id="lf" style="width:50%;height:35%;float:left;background-color: #008CBA;color: white;border-radius: 50%;font-size: 24px;"><strong>ON</strong></button>

<button id="rf" style="width:50%;height:35%;float:left;background-color: #008CBA;color: white;border-radius: 50%;font-size: 24px;"><strong>OFF</strong></button>

</**div**>

<**div**>

<button id="av" style="width:50%;height:35%;float:left;background-color: #008CBA;color: white;border-radius: 50%;font-size: 24px;"><strong>Avancer</strong></button>

<button id="rc" style="width:50%;height:35%;float:left;background-color: #008CBA;color: white;border-radius: 50%;font-size: 24px;"><strong>Reculer</strong></button>

</**div**>

<**div**>

<button id="acc" style="width:50%;height:35%;float:left;background-color: #008CBA;color: white;border-radius: 50%;font-size: 24px;"><strong>Accelerer</strong></button>

<button id="re" style="width:50%;height:35%;float:left;background-color: #008CBA;color: white;border-radius: 50%;font-size: 24px;"><strong>Relantir</strong></button>

</div>

</div>

</body>

</html>

)=====";

1. <https://www.openhab.org/> [↑](#footnote-ref-1)
2. <https://amiqual4home.inria.fr/fr/> [↑](#footnote-ref-2)
3. [↑](#endnote-ref-1)