



REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Mohamed Khider – BISKRA
Faculté des Sciences Exactes, des Sciences de la Nature et de la Vie
Département d'informatique

N° d'ordre :14RTIC/M2/2019

Mémoire

présenté pour obtenir le diplôme de master académique en

Informatique

Parcours : Réseaux et Technologies d'Informations et Télécommunications

Réalisation et implémentation d'une application a base de protocole MQTT dans IoT

Par :

HIDOUCI FARID

Soutenu le 07 juillet 2019, devant le jury composé de :

Bouekkache Samir	MCB	Président
Kahloul Laid	MCA	Rapporteur
Tibermacine Ahmed	MCB	Examineur

Remerciement

Je tiens à la fin de ce travail à remercier ALLAH le tout puissant de m'avoir donné la foi et de m'avoir permis d'en arriver là.

Mes vifs remerciements accompagnés de toute ma gratitude vont également à mon encadreur Mr KAHLUOL LAID. Je lui suis particulièrement reconnaissante pour son soutien, sa disponibilité, ses précieux conseils avisés, ses orientations et ses encouragements.

Je remercie mes amis AMMAR BELKACEMI et TAHA ARADJ pour leurs conseils et leur aide pour terminer ce travail.

Je remercie mon ami TAREK DJABER pour ses conseils et son aide pour terminer ce travail.

Mes remerciements vont également à ma mère, mon père de toutes les sacrifices qu'ils ont consentis pour me permettre de poursuivre mes études dans les meilleures conditions possibles et n'avoir jamais cessé de m'encourager tout au long de mes années d'étude.

Je remercie les membres de jury.

Je remercie ceux qui m'ont aidé à l'aboutissement de ce travail; ma formidable promotion 2019 du département d'Informatique de l'université Mohamed Khider Biskra.

Dédicace

Je dédie ce mémoire à ma mère et à mon père, et mes amis.

A mes frères et mes sœurs.

A tous mes oncles et tantes, cousins et cousines.

A mes collègues de travail.

A tous mes amis que je n'ai pas cités et à tous ceux qui me connaissent, les étudiants de ma promotion Master2 2019 Réseaux et Technologies d'Informations et Télécommunications de l'université Mohamed Khider Biskra.

A tous qui m'aidaient pour terminer ce travail.

A la fin, je veux dédier ce modeste travail à tous les enseignants et enseignantes qu'ils m'ont aidé à arriver ici depuis le début de mes études.

Résumé

L'internet a connu, depuis son apparition, un grand essor et a touché pratiquement à tous les domaines de notre vie de tous les jours : politique, économique et socioculturel, et donc il est devenu une principale source de l'information.

L'internet classique tel que nous le connaissons, à savoir l'accès à une multitude de site web à travers d'un navigateur, repose sur plusieurs protocoles. Un premier protocole utilisé est HTTP (Hyper Text Transfer Protocol). Il s'agit du protocole de couche applicative utilisé pour la communication web. Comme son nom l'indique, son rôle est de transférer des données hypertextes, du serveur jusqu'au client final qui souhaite cette ressource. Donc, l'internet classique consiste principalement à l'internet des données.

Le IoT (Internet of Things) est plus vaste et plus varié puisqu'elle exploite des innovante technologies et des normes que l'internet classique. L'apparition du IoT offre un grand potentiel pour le développement de nouveaux services et d'applications immergeant le monde physique et le monde virtuel, une perspective qui n'était pas possible auparavant.

De nombreuses applications de l'Internet des Objets mènent des projets pour améliorer la vie des gents grâce aux objets et infrastructures connectés. La ville de futur s'organise autour de plusieurs contraintes : environnement, contraintes énergétiques, collaboration entre les gents, les collectivités et les opérateurs de la ville, développement des modèles économiques orientés sur les usagés et l'intégration numérique dans la ville.

Ce projet vise à la mise en œuvre d'une application d'IoT sur la façon d'utiliser MQTT comme protocole de communication pour le déploiement d'applications IoT et la gestion à distance de ces applications, de sorte qu'il peut fonctionner dans toutes les conditions du réseau.

L'objectif de cette application est l'implantation de protocole MQTT dans les applications IOT et prendre en compte les valeurs transmises par les différents dispositifs situés dans l'environnement (capteur d'humidité, capteur de température), et visualiser ces données a distance sur un smart phone.

Mots clés : Internet of Things, Smart phone, IoT, MQTT.

ABSTRACT

The internet has, since its inception, experienced a great expansion and has touched practically every area of our everyday life: political, economic and socio-cultural, and thus it has become a major source of information.

The classic internet as we know it, namely the access to a multitude of web site through a browser, relies on several protocols. A first protocol used is HTTP (Hyper Text Transfer Protocol). This is the application layer protocol used for web communication. As the name implies, its role is to transfer hypertext data, from the server to the end customer who wants this resource. So, the classic internet mainly consists of the internet of data.

The Internet of Things (IoT) is larger and more diverse as it exploits innovative technologies and standards than conventional internet. The emergence of IoT offers great potential for the development of new services and applications that immerse the physical world and the virtual world, a perspective that was not possible before.

Many Internet of Things applications are leading projects to improve people's lives through connected objects and infrastructure. The city of the future is organized around several constraints: environment, energy constraints, collaboration between people, communities and operators of the city, development of user-oriented economic models and digital integration in the city.

This project aims to implement an IoT application on how to use MQTT as a communication protocol for IoT application deployment and remote management of these applications, so that it can work in all the conditions of the network.

The objective of this application is the implementation of MQTT protocol in IOT applications and take into account the values transmitted by the various devices located in the environment (humidity sensor, temperature sensor), and view these data remotely on a smart phone.

Keywords: Internet of Things, Smart phone, IoT, MQTT.

Table des matières

	Remerciement	II
	Résumé	IV
	Table des matières	VI
	Liste des figures	XI
	Liste des tableaux	XV
	Introduction Générale	XVI
	Chapitre 1 - L'internet des objets	1
1	Introduction.....	2
2	InternetofThings.....	3
2.1	Notion del'objet connecté.....	3
2.2	Caractéristiques d'un objet connecté.....	3
2.3	Définition de L'internet des objets.....	4
3	L'importance d'IoT.....	6
4	Les composants d'internet des objets.....	6
5	Domaines d'application d'IoT.....	7
6	Exemples d'objets connectés.....	8
7	Etude détaillée des composants d'un objet connecté.....	11
7.1	Le microcontrôleur.....	11
7.1.1	Définition d'un Microcontrôleur.....	11
7.1.2	Domaines d'application.....	12
7.1.3	Les différents éléments composants le microcontrôleur.....	12
7.1.4	Les différantes modeles de cartes microcontrolleur.....	15
7.2	Les plaquettes d'essais sans soudures(BreadBoard).....	19
7.3	Les shields.....	21
7.4	Les capteurs.....	22
7.4.1	Définition d'un capteur.....	22
7.4.2	Caractéristiques des capteurs.....	23
7.4.3	Critères de choix du capteur.....	23
7.4.4	Exemples de capteurs.....	24
7.5	Les actionneurs.....	31
7.6	Les câbles.....	32
7.7	La communication dans un systèmed'IoT.....	34
7.7.1	Technologies de communication.....	34
7.7.2	Protocoles de communication.....	38
7.8	Les LEDs.....	39
7.9	La résistance.....	40
8	Les plates-formes d'internet des objets.....	41
8.1	Définition d'une plate-forme d'IoT.....	41

Table des matières

8.2	Composants d'une plateforme IoT moderne.....	42
8.3	Différentes plates-formes d'IoT.....	43
8.4	Comparaison de quelques plateformes IoT.....	45
9	Les défis d'IoT.....	46
10	Conclusion.....	47
	Chpitre 2 : la communication dans IoT	48
1	Introduction.....	49
2	Les protocoles de transfert des données dans l'IoT.....	49
2.1	Hypertext Transfer Protocol, un protocole bien connu.....	49
2.2	CoAP - Contrainte Application Protocol, le protocole pour le web des objets.....	50
2.3	Message Queuing Telemetry Transport.....	53
3	Quel est le meilleur protocole?.....	53
4	Conclusion.....	55
	Chapitre 3- Le protocole MQTT	56
1	Introduction.....	57
2	Mode de fonctionnement.....	58
2.1	Topologies MQTT.....	58
2.2	Fonctionnement.....	59
2.2.1	Connexion et Déconnexion.....	59
2.2.2	Abonnements et Publications.....	60
2.3	Topic et motifs de filtrage.....	60
2.4	Qualité de service.....	61
2.5	Dernière volonté et testament (Last Will & Testament).....	63
2.6	Sécurité.....	64
3	Description informelle du protocole.....	64
3.1	Représentations de données.....	64
3.2	Structure d'un paquet de contrôle MQTT.....	64
3.2.1	Entête fixe.....	64
3.2.2	CONNECT.....	66
3.2.3	CONNACK.....	71
3.2.4	PUBLISH.....	72
3.2.5	PUBACK, PUBREC, PUBREL, PUBCOMP.....	73
3.2.6	SUBSCRIBE.....	74
3.2.7	SUBACK.....	75
3.2.8	UNSUBSCRIBE.....	76
3.2.9	UNSUBACK.....	76

Table des matières

3.2.10	PINGREQ, PINGRESP.....	77
3.2.11	DISCONNECT.....	77
3.3	La synchronisation des requêtes envoyées au broker.....	78
3.4	L'identifiant de client MQTT.....	78
3.5	Callbacks.....	79
4	Conclusion.....	81
	Chapitre 4 : Conception et réalisation de système de control à base de MQTT	84
1	Introduction.....	84
2	Analyse des besoins du système.....	84
2.1	Exigences Fonctionnelles.....	84
2.2	Exigences non fonctionnelles.....	85
2.3	Vision de la solution proposée.....	85
2.3.1	Besoin matériels.....	85
2.3.2	Besoin Software.....	85
3	Spécification des fonctionnalités du système.....	85
3.1	Identification des acteurs :	85
3.2	Identification des cas d'utilisation :.....	86
3.2.1	Description textuelle des cas d'utilisation.....	86
3.2.1.1	Diagramme de cas d'utilisation « Consultation des données ».....	87
3.2.1.2	Diagramme de cas d'utilisation « Acquisition des données ».....	88
3.3	Diagrammes de séquences.....	89
3.3.1	Diagrammes de séquence : Acquisition des données.....	89
3.3.2	Diagrammes de séquence pour le cas d'utilisation : Consultation des données.....	89
4	La conception	90
4.1	Architecture matérielle.....	90
4.2	Motivation des choix.....	91
4.2.1	Le choix de la carte Arduino	91
4.2.2	Le choix de Wifi.....	91
4.3	Description technique des despositif impliqué dans le système.....	91

Table des matières

4.4	Architecture logicielle.....	92
4.4.1	Description de l'architecture logicielle.....	93
4.5	Diagramme de classe.....	93
4.6	Diagramme d'activité.....	95
4.6.1	Diagramme d'activité : Consultation des données.....	95
4.6.2	Diagramme d'activité : Acquisition des données	96
5	La réalisation.....	96
5.1	Description des langages et outils utilisés.....	97
5.1.1	Les langages de programmation.....	97
5.2	Plateforme de développement et outils logiciels.....	97
5.2.1	Plateformes utilisés.....	97
5.2.2	Logiciels utilisés.....	98
5.3	Description de la réalisation physique de l'objet.....	98
5.3.1	Réalisation de l'objet connecté.....	99
6	Les codes source de diférant composant de système.....	100
6.1	Portion de code arduino pour l'objet connecté	100
6.2	Portion de code android pour les smart phones.....	102
6.3	Discription coté de broker.....	105
7	Résultat.....	106
7.1	Interface des l'application pendant l'exploitation de système.....	106
8	Conclusion.....	109
	Chapitre 5 : Comparisan avec RESTful	111
1	Introdution.....	112
2	Les éléments essentiels de l'architecture RESTful.....	112
2.1	identifiant de la ressource.....	112
2.2	Interface uniforme.....	112
2.3	la réprésantation.....	113
2.4	Sans état (Statelessness).....	113
3	Comparison entre le REST et MQTT.....	113

Table des matières

3.1	Client-Serveur / Publisher-Subscriber.....	113
3.2	URI / MQTT Sujet (topic).....	114
3.3	Requête -réponse / publish-subscribe.....	114
4	Conclusion.....	114
	Conclusion général	115
	bibliographie	117

Liste des figures

Figure 1.1 : Une nouvelle dimension pour l'IoT.....	5
Figure 1.2: Différentes Domaines d'application d'IoT.....	8
Figure 1.3: Parrot Capteur Intelligent pour Plantes Intérieur/Extérieur Résine 30,0 x 21,0 x 30,cm.....	8
Figure 1.4 : Arlo Pro Pack de 2 Caméras, Smart caméra HD grand angle avec batterie rechargeable 6 mois et audio bidirectionnel, Intérieure / extérieure et alarme intégré	9
Figure 5 : Capteur pour plantes PARROT FLOWERPOWERVERT.....	9
Figure 1.6 : la montre sport GPS / Cardio Garmin Forerunner 735XT.....	10
Figure 1.7 : Le casque Oculus Go.....	10
Figure 1.8 : Lentille connectée Samsung.....	11
Figure 1.9: Microcontrôleur.....	11
Figure 1.10 : Architecture interne d'un microcontrôleur.....	12
Figure 1.11 : Carte Arduino modèle UNO REV3.....	15
Figure 1.12 : Carte Arduino modèle LEONARDO.....	16
Figure 1.13 : Carte Arduino modèle Yún.....	17
Figure 1.14 : Le Raspberry Pi modèle A.....	18
Figure 1.15 : Le Raspberry Pi modèle B.....	18
Figure 1.16 : Le Raspberry Pi modèle A+.....	18
Figure 1.17 : Le Raspberry Pi modèle B+.....	18
Figure 1.18 : Le Raspberry Pi modèle 2.....	19
Figure 1.19 : Le Raspberry Pi modèle 3.....	19
Figure 1.20 : Une plaquette d'essais sans soudure.....	19
Figure 1.21 : BreadBoard SD12N.....	20
Figure 1.22 : BreadBoard SD35N.....	20

Liste des figures

Figure 1.23 : Shield de prototypage arduino UNO avec plaquette «essai».....	21
Figure 1.24 : ShieldMétéoDEV12081.....	21
Figure 1.25 : principe de fonctionnementd'uncapteur.....	22
Figure 1.26 : Capteur d'humidité–Grove.....	24
Figure 1.27 : Capteur d'humidité–Grove.....	24
Figure 1.28 : Capteur d'humidité–Grove.....	25
Figure 1.29 : Capteur d'humidité–Grove.....	25
Figure 1.30 : Capteur de température et humidité digital–DHT22.....	26
Figure 1.31 : Capteur de température et humiditéSENSIRONSH31-D.....	26
Figure 1.32 : Capteur de température et humidité–GROVE.....	26
Figure 1.33 : Capteur detempérature LM35DZ.....	27
Figure 1.34 : Capteur de températurenumériqueTCN75A.....	27
Figure 1.35 : Capteur de pressionFreescale MPX4115A.....	28
Figure 1.36 : Capteur de pression VellemanMM103.....	28
Figure 1.37 : Capteur de Météo.....	29
Figure 1.38 : Capteur de mouvement PIRavec réglages.....	30
Figure 1.39 : Capteur demouvement Grove.....	30
Figure 1.40 : ElectrovanneG1/2 12V.....	31
Figure 1.41 : Mini SolénoïdePush-Pull 5V.....	31
Figure 1.42 : Câble alimentation arduino pourpiles9V.....	33
Figure 1.43 : Câble USB vers fiche DC JACK 2.1mm– 1m.....	33
Figure 1.44 : Câble antenne d'adaptation RP-SMAversuFL--u.FL-IPX-IPEX-RF.....	33
Figure 1.45 :Câble USB.....	33
Figure 1.46 : Classement des réseaux sans fil.....	35

Liste des figures

Figure 1.47 : Extrait d'un tableau comparatif de protocoles.....	38
Figure 1.48 : Principe de fonctionnement du protocole MQTT.....	38
Figure1. 49 :les LEDs.....	39
Figure1. 50 : Symbole d'une résistance électrique.....	40
Figure 1.51 : Résistance a4anneaux.....	40
Figure 1.52 : Composants d'une plateforme IoT moderne.....	42
Figure 2.1: Protocole CoAP activer les services Web en intégrant `a l'architecture Web et HTTP.....	50
Figure 2.2: CoAP diagramme.....	51
Un acc`es web demande l'utilisation de CoAP exemple `a l'aide de ressources `a distance via URI.....	51
Figure 2.4: Observer ressources dans CoAP.....	52
Figure 2.5: un message CoAP avec l'en-t`ete seulement 4 octets (en-t`ete, option, donn´ee).....	52
Figure 3-1 : Google Trends pour MQTT.....	57
Figure.3-2 : Exemples de topologies MQTT.....	59
Figure 3-3 : Souscription au topic.....	61
Figure 3-4 : Niveaux de Qualité de Service MQTT	62
Figure 3-5 : entête fixe du protocole MQTT.....	64
Figure 3-6. : Format d'un paquet CONNECT.....	66
Figure 3-7 : Exemple d'utilisation du drapeau clean session.....	69
Figure 3-8 : Exemple d'utilisation du drapeau Keep alive.....	70
Figure 3-9 : Format d'un paquet CONNACK.....	71
Figure 3-10 : Format d'un paquet PUBLISH.....	72
Figure 3-11 : Format d'un paquet PUB [ACK/REC/REL/COMP].....	73
Figure 3-12 : Format d'un paquet SUBSCRIBE.....	74
Figure 3-13 : Format d'un paquet SUBACK.....	75

Liste des figures

Figure 3-14 : Format d'un paquet UNSUBSCRIBE.....	76
Figure3-15 : Format d'un paquet UNSUBACK.....	76
Figure 3-16. Format d'un paquet PING [REQ/RESP].....	77
Figure 3-17. Format d'un paquet DISCONNECT.....	77
Figure 4.1: Diagramme de cas d'utilisation général.....	87
Figure 4.2 : Diagramme de cas d'utilisation « Consultation des données ».....	88
Figure 4.3 : Diagramme de cas d'utilisation « Acquisition des données ».....	89
Figure 4.4 : Diagramme de séquence : Acquisition des données.....	90
Figure 4.5 : Diagramme de séquence : Consultation des données.....	90
Figure 4.6 : Architecture matérielle du système.....	91
Figure 4.7 : Architecture logicielle du système.....	93
Figure 4.7 : diagramme de classe.....	95
Figure 4.8 : Diagramme d'activité " Consultation des données ".....	96
Figure 4.9 : Diagramme d'activité " Acquisition des données ".....	97
Figure 4.10 : logo C++.....	98
Figure 4.11: Logo Android.....	98
Fiquer4.12 :logo Arduino IDE.....	99
Figure 4.13 : L'objet connecté en état éteint	100
Figure 4.14 : L'objet connecté en état allumé.....	101
Figure 4.15 : Code de connection au broker.....	101
Figure 4.16 : code d'initialisation module Wifi.....	102
Figure 4.17: code de calcule et piblier les topics.....	102
Figure 4.18 : code de connexion au broker.....	104
Figure 4.19 : code d'inscription au topic.....	104
Figure 4.20 : code de désinscription au topic.....	105
Figure 4.21 : code source de déconnexion au broker.....	105
Figure 4.22 : discription de l'interface de l'application mobile.....	106
Figure 4.23 : mosquito broker démarrer.....	107
Figure 4.24 : acquisition des données.....	107
Figure 4.25 : l'inscription et l'envoi des données par le broker.....	108
Figure 4.26 : connexion au broker par smart phone.....	108
Figure 4.27 : inscription au topic par smart phone.....	109
Figure 4.28 : affichage du résultat avec smart phone.....	109

Liste des tableaux

Tableau 1.1 : Caractéristiques techniques de le carte ArduinoUNO REV3.	16
Tableau 1.2 : Caractéristiques techniques de le carteArduinoLEONARDO	16
Tableau 1.3 : Caractéristiques techniques de le carteArduinoYún.	17
Tableau 1.4 : Description technique de laBreadBoardSD12N.	20
Tableau 1.5 : Description technique de laBreadBoardSD35N.	20
Tableau 1.6 : Shield de prototypage arduino UNO avec plaquetted »essai.	21
Tableau 1.7 : Description et caractéristiques d'un shield demétéoDEV12081	21
Tableau 1.8 : Description de deux types decapteur d'humidité.	24
Tableau 1.11 : Description et caractéristiques de quelques modèles de capteurs De températureetd'humidité.....	26
Tableau 1.12 : Description et caractéristiques de quelques modèles de capteurs de température.	27
Tableau 1.13 : Description et caractéristiques de quelques modèles de capteurs de pression.	28
Tableau 1.14 : Description et caractéristiques du capteur de météo et ses composantes	29
Tableau 1.15 : Description et caractéristiques de quelques modèles du capteur de mouvement.	30
Tableau 1.16 : Description et caractéristiques de l'actionneur : ElectrovanneG1/212V	31
Tableau 1.17 : Description et caractéristiques de quelques modèles de câblesutilisés.....	33
Tableau 1.18 : Comparaison entre quelques plateformes IoT.....	46
Tableau 1.28 : Fiche descriptive du cas d'utilisation : " Gestion de l'objet connecté".....	102
Tableau 3.1. :Type des paquets MQTT.....	65
Tableau 3-2. :Description du drapeau dans l'entête fixe du protocole MQTT.....	66
Tableau 3-3. : Les valeurs du code de retour.....	71
Tableau 3-4 :.Code de retour de paquet SUBACK.....	75
Tableau 3-5 :.Comportement de synchronisation des méthodes qui donnent lieu à des requêtes au broker.....	78
Tableau 4.1 : Fiche descriptive du cas d'utilisation : " Consultation des données ".....	88
Tableau 4.2 : Fiche descriptive du cas d'utilisation : " Acquisition des données ".....	89
Tableau 4.3 : Description textuelle du diagramme de classe.....	95
Tableau 5.1 : Description textuelle du diagramme de classe.....	95

INTRODUCTION GÉNÉRALE

Introduction générale

1. Contexte de travail

L'internet des objets est un nouvel outil de connectivité et de mobilité, qui transforme les affaires et la vie quotidienne à des objets connectés.

Les objets courants deviennent actifs et intelligents, s'intégrant de façon transparente à un réseau mondial et sont en mesure de produire et d'échanger des données utiles sans intervention humaine. Il s'agit d'un réseau de réseaux qui permet, via des systèmes d'identification électronique normalisés et sans fil, d'identifier et de communiquer numériquement avec des objets physiques afin de pouvoir mesurer et échanger des données entre les mondes physiques et virtuels. A travers un tel paradigme, aujourd'hui l'IoT couvrira un large éventail d'applications et touchera quasiment à tous les domaines que nous affrontons au quotidien, ceci permettra l'émergence d'espaces intelligents [1].

Notre monde connaît une prolifération étonnante des objets intelligents ce qui a poussé les sociétés des services de cloud à mettre à la disposition des intéressés des plateformes connues sous le nom de plateforme d'internet des objets (plateformes d'IoT) qui contient tous ce que l'internaute ciblé a besoin: prestations, analyses, statistiques, bibliothèques et des remèdes (solutions) qui facilitent aux programmeurs et aux développeurs l'opération de la communication et de sa maîtrise, ainsi qu'elles permettent d'accélérer et de réduire le coût de développement de produits et applications IoT.

2. Problématique et objectifs

Les dispositifs IoT peuvent être intégrés dans le Web à l'aide des ressources RESTful pour exposer la fonctionnalité des dispositifs IoT [2]. REST utilise URI comme identifiants des ressources sur le Web. Et dans le cas des systèmes IoT, URI sera utilisé pour identifier les périphériques IoT comme des ressources ou des ressources fournies par les dispositifs IoT.

La plupart des dispositifs IoT : capteurs, actionneurs et systèmes embarqués et sont nombre en des millions et augmente de jour par jour. Considérons des millions d'appareils qui vont active et/ou passive dans leur fonctionnement, il sera impossible d'allouer une adresse IP pour tous les dispositifs [3]. L'adresse IPv4 ne suffira pas pour l'adressage de tous les périphériques IoT [4]. Par conséquent, un schéma tel qu'un (VPN) ou (NAT) doit être utilisé pour connecter tous les dispositifs IoT disponibles à Internet.

NAT apparaît pour extraire l'adresse des périphériques qui fonctionnent derrière. Cela peut être un problème dans REST, car il est basé sur une requête-réponse HTTP entre le client et le serveur et que ceux-ci doivent avoir une adresse IP pour communiquer entre eux.

Par conséquent, il peut y avoir un problème lorsque le serveur essaie de communiquer avec le client qui réside derrière le NAT. Ici, le serveur ne peut pas se connecter au client car il ne connaît pas l'adresse IP du client qui se trouve derrière le NAT.

La solution pour la communication entre serveur et client dans le cas NAT peut être d'utiliser le protocole qui ne dépend pas de l'adresse IP des nœuds finaux client-serveur mais se concentre uniquement sur la livraison de données entre ces points finaux.

3. Description de la solution proposée

MQTT est un protocole qui est léger, centrée sur les données, et spécialement conçus pour les dispositifs limités en ressources comme des dispositifs IoT [5]. MQTT utilise publish-subscribe messagerie pour la communication entre les dispositifs IoT.

Lors de l'utilisation de ce protocole, les dispositifs n'ont pas besoin de connaître l'existence d'autres appareils. Tous les appareils publient les messages s'abonnent au broker, et ce broker se chargera de livrée des messages aux abonnés. Par conséquent, en utilisant MQTT, la communication entre les dispositifs IoT devrait fonctionner sur le réseau local, ainsi que dans le réseau qui utilise transformation d'adresses réseau (NAT).

Notre projet s'inscrit dans le cadre des systèmes d'IoT (Internet of Things), et il s'intéresse particulièrement à l'implémentation du Protocol MQTT dans IoT et de prendre en compte les valeurs transmises par les différents dispositifs situés dans l'environnement cible (capteur d'humidité, capteur de température), pour effectuer la lecture a distance et mobile. Pour atteindre ces objectifs, il est nécessaire d'acquérir une bonne connaissance de l'Internet des objets (IoT). L'Etude du protocole de transfert de données MQTT et ses application.

4. *Structuration de mémoire*

Le reste du mémoire est organisé comme suit :

A. **Partie 1** : État del'art :

- i. **Chapitre 1** : Internet des Objets (Internet of Things) (définition, composantes, importance, domaines d'applications, les objets connectés,...)
- ii. **Chapitre 2** : La communication dans IoT.
- iii. **Chapitre 3** : le protocole MQTT.

B. **Partie 2** : Contributions

- i. **Chapitre 4** : conception et réalisation de système de contrôle a base de MQTT (objectif, conception, réalisation, résultat.)
- ii. **Chapitre 5** : comparaison MQTT et RESTful

En fin Conclusion général.

PREMIÈRE PARTIE : ETAT DE
L'ART

CHAPITRE 1 : INTERNET DES OBJETS (INTERNET OF THINGS)

1. Introduction

L'internet a connu depuis son apparition, un grand essor et a touché pratiquement tous les domaines de notre vie de tous les jours : politique, économique, socioculturel,...etc. Elle est devenue la principale source d'information.

L'internet a également facilité la Communication entre les gens, en transformant le monde en une petite ville.

Le progrès scientifique et technologique a eu, spécialement dans le domaine de l'électronique , de la communication des objets intelligents et des systèmes de capteurs très développés, variés et de différentes tailles, des répercussions positives sur l'internet et l'apparition également d'un nouveau paradigme qui est « l'Internet des Objets », en anglais : Internet of Things(IoT).

L'IoT constitue la prochaine génération de la révolution numérique. Ce sont des milliards d'objets du quotidien qui seront connectés à l'horizon 2020. Les objets connectés ont la possibilité de simplifier la vie quotidienne et d'améliorer le bien-être des individus, chez eux, dans leur jardin privé, au bureau, en famille ou au sein de toute autre communauté [6]. Un objet connecté peut être une personne avec un implant cardiaque qui transmet des données, un objet intelligent pour l'économie de l'eau permettant ainsi le contrôle automatique et intelligent de l'environnement, un animal qui porte une puce intelligente, une voiture dotée de capteurs indiquant la pression des pneus, ou tout autre objet auquel une adresse IP est assignée et qui peut transmettre des informations.

En outre, l'Internet des Objets a la possibilité d'intégration avec le réseau Internet, ce qui va permettre une connectivité pour tout le monde, tout le temps et partout et idéalement depuis n'importe quelle plate-forme [7].

Ce chapitre est consacré à l'étude des systèmes d'IoT. L'objectif de telle étude est de comprendre leurs caractéristiques et spécificités afin de pouvoir développer un système d'IoT assurant le transfert des données à base de protocole MQTT.

2. Internet des objets

2.1 Notion de l'objet connecté

L'IoT repose avant tout sur les objets connectés. On peut dire qu'un objet connecté a la capacité de capter une donnée et de l'envoyer, via le réseau Internet ou d'autres technologies, pour que celle-ci soit analysée et visualisée sur des tableaux de bord dédiés. Les objets connectés interagissent avec leur environnement par le biais de capteurs (température, vitesse, humidité, vibration...). Dans l'Internet des Objets, un objet peut aussi bien être un véhicule, une machine industrielle ou bien une place de parking [8].

Un objet connecté est défini comme un équipement possédant les sept attributs suivants :

- i. Capteurs
- ii. Connectivité à internet
- iii. Processeurs
- iv. Efficacité énergétique
- v. Coût optimisé
- vi. Fiabilité
- vii. Sécurité

2.2 Caractéristiques d'un objet connecté

Généralement, un objet connecté est caractérisé par [9]:

- **Identité** : pour que les objets soient gérables il est essentiel que chaque objet connecté possède une identité unique qu'il lui propre et qui le distingue des autres objets du système.
- **Interactivité** : les progrès technologiques ont permis de connecter une grande variété d'objets et de dispositifs. Un objet n'a pas besoin d'être connecté à un réseau à tout moment. Pour des objets dits passifs tels que des livres ou des DVD, des étiquettes RFID doivent seulement être en mesure de signaler leur présence, de temps en temps, comme le moment de quitter le magasin.
- **Programmable**: l'objet connecté doit être programmé et piloté à distance via un ordinateur, une tablette ou un Smartphone.
- **Sensibilité** : un objet a la capacité de percevoir son environnement et peut collecter ou transmettre des informations à celui-ci. Il peut ainsi avoir des capteurs signalant les niveaux de température, d'humidité, de vibrations, d'emplacement ou de bruit.
- **Autonomie** : cette caractéristique est, peut-être, la caractéristique la plus importante pour l'objet connecté. On désigne par cette caractéristique la capacité de l'objet d'agir sans l'intervention d'un tiers. En d'autres termes, les objets doivent pouvoir être traités et surveillés individuellement, généralement depuis un point éloigné, et doivent fonctionner indépendamment d'une télécommande, c.-à-d. que chaque objet devient responsable de lui-même.

2.3 Définition de L'internet des objets

L'expression «Internet des objets» a vu le jour il y a une dizaine d'années par les fondateurs du MIT Auto-ID Center original, avec une mention spéciale du pionnier de la technologie britannique « Kevin Ashton » en 1999¹, pour le but d'établir un système dans lequel les objets du monde physique peuvent être connectés à l'internet par des capteurs.

Ashton a proposé le terme pour montrer la puissance des étiquettes d'identification par la radiofréquence (RFID), utilisées dans les chaînes d'approvisionnement d'entreprise, à l'Internet pour compter et suivre les marchandises sans intervention humaine [10].

L'internet des objets se situe à l'intersection des domaines de l'informatique et des communications électroniques, où tout objet communique, peut être interrogé, envoie de l'information et interagit avec son environnement.

UIT² définit ainsi, en 2012, l'internet des objets comme une « infrastructure mondiale pour la société de l'information, qui permet de disposer de services évolués en interconnectant des objets (physiques ou virtuels) grâce aux technologies de l'information et de la communication interopérables existantes ou en évolution » [11].

De manière plus pragmatique, l'internet des objets correspond à un ensemble d'objets connectés, de communications et d'internet, qui se conjugue avec les vagues du cloud et du Big-data :

- ✓ Les objets physiques possèdent des technologies embarquées de capteurs, d'intelligence et de connectivité, leur permettant de communiquer avec d'autres objets.
- ✓ Les réseaux de communications électroniques permettent de transporter les données issues des objets.

Il n'existe pas de définition standard, unifiée et partagée de l'internet des objets.

Certaines définitions insistent sur les aspects techniques de l'IoT, tandis que d'autres se concentrent plutôt sur les usages et les fonctionnalités [12].

¹ "Je peux me tromper, mais je suis à peu près sûr que l'expression "Internet of Things" a commencé comme titre d'une présentation que j'ai faite chez Procter & Gamble (P & G) en 1999", Kevin Ashton, RFID Journal, 22 juin 2009.

² L'Union Internationale des Télécommunications.

l'IEEE³ définit l'IoT comme un « réseau d'éléments chacun muni de capteurs qui sont connectés à Internet » [13].

Le CERP-IoT⁴ définit l'Internet des objets comme : « une infrastructure dynamique d'un réseau global.Ce réseau global a des capacités d'auto-configuration basée sur des standards et des protocoles de communication interopérables. Dans ce réseau, les objets physiques et virtuels ont des identités, des attributs physiques, des personnalités virtuelles et des interfaces intelligentes, et ils sont intégrés au réseau d'une façon transparente » [14].

Dans [14,15] les auteurs considère l'IoT comme étant un réseau qui permet, via des systèmes d'identification électronique normalisés et unifiés, et des dispositifs mobiles sans fil, d'identifier directement et sans ambiguïté des entités numériques et des objets physiques et ainsi de pouvoir récupérer, stocker, transférer et traiter, sans discontinuité entre les mondes physiques et virtuels, les données s'y rattachant. Cette vision de l'Internet des objets introduira une nouvelle dimension aux technologies de l'information et de la communication : en plus des deux dimensions temporelle et spatiale qui permettent aux personnes de se connecter de n'importe où à n'importe quel moment, une nouvelle dimension donc est ajoutée, «*objet* », qui leur permettra de se connecter à n'importe quel objet (voir la Figure1).

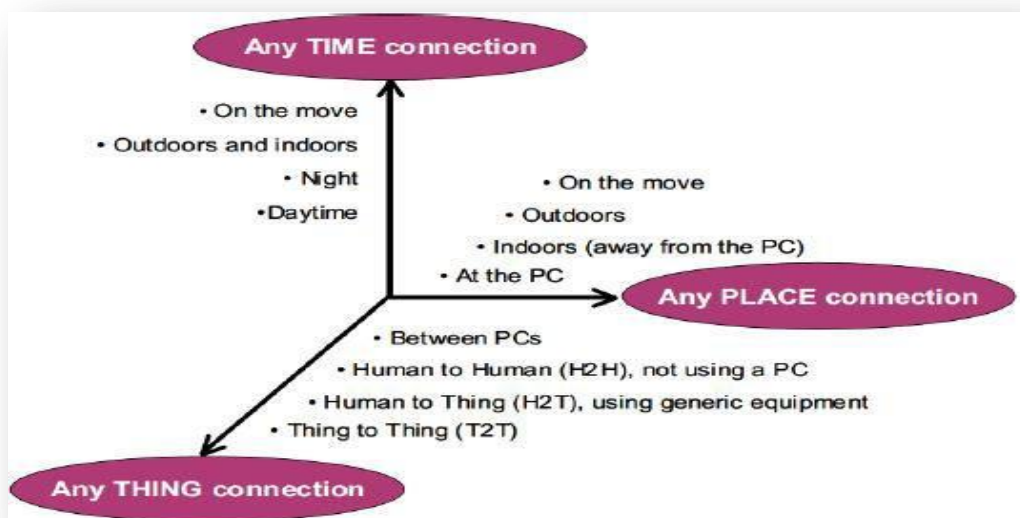


Figure 1.1: Une nouvelle dimension pour l'IoT [14,15]

³ Institute of Electrical and Electronics Engineers

⁴ Cluster des projets européens de recherche sur l'Internet des objets

3 L'importance d'IoT

L'importance de l'IoT devient considérable, puisqu'il s'agit de la première véritable évolution de l'Internet.

Celle-ci donnera lieu à des applications révolutionnaires capables de transformer profondément notre mode de vie, et notre façon d'apprendre, de travailler et de nous divertir. L'IoT a déjà doté l'Internet de capacités sensorielles (température, pression, vibration, luminosité, humidité, tension), ce qui nous permet d'anticiper plutôt que de simplement réagir.

En outre, l'Internet couvre maintenant des endroits jusqu'alors inaccessibles. Des patients ingèrent même des dispositifs connectés qui aident les médecins à diagnostiquer certaines pathologies et à en déterminer les causes, des capteurs extrêmement miniaturisés peuvent être placés sur des plantes, des animaux et des sites géologiques, et connectés à l'Internet, dans le même temps, l'Internet est en train de conquérir l'espace grâce au programme IRIS⁵ de Cisco⁶ [6].

4 Les composants d'internet des objets

Le concept d'Internet des Objets exige la coordination des dispositifs suivants [16] :

- une étiquette physique identifie chaque objet / une étiquette virtuelle identifie chaque lieu;
- un dispositif mobile (téléphone cellulaire, organiseur, ordinateur portable...) doté d'un logiciel additionnel, lit les étiquettes physiques ou localise les étiquettes virtuelles;
- un réseau sans fil relie le dispositif portable à un serveur contenant l'information relative à l'objet étiqueté;
- les informations sur les objets sont gérées dans des pages existantes du web;
- un dispositif d'affichage (écran d'un téléphone mobile) permet de consulter les informations relatives à l'objet ou à un ensemble d'objets.

⁵ Internet Routing inSpace

⁶ Le programme de routage Internet de Cisco dans l'espace (IRIS) utilise le routeur spatial Cisco et des satellites pour étendre l'accès IP. Ce routeur ne nécessite aucune station terrestre pour l'envoi et la réception de données, ce qui permet d'économiser du temps et de l'argent. En outre, les routeurs spatiaux de Cisco étendent l'accès IP aux zones qui ne sont pas desservies par les réseaux terrestres traditionnels ni par les réseaux 3G. Ils fournissent des capacités IP constantes et omniprésentes, quel que soit le lieu.

5 Domaines d'application d'IoT

L'IoT couvrira un large éventail d'applications et touchera quasiment à tous les domaines que nous affrontons au quotidien, ceci permettra l'émergence d'espaces intelligents (Figure 2). Parmi ces espaces intelligents, on peut citer :

- ✓ **Les villes intelligentes (Smart Cities)** [14] : l'IoT permettra une meilleure gestion des réseaux divers qui alimentent nos villes (eaux, électricité, gaz, etc.) en permettant un contrôle continu en temps réel et précis. Des capteurs peuvent être utilisés pour l'économie de l'eau et pour améliorer la gestion des parkings et du trafic urbain et diminuer les embouteillages et les émissions en CO₂.
- ✓ **La santé (Smart Health)** [14] : dans le domaine de la santé, l'IoT permettra le déploiement de réseaux personnels pour le contrôle et le suivi des signes cliniques, notamment pour des personnes âgées, les objets connectés permettent de suivre la tension, le rythme cardiaque, la qualité de respiration ou encore la masse grasseuse. Ceci permettra ainsi de faciliter la télésurveillance des patients à domicile, et apporter des solutions pour l'autonomie des personnes à mobilité réduite.
- ✓ **Le Transport** [17] : Des voitures connectées ou autonomes aux systèmes de transport/logistique intelligents, l'IoT peut sauver des vies, réduire le trafic et minimiser l'impact des véhicules sur l'environnement.
- ✓ **L'énergie** [17] : L'Internet des objets permet aux innombrables appareils qui composent le réseau électrique de partager des informations en temps réel pour une distribution et une gestion plus efficaces de l'énergie.
- ✓ **L'industrie** [14] : La technologie IoT permettra un suivi total des produits, de la chaîne de production, jusqu'à la chaîne logistique et de distribution en supervisant les conditions d'approvisionnement. Cette traçabilité de bout en bout permet aux usines d'améliorer l'efficacité de ses opérations, d'optimiser la production et d'améliorer la sécurité des employés.
- ✓ **Le bien-être et le confort** [18] : La domotique ou la maison intelligente est un classique. Imaginez un instant que votre thermostat soit capable de se mettre en marche tout seul en fonction de l'emplacement de votre voiture vous permettant de vous réchauffer une fois rentré à la maison. Aussi, imaginez que votre réfrigérateur vous informe lorsque vous aurez besoin d'acheter du lait ou qu'il soit capable de créer une liste d'achats personnalisée en fonction de vos articles les plus achetés. Ou encore vous dire quand votre nourriture est sur le point de périmer!

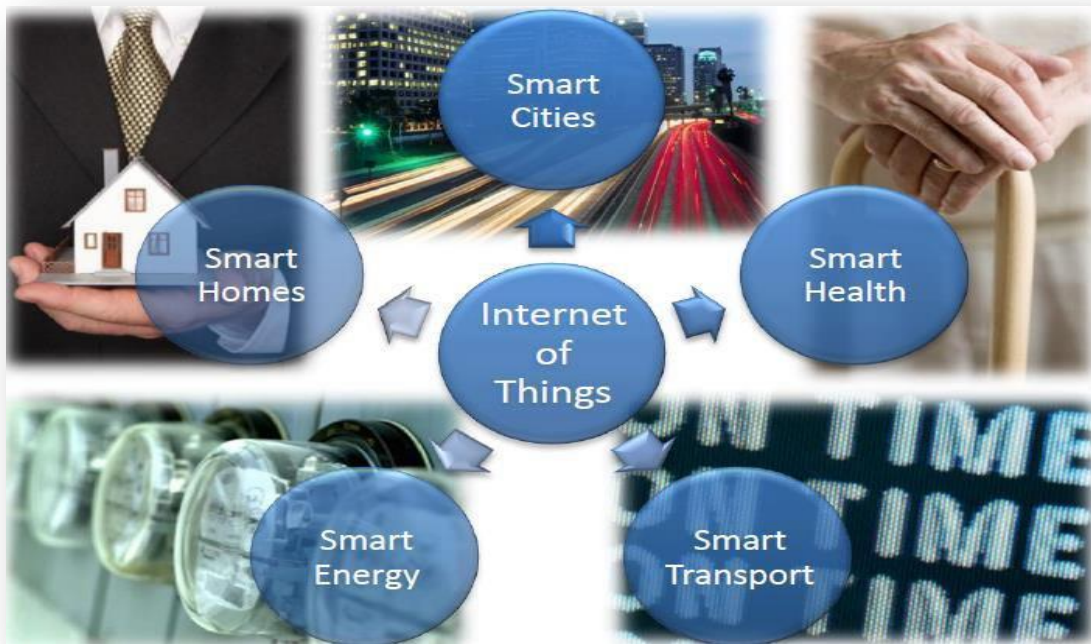


Figure 1.2 : Différentes Domaines d'application d'IoT [16]

6 Exemples d'objets connectés

- **Parrot POT (Pot de fleurs intelligent et connecté pour plantes)**



Figure 1.3 : Parrot Capteur Intelligent pour Plantes Intérieur/Extérieur Résine 30,0 x 21,0 x 30,0 cm [19] [20]

Un pot de fleur connecté qui s'occupe de vos plantes ! Doté de divers capteurs et d'un réservoir d'eau, le POT pourra entretenir vos fleurs même lors de vos absences du domicile. L'application vous donnera des conseils d'entretien et des rappels divers. Un gadget idéal pour les personnes qui n'ont pas forcément la main verte [19].

➤ **Netgear Arlo Pro 100% sans-fil**



Figure 1.4 : Arlo Pro Pack de 2 Caméras, Smart caméra HD grand angle avec batterie rechargeable 6 mois et audio bidirectionnel, Intérieure / extérieure et alarme intégrée [19] [21]

La plupart des caméras connectées utilisent une alimentation filaire, Netgear fait le pari un peu fou du 100% sans-fil. La solution Arlo Pro comporte une base qui peut ensuite contrôler plusieurs caméras adaptées aux usages intérieurs et extérieurs. De plus, Netgear offre une excellente solution d'enregistrement sur le Cloud adaptable à tous vos besoins pour la surveillance [19].

➤ **Flower power**



Figure 1.5 : Capteur pour plantes PARROT FLOWER POWER VERT [22]

*Parrot, célèbre pour ses drones, œuvre aussi dans le jardinage avec le Flower Power. Ce capteur, au look de lance-pierre, se plante dans la terre (en pot ou dans le sol) juste à côté de la plante à surveiller. Le Flower Power mesure quatre paramètres : la **luminosité**, la **température**, l'**engrais** et l'**humidité**. Lorsque votre téléphone est à proximité du capteur (et avec le Bluetooth activé), les données récoltées sont synchronisées, et vous êtes informé : elle manque un peu d'eau, d'engrais, la température est bonne, etc. [22].*

➤ **La montre sport (Garmin Forerunner735XT)**



Figure 1.6 : la montre sport GPS / Cardio Garmin Forerunner 735XT [23]

La montre sport (Garmin Forerunner 735XT) est une montre destinée aux sportifs et notamment aux coureurs. Elle offre une autonomie allant jusqu'à 11 heures et dispose d'un GPS et cardio-fréquence-mètre intégré, le tout piloté via un écran tactile. Le point fort de cette Forerunner est toutefois la possibilité d'organiser des entraînements comme pour des professionnels. Si elle parlera aux adaptés de running avec l'analyse de la foulée, les nageurs et les cyclistes pourront compter sur des modes permettant d'analyser la nage et le pédalage. On retrouvera également les suivies des VO2 Max et des aides à la récupération [23].

➤ **Le casque OculusGo**



Figure1. 7 : Le casque Oculus Go [24]

Pas besoin de PC ou de smartphone pour faire fonctionner le nouveau casque de Facebook, au contraire des concurrents HTC Vive ou Samsung Gear VR par exemple. L'Oculus Go propose un champ de vision à 110 degrés et un écran LCD doté d'une définition 2560 x 1440 pixels. Il intègre des haut-parleurs qui offrent un son "spatialisé" et une prise jack pour le casque audio. Accompagné de son contrôleur, il sera vendu 199 dollars.

Sortie prévue : "Au début de l'année 2018 pour les États-Unis" selon Facebook. Sûrement un peu plus tard dans l'Hexagone [24].

➤ **Lentilles connectées Samsung**

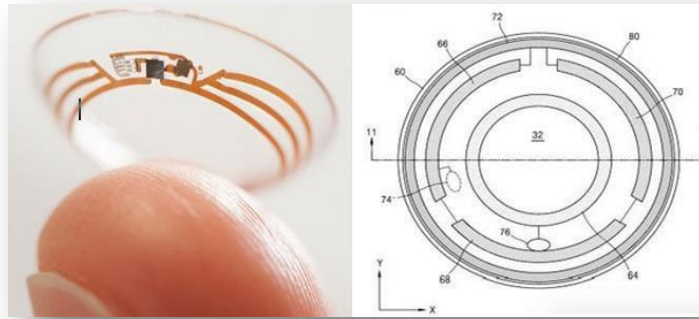


Figure 1.8 : Lentille connectée Samsung [25]

C'est là un gros projet que Samsung s'est mis en tête de réaliser. Mais étant donné que le projet ne reste pour l'instant qu'un simple brevet que la marque a déposé dans le but de ne pas se faire voler son idée, il faudra encore laisser ces lentilles quelques temps dans le domaine de la science-fiction... En effet, nombreuses sont les marques ayant déposé un nombre invraisemblable de brevets sans jamais les avoir vu passer au stade de production ou même de développement.[25].

7 Etude détaillée des composants d'un objet connecté

7.1 Le microcontrôleur

7.1.1 Définition d'un Microcontrôleur

Un microcontrôleur se présente sous la forme d'un circuit intégrés qui contient en interne, l'équivalent de la structure complète d'un micro-ordinateur. C'est-à-dire qu'il rassemble dans un même boîtier un microprocesseur, plusieurs types de mémoires et des périphériques de communication (Entrées-Sorties) [26].

Les microcontrôleurs améliorent l'intégration et le coût (lié à la conception et à la réalisation) d'un système à base de microprocesseur en rassemblant ces éléments essentiels dans un seul circuit intégré. Nous parlons alors de « système sur une puce » (en anglais : « System on Chip ») [27].

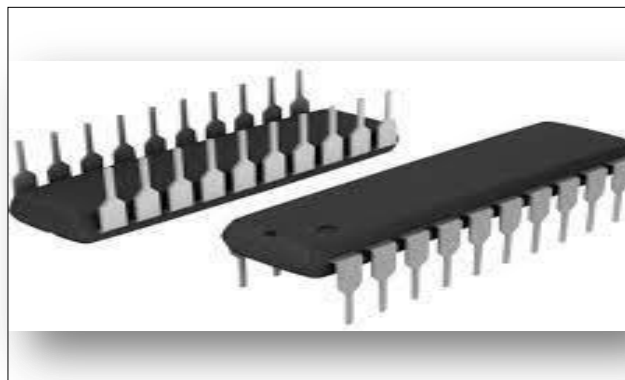


Figure 1.9 : Microcontrôleur

7.1.2 Domaine d'application

Les microcontrôleurs sont fréquemment utilisés dans les systèmes embarqués, de multiples industries et applications, notamment dans les domaines de l'automatisation, de la fabrication, de la robotique, de l'automobile, de l'éclairage, de l'énergie intelligente, de l'automatisation industrielle, des communications et de l'internet des objets [27].

7.1.3 Les différents éléments composants le microcontrôleur

Un microcontrôleur dispose d'un ensemble d'éléments qui ont chacun une fonction bien déterminée. La figure suivante montre ces éléments dont voici leurs fonctions :

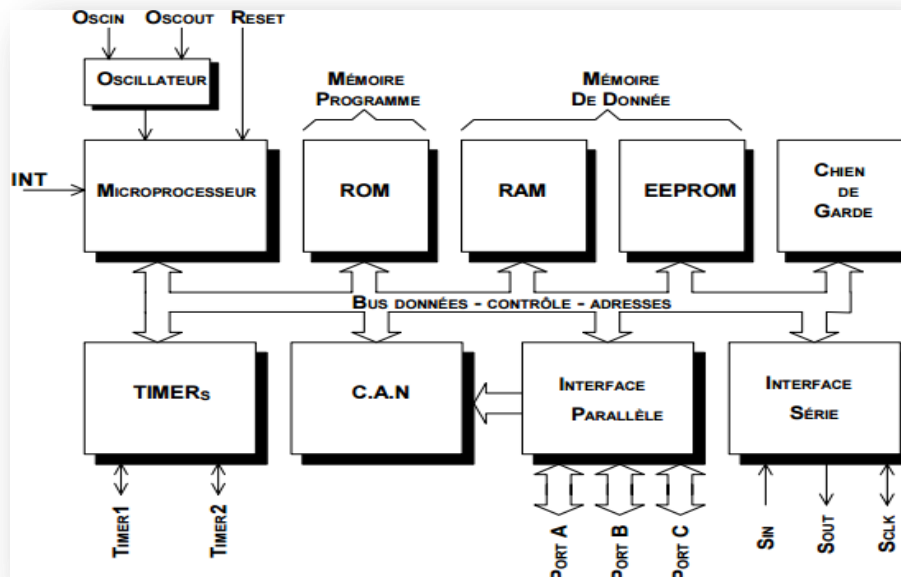


Figure 1.10 : Architecture interne d'un microcontrôleur [28]

- **Le processeur ou CPU (Central Processing Unit) :** Est le cœur du microcontrôleur. C'est l'équivalent du microprocesseur avec une puissance généralement moindre, la vocation n'étant pas la même. C'est cette unité centrale qui exécute le programme et pilote ainsi tous les autres éléments. Elle dispose généralement de deux connexions avec l'extérieur, une pour son horloge et une pour sa réinitialisation ou reset [27].
- **Les mémoires :** Afin de stocker un programme, des données temporaires ou des données persistantes ; le microcontrôleur dispose de plusieurs types de mémoires, ayant chacune des caractéristiques particulières. Parmi ces types nous trouvons [27]:
 - a. **Mémoires programmes :** Ce dispositif contient les instructions du programme que doit exécuter le microprocesseur. Ce type de mémoire (baptiser : morte), est uniquement accessible en lecture. Sa programmation nécessite une procédure particulière et un matériel adéquat, Il en existe différents types selon leur mode de programmation:

- **La mémoire morte (ROM) :** est une mémoire dont le contenu est conservé même en cas de coupure de courant. Elle contient le programme que va exécuter le processeur. Différentes technologies peuvent être employées : PROM⁷, EPROM⁸, EEPROM⁹, mémoire flash (la plus récente).
 - **PROM:** mémoire morte programmable électriquement une seule fois par le développeur (appelée aussi OTPROM).
 - **EPROM & EEPROM :** se sont de type mémoire morte effaçable électriquement et programmable dont la différence entre eux est que la première nécessite de retirer de l'appareil et la soumettre à un rayonnement ultra-violet pour effacer la mémoire (aussi appelée UVEPROM), par contre la mémoire EEPROM peut être effacée par un simple courant électrique sans qu'il soit nécessaire de la retirer de l'appareil qui la contient.
 - **La mémoire flash (flash-ROM):** c'est une variété de mémoire EEPROM rapide et effaçable par secteur complet, et non par case individuelle. On trouve plusieurs interfaces pour les adresser. C'est ce type de mémoire qui est souvent utilisé dans les microcontrôleurs pour enregistrer le programme qui va gérer le fonctionnement du système technique.
- b. **Mémoires de données :** Ce dispositif permet de mémoriser temporairement les données générées par le microprocesseur pendant les différentes phases du traitement numérique (résultats d'opérations, états des capteurs...). Ces mémoires sont accessibles en écriture et en lecture. On en trouve 2 types:
- **La mémoire vive (RAM) :** elle permet le stockage des données temporairement qui sont nécessaires à l'exécution du programme de gestion du système technique. Elle est dite "Volatile" car ces données ne seront plus disponibles si le système technique est mais hors tension.
 - **La mémoire morte (EEPROM) :** non-volatile (données conservées en cas de coupure de l'alimentation) ayant un temps d'écriture assez élevé (quelques ms) par rapport au temps de lecture qui est assez faible (quelquesns).

⁷ Programmable Read-Only Memory.

⁸ Erasable Programmable Read-Only Memory.

⁹ Electrically Erasable Programmable Read-Only Memory.

- **Les Bus :** Un bus est un ensemble de lignes, transportant des informations codées binaires. Chacune de ces lignes est affectée d'un poids binaire. C'est par l'intermédiaire de ces lignes que s'effectuent les échanges entre les différents éléments du système. On distingue 3 types de bus : Le bus de données, Le bus d'adresse et Le bus de contrôle [32].
- **Les périphériques :** Sont des circuits électroniques intégrés au microcontrôleur capables d'effectuer des tâches spécifiques. On peut mentionner entre autres:
 - ✓ les convertisseurs analogiques-numériques (CAN) (donnent un nombre binaire à partir d'une tension électrique);
 - ✓ les convertisseurs numériques-analogiques (CNA) (effectuent l'opération inverse);
 - ✓ les générateurs de signaux à modulation de largeur d'impulsion (MLI, ou en anglais, PWM pour *Pulse Width Modulation*);
 - ✓ les timers (compteurs d'impulsions d'horloge interne ou d'événements externes);
 - ✓ les chiens de garde (watchdog);
 - ✓ les comparateurs (comparent deux tensions électriques);
 - ✓ les contrôleurs de bus de communication (UART,IIC,SSP,CAN,FlexRay,USB, Ethernet, etc.)[29].
- **Les entrées/sorties :** Permettent au microcontrôleur de communiquer avec le monde extérieur. C'est donc là que vont être connectés les claviers, afficheurs, poussoir, moteurs, relais, etc. que va utiliser l'application [27].

7.1.3 Critères de choix d'un microcontrôleur

Les critères principaux que nous devons retenir pour choisir un microcontrôleur sont les suivants [26] :

- ✓ le ou les circuits de la famille doivent être facilement disponibles sur le marché ;
- ✓ le prix des circuits doit être à la portée ;
- ✓ la programmation de la mémoire morte interne doit être facile;
- ✓ il est important de considérer la vitesse maximale, la quantité de RAM ou de ROM;
- ✓ la consommation d'énergie et les contraintes et le soutien au développement;

7.1.4 Différentes modèles de cartes à microcontrôleur

Cette catégorie de cartes renferme un microcontrôleur complet ; c'est à dire l'association en un seul circuit d'une unité centrale de microprocesseur, de mémoire morte, de mémoire vive, de mémoire EEPROM, d'une interface d'entrée/sortiesérie et de toute la logique nécessaire pour faire fonctionner tout ce la.

Il existe plusieurs types de cartes à microcontrôleur, nous citons à titre d'exemple juste les deux suivants : Arduino et RaspberryPi.

A. Cartes Arduino

Les cartes Arduino possèdent un microcontrôleur facilement programmable ainsi que de nombreuses entrées-sorties. Plusieurs cartes Arduino existent et qui se différencient par la puissance du microcontrôleur ou par la taille et la consommation de la carte.

Le choix du type de la carte Arduino s'effectue en fonction des besoins de votre projet.

La carte Arduino UNO est la carte la plus couramment utilisée qui constitue un bon choix pour les débutants.

L'ensemble des cartes Arduino se programment en C++ à l'aide d'un logiciel de programmation gratuit et open-source fourni par Arduino [30]



Figure 1.11 : Carte Arduino modèle UNO REV3 [31]

Caractéristiques techniques de la carte Arduino UNO REV3

- ✓ version : Rev.3
- ✓ alimentation : via port USB ou 7 à 12 V sur connecteur alim (Tns d'alimttion mini et maxi 6-20V)
- ✓ microprocesseur : ATmega328
- ✓ mémoire flash: 32kB
- ✓ mémoire SRAM: 2 kB
- ✓ mémoire EEPROM: 1kB
- ✓ 14 broches d'E/S dont 6PWM
- ✓ 6 entrées analogiques 10bits
- ✓ intensité par E/S: 40mA
- ✓ cadencement : 16MHz
- ✓ bus série, I2C etSPI
- ✓ gestion des interruptions
- ✓ fiche USB
- ✓ dimensions : 74 x 53 x 15mm
- ✓ version d'origine fabriquée enItalie.

Tableau 1.1 : Caractéristiques techniques de la carte Arduino UNO REV3 [30]



Figure1. 12 : Carte Arduino modèle LEONARDO [31]

Caractéristiques techniques de la carte Arduino LEONARDO

- ✓ Microcontrôleur : ATmega32u4
- ✓ Tension de fonctionnement : 5V
- ✓ Tension d'entrée recommandée : de 7 à 12V
- ✓ Limites de la tension d'entrée : entre 6 et 20V
- ✓ Nombre de pins d'entrées/sorties numériques : 20
- ✓ Connecteurs PWM (Pulse Width Modulation) : 7
- ✓ Connecteurs d'entrées analogiques : 12
- ✓ Courant direct par connecteur d'entrée/sortie : 40mA
- ✓ Courant direct pour le connecteur 3,3V : 50mA
- ✓ Mémoire Flash 32 KO (ATmega32u4) dont 4 KO est utilisée par le bootloader
- ✓ SRAM 2.5 KO (ATmega32u4)
- ✓ EEPROM 1 KO (ATmega32u4)
- ✓ Vitesse d'horloge : 16MHz

Tableau 1.2 : Caractéristiques techniques de la carte Arduino LEONARDO [31]



Figure 1.13 : Carte Arduino modèle Yún [31]

Caractéristiques techniques de la carte Arduino Yún

La carte Arduino Yun est basée sur un ATmega32u4 (comme la Leonardo) et sur un Atheros AR9331 :

Caractéristiques ATmega32u4:

- ✓ *mémoire flash: 32 kB (dont 4kB utilisés pour le BootloaderArduino)*
- ✓ *mémoire SRAM: 2,5 kB*
- ✓ *mémoire EEPROM: 1kB*
- ✓ *20 broches d'E/S dont 6PWM*
- ✓ *6 entrées analogiques 10bits*
- ✓ *intensité par E/S: 40mA*
- ✓ *cadencement : 16MHz*
- ✓ *bus série, I2C etSPI*
- ✓ *gestion des interruptions*
- ✓ *fiche microUSB*

Caractéristiques Atheros AR9331:

- ✓ *architecture : MIPS 400MHz*
- ✓ *Ethernet : IEEE 802.3 10/100Mbits/s*
- ✓ *wifi : IEEE802.11b/g/n*
- ✓ *lecteur de carte : micro SD (nonincluse)*
- ✓ *mémoire RAM : 64 MbDDR2*
- ✓ *mémoire flash : 16Mb*
- ✓ *Dimensions : 72 x 53 x 17mm*

Connectique :

- ✓ *1 portEthernet*
- ✓ *1 portUSB*
- ✓ *1 portmicro-USB*
- ✓ *1 portmicro-SD*

Tableau 1.3 : Caractéristiques techniques de la carte Arduino Yún [31]

B. Cartes RaspberryPi

Le Raspberry Pi est un mini-ordinateur muni d'un processeur ARM, il a été conçu par le créateur de jeux vidéo David Braden.

Au début, le Raspberry Pi ne se déclinait qu'en 2 modèles : le A et le B. Puis s'en est suivie une amélioration de ces 2 modèles, à savoir le A+ et le B+. Depuis février 2015, un 5ème modèle a été commercialisé sous le nom de Raspberry Pi 2 [32].

Maintenant il existe un modèle plus récent qui est le Raspberry Pi 3[33].

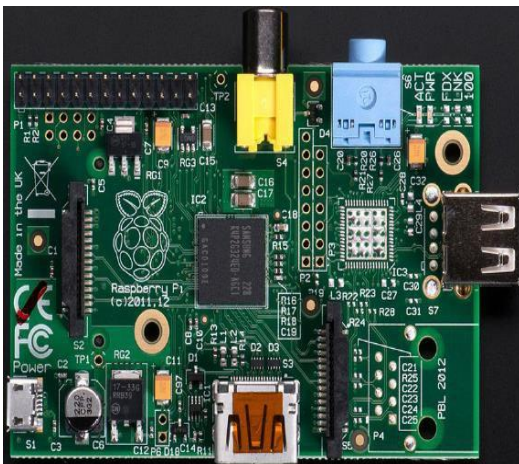


Figure 1.14 : Le Raspberry Pi modèle A [32]

- L'absence de contrôleur et de prise réseau est bien visible.

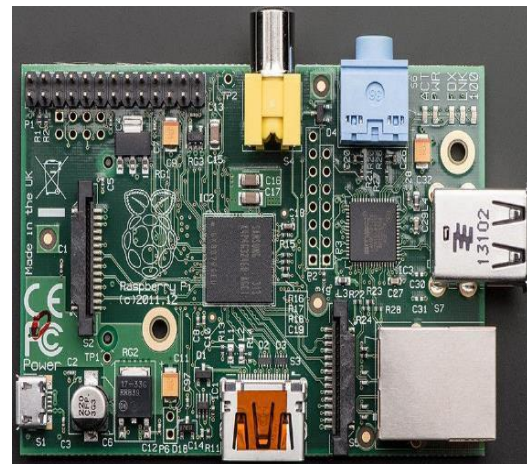


Figure 1.15 : Le Raspberry Pi modèle B [32]

- La présence d'une prise réseau et deux prises USB.

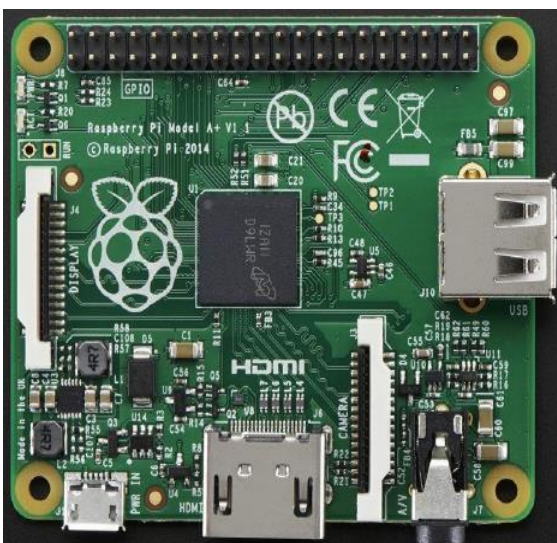


Figure 1.16 : Le Raspberry Pi modèle A+ [32]

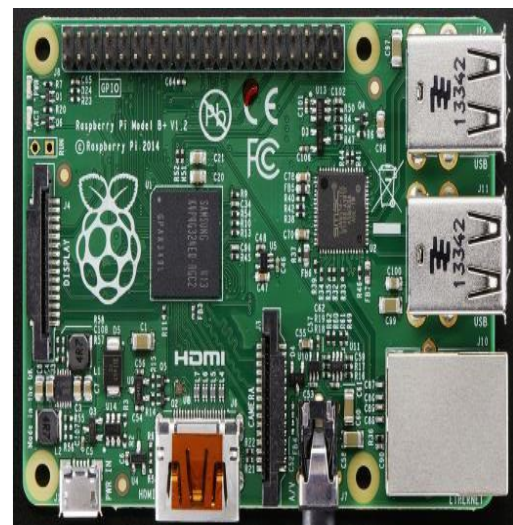


Figure 1.17 : Le Raspberry Pi modèle B+ [32]



Figure 1.18 : *Le Raspberry Pi modèle 2* [32]

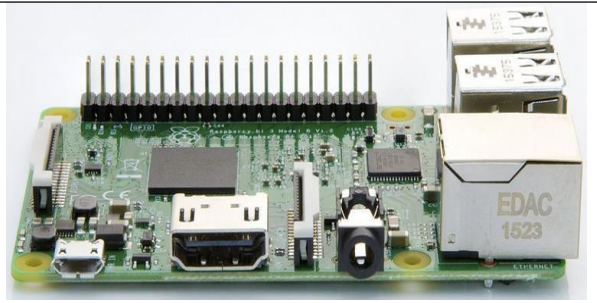


Figure 1.19: *Le Raspberry Pi modèle 3* [33]

- 1.2GHz Quad-Core ARM Cortex-A53
- 802.11 bgn Wireless LAN and Bluetooth 4.1 (Bluetooth Classic and LE)
- 1 Go RAM
- 4 x USB ports

7.2 Les plaquettes d'essais sans soudures (BreadBoard)[38]:

La BreadBoard est une plaquette ou plaque d'essai sans soudures utilisée comme un outil pédagogique indispensable pour découvrir l'électronique. Son principal avantage est de permettre de tester et réaliser des montages rapidement des circuits électroniques sans souder aucuns composants. Il est donc possible de réutiliser les composants.

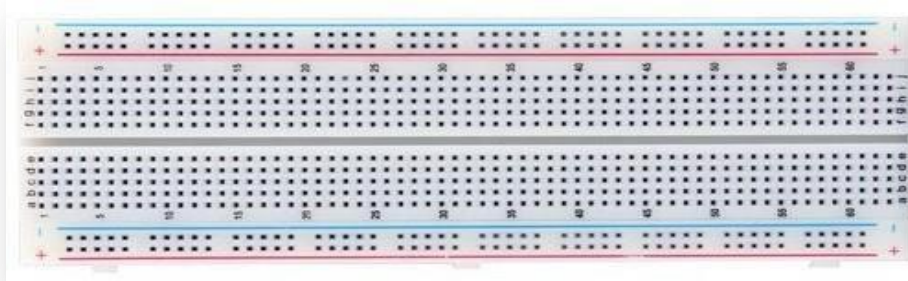


Figure 1.20 : *Une plaquette d'essai sans soudure* [34]

La photo précédente montre un exemple d'une plaquette d'essai. Cette dernière est composée d'une multitude de trous dont certains sont reliés électriquement entre eux, d'un plastique isolant avec des rangées verticales de 5 contacts et 4 lignes horizontales pour l'alimentation.

Les lignes rouges pour relier les composants au (+) et les lignes bleues pour le (-) .

Les composants sont plantés dans les trous de diamètre 0,8mm. Il ne faut pas essayer de planter des composants ayant des connexions de diamètre supérieur.

Le bloc principal permet de placer des composants électroniques typiques comme un circuit intégré, au milieu, sans court-circuiter ses broches.

Il existe différents formats et différentes tailles de BreadBoard ; Voici quelques-unes les plus utilisés :

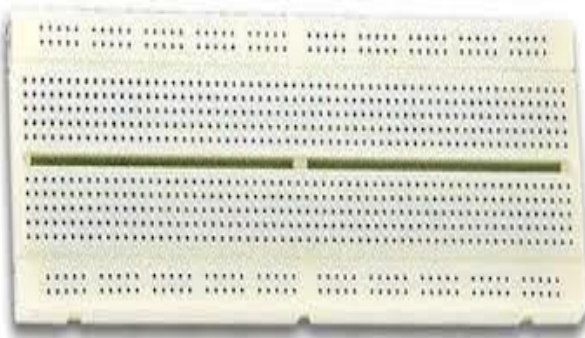
Description	Image
<p>Caractéristiques :</p> <ul style="list-style-type: none"> • module de base (voir fig.): les deux barres de 5 x 64 connecteurs sont séparées par un canal de 7.62mm • barres bus (voir fig.): 4 barres de 25 connecteurs sont réparties sur 4lignes <p>Spécifications :</p> <p>matériau:</p> <ul style="list-style-type: none"> - revêtement extérieur enABS - connexions: alliage d'argent et denickel <p>module de base: 1</p> <p>barres bus: 2</p> <p>840 connections</p> <p>Nombre de CI de 14</p> <p>pattes: 9 borne de connexion:</p> <p>0 dimensions: 168 x 55 x 10mm</p> <p>poids: 95 g</p>	 <p>Figure1.21: BreadBoard SD12N [35]</p>

Tableau1.4: Description technique dela BreadBoard SD12N [35]

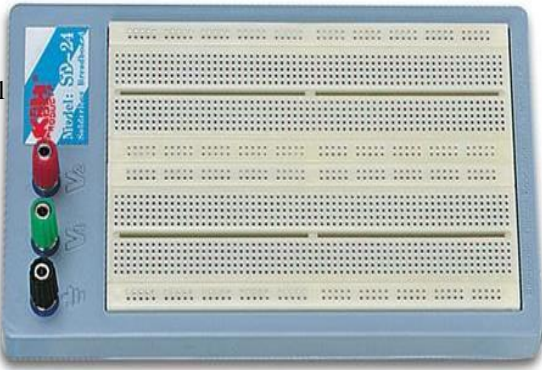
Description	Image
<p>Caractéristiques :</p> <ul style="list-style-type: none"> • module de base (voir fig.): les deux barres de 5 x 64 connecteurs sont séparées par un canal de 7.62mm • barres bus (voir fig.): 4 barres de25 connecteurs sont réparties sur 4 lignes <p>Spécifications :</p> <p>matériau :</p> <ul style="list-style-type: none"> - revêtement extérieur en ABS - connexions : alliage d'argent et de nickel <p>module de base: 2</p> <p>barres bus:4</p> <ul style="list-style-type: none"> • 1680connections • nombre de CI de 14 pattes:18 • bornes de connexion:3 • dimensions: 220 x 127 x 18.5mm • poids: 292g 	 <p>Figure 1.22 : BreadBoard SD35N [35]</p>

Tableau1.5:Description technique dela BreadBoard SD35N [35]

7.3 Les shields

Concernant le coté de cartes à microcontrôleurs il est possible d'ajouter des shields pour le but de faciliter la conception de circuits personnalisés et pouvoir d'ajouter des composants sur la zone de prototypage pour les tester avec la carte Arduino , en plus le shield offre la possibilité de coller uneminiplanche(miniplaquetted'essai)surunezonespécialepourunfonctionnementsanssoudure [36].

Il existe plusieurs shields tel que : shields moteur et servos, shields afficheurs, shields Ethernet, shields wifi ...etc. [37]


caracteristique	image
<p>Livré avec plaquette d'essais sanssoudure.</p> <ul style="list-style-type: none"> - Directement compatible avec le brochage descartes Arduino. - 1 bouton de redémarrageRESET - 1 bouton d'usagegénéral - 2 circuits de LED peut être utiliséedirectement. - Alimentation5V. - plaquette d'essai pour le prototypage peut secoller sur la carte imprimée du shield avec son dos auto- adhésif. Elle possède 170 pointsde branchements dont 2 lignes d'alimentation, et une zone centrale pour CI10 jusqu'à 2 x 10 broches. - Taille 7 x 5.4 x 2cm. - Plaquette d'essai (BreadBoard) de 4.4 x 3.4 x 1cm. - Poids du colis: 33g. 	

Figure 1.23 : Shield de prototypage arduino UNO avec plaquette d »essai [38]

Tableau 1.6 : Shield de prototypage arduino UNO avec plaquette d »essai [38]

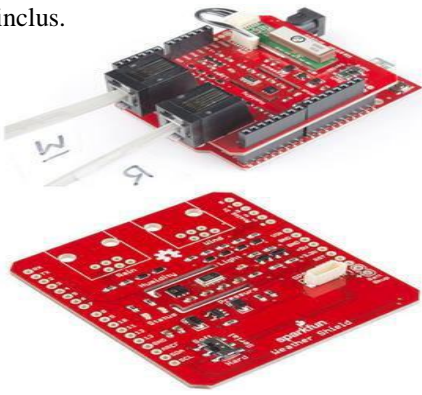
Description	et	I
<p>Shield météo permettant de mesurer l'humidité et la température (HTU21D), la pression atmosphérique (MPL3115A2) et la luminosité (ALS-PT19). Deux emplacements pour connecteurs RJ11 (non inclus) permettent d'ajouter un capteur de pluie et de vent. Une librairie Arduino est disponible pour l'utilisation. L'utilisation de ce shield sur une carte Arduino nécessite la soudure des connecteurs latéraux non inclus.</p> <p>Caractéristiques :</p> <ul style="list-style-type: none"> ✓ Alimentation: 3,3 à 16Vcc ✓ Humidité/température:HTU21D ✓ Pression atmosphérique:MPL3115A2 ✓ Luminosité:ALS-PT19 ✓ Dimensions: 60 x 54mm 		

Figure 1.24 : Shield Météo DEV12081 [39]

Tableau 1.7 : Description et caractéristiques d'un shield de météo DEV12081 [39]

7.4 Les capteurs

7.4.1 Définition d'un capteur

Un capteur est un dispositif qui permet de convertir une grandeur physique observée (température, luminosité, humidité, débit, présence d'objet,...) en une grandeur normée et utilisable, généralement un signal électrique (courant, tension, niveaux logiques, valeur moyenne, fréquence, amplitude, nombre binaire,...), qui peut être interprétée par un dispositif de contrôle commande [40] (voir la figure 25).

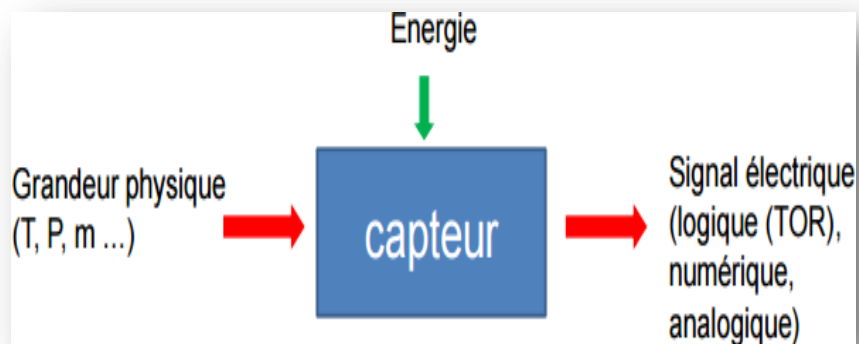


Figure 1.25 : principe de fonctionnement d'un capteur [40]

Les capteurs jouent un rôle indispensable dans le domaine d'IoT, ils peuvent être classés selon deux critères à savoir [40] :

- **Grandeur mesurée** : nous distinguons par exemple un capteur de température, de position, d'humidité, de vitesse, de force, de pression,..etc.
- **Le signal délivré** : nous trouvons par exemple des capteurs logiques (appelés aussi capteurs TOR¹¹), capteurs analogiques et numériques.
- **Capteurs logiques** : La sortie est un état logique que l'on note 1 ou 0. La sortie peut prendre ces deux valeurs exemple: capteurs de fin de course.
- **Capteurs analogiques** : La sortie est une grandeur électrique dont la valeur est une fonction de la grandeur physique mesurée par le capteur.
- **Capteur numérique** : La sortie est une séquence d'états logiques qui, en se suivant, forment un nombre. La sortie peut prendre une infinité de valeurs discrètes. Le signal des capteurs numériques peut être du type : code numérique binaire, train d'impulsions, etc.

¹¹ Tous ou Rien

7.4.2 Caractéristiques des capteurs

Les capteurs sont distingués par les caractéristiques suivantes [41] :

- **Étendue de mesure (Pleine Echelle) :** L'étendue de mesure est la différence entre la limite supérieure et la limite inférieure de la grandeur mesurable par un capteur. Lorsque le capteur fournit une valeur de la grandeur entre 0 et le maximum, ce maximum est appelé « Pleine Echelle».
- **Résolution :** plus petite variation de grandeur mesurable par le capteur.
- **Rapidité :** temps de réaction du capteur. La rapidité est liée à la bande passante.
- **Précision :** Aptitude du capteur à donner une mesure proche de la valeur vraie.
- **Sensibilité :** représente la variation du signal de sortie par rapport à la variation du signal d'entrée.
- **Linéarité :** représente l'écart de sensibilité sur l'étendue de mesure.
- **Finesse :** Elle permet d'estimer l'influence que peut avoir le capteur et de son support ou de ses liaisons sur la grandeur à mesurer. Par exemple, dans le cas d'un capteur de température, une capacité calorifique importante réduit sa finesse.
- **Fréquence de résonance :** Un capteur possède une réponse qui peut dépendre de la fréquence de la grandeur mesurée. Lorsqu'il existe une fréquence à laquelle la réponse est particulièrement élevée, celle-ci est appelée fréquence de résonance.

7.4.3 Critères de choix du capteur [40]

Le choix d'un capteur peut être adopté des points suivants :

- Le type de l'évènement à détecter;
- La nature de cet évènement;
- La grandeur de l'évènement;
- L'environnement de l'évènement;
- L'efficacité, la qualité et la précision du capteur;
- La disponibilité dans le marché.

7.4.4 Exemples des capteurs

En raison du développement technologique rapide qui a connues le monde dans tous les domaines surtout dans le domaine électronique, cela a contribué à l'existence d'une grande variété de modèles pour chaque type de capteur, parfois on les trouve combinés dans un seul capteur. Nous présentons quelques exemples, à savoir :

- **Capteur d'Humidité:**


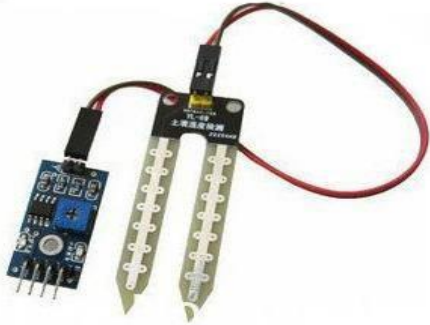
Description	Image
<ul style="list-style-type: none">- Ce capteur d'humidité peut être utilisé pour détecter l'humidité du sol ou comme détecteur de niveau d'eau. Il est idéal pour créer un objet connecté qui indique si une plante asoif.- Un câble au standard Grove est fourni avec ce module (Compatible de l'interfaceGrove, longueur : 50 cm, nombre de pins :4).- Compatible de l'interfaceGrove- Dimension : 20mm x60mm <p>Figure 1.26 : Capteur d'humidité - Grove [37]</p>	
<p>Ceci est un capteur d'humidité simple qui peut être utilisé pour détecter lorsque qu'un sol est en déficit d'eau (niveau haut) et vice versa (niveau bas). Ce module peut être utilisé pour réaliser des systèmes d'arrosage automatique.</p> <p>Le capteur d'hygrométrie est très sensible à l'humidité ambiante. Il est généralement utilisé pour mesurer le taux d'humidité dans lessols.</p> <ul style="list-style-type: none">✓ VCC: .3V-5V✓ GND:GND✓ DO: digital output interface (0ou1)✓ AO: AnalogOutputInterface <p><i>sol</i>[42]</p>	 <p>Figure 1.27 : Capteur hygrométrie du</p>

Tableau1.8 : Description de deux types de capteur d'humidité [37, 42]

• **Capteur d'Eau:**


Description	Image
<p>Ce module va vous permettre de détecter une présence d'eau. Constitué de plusieurs lamelles sur le circuit entre la masse GND et le pin SIG. Une résistance de pull up de 1MΩ permet de protéger le circuit. Ce circuit peut être connecté à une broche analogique de votre carte Arduino UNO.</p> <p>Un câble au standard Grove est fourni avec ce module.</p> <p>Caractéristiques :</p> <ul style="list-style-type: none"> ✓ Compatible de l'interfaceGrove ✓ Faible consommationd'énergie ✓ Dimension : 20mm x60mm ✓ Hautesensibilité 	

Figure 1.28 : Capteur d'Eau- Grove [37]

Tableau 1.9 : Description et caractéristiques d'un capteur d'eau - Grove [37]

• **Capteur de niveau d'Eau:**

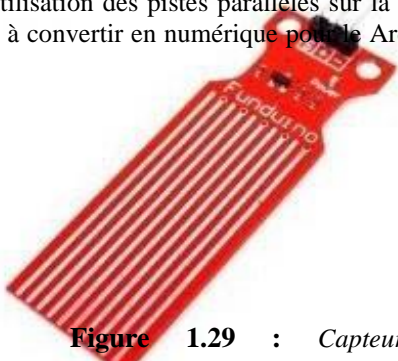
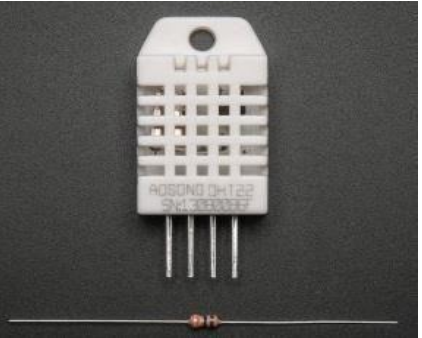
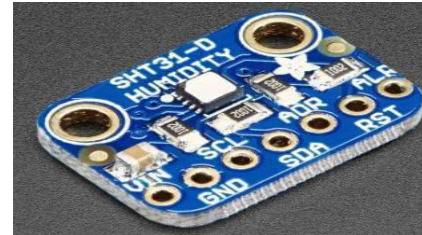
Description	Image
<p>Capteur de niveau d'eau facile à utiliser, bon rapport qualité / prix. L'utilisation en parallèle de plusieurs capteurs ouvre la porte à de grandes installations automatisées. L'utilisation des pistes parallèles sur la Board permettent de détecter le niveau d'eau de manière analogique, facile à convertir en numérique pour le Arduino.</p> <p>Caractéristiques:</p> <ul style="list-style-type: none"> ✓ Tension de fonctionnement:3-5Vdc ✓ Courant en fonctionnement: moins de20mA ✓ Type de capteur: analogique ✓ Dimension de la zone de détection: 40 x16mm ✓ Carte de type: FR4 double faceHASL ✓ Température de fonctionnement: 10-30° ✓ Humidité ambiante: 10-90% (Sanscondensation) ✓ Taille de lacarte: 62mmx20mmx8mm <p><i>niveaud'Eau.</i></p> <p>✓ Connexions:</p> <ul style="list-style-type: none"> ✓ Vcc ✓ GND ✓ Signal 	

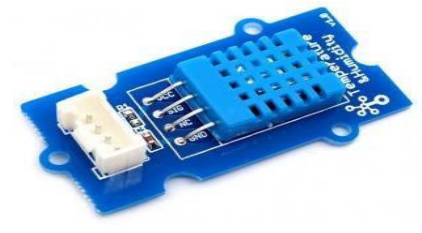
Figure 1.29 : Capteur de (grande version)[42]

Tableau 1.10 : Description et caractéristiques d'un capteur de niveau d'eau [42]

- **Capteur de température et d'humidité** : nous présentons trois modèles de ce type de capteur:

Description et caractéristiques	Image
<p>- La DHT22 est un capteur à bas cout permettant d'acquérir une température et une humidité ambiante d'une manière numérique. Il utilise un capteur d'humidité capacitif et une thermistance pour mesurer la température et l'humidité de l'air et la transmet d'une manière numérique sur un bus série. Les données sont actualisées toutes les 2secondes.</p> <p>- La connexion de ce capteur est très simple, il suffit de relier le premier pin à gauche à l'alimentation (3V à 5V, le pin centrale sur une pin Arduino déclarée en entrée (INPUT) et le pin de droite à la masse (GND).</p> <p>Caractéristiques:</p> <ul style="list-style-type: none"> - Bascout. - Alimentation : 3 to 5V alimentation et data - Consommation : 2.5mA max pendant la conversion - Etendue de mesure humidité : de 0% à 100% avec une précision à 2-5% - Etendue de mesure température : de -40°C à 80°C avec une précision ±0.5°C - Echantillonnage à 0.5Hz (toute les 2seconds) - Dimension : 27mm x 50mm x 12.5mm (1.05" x 2.22" x 0.52") 	 <p>Figure 1.30: Capteur de température et humidité digital – DHT22 [37]</p>

<p>Les capteurs de température et d'humidité Sensiron font partie des plus précis que vous pouvez trouver. De plus, avec une liaison I2C pour une utilisation plus aisée. Le capteur SHT31-D permet une mesure d'humidité relative précise à ±2% et une mesure de température précise à ±0.3°C.</p> <p>Caractéristiques :</p> <ul style="list-style-type: none"> ✓ Utilise les adresses I2C 0x44 ou 0x45 ✓ Dimensions : 12.7mm x 18mm x 2.6mm / 0.5" x 0.7" x 0.1" 	 <p>Figure 1.31: Capteur de température et humidité SENSIRON SHT31-D [37]</p>
--	--

<p>Le module Grove capteur de température et d'humidité utilise un capteur DHT11. La technologie utilisée par le capteur DHT11 garantit une grande fiabilité, une excellente stabilité à long terme et un temps de réponse très rapide.</p> <p>Un câble au standard Grove est fourni avec ce module.</p> <p>Caractéristiques :</p> <ul style="list-style-type: none"> ✓ Compatible de l'interface Grove ✓ Plage de Température : 0°C à 50°C ± 1°C ✓ Plage de d'humidité : 20% à 90%RH ± 20% DL 	 <p>Figure 1.32 : Capteur de température et humidité – GROVE [37]</p>
---	--

- **Tableau 1.11** : Description et caractéristiques de quelques modèles de capteurs de

- **Capteur de pression** : nous présentons les deux modèles suivants:



Description et caractéristiques	Image
<p>- Capteur de pression Freescale à grande sensibilité et linéarité.</p> <p>- Faible hystérésis de pression.</p> <p>- Réponse rapide.</p> <p>- Grande stabilité.</p> <p>- Circuit de compensation nécessaire en cas de variation de température.</p> <p>- Plage de mesure: 15 à 115kPa</p> <p>- Sensibilité: 45,9mV/kPa</p> <p>- Tension: 4,85 à 5,35Vcc</p>	 <p>Figure 1.35 : Capteur de pression Freescale MPX4115A [39]</p>
<p>Module capteur de pression Velleman basé sur un MPXHZ6115A permettant de mesurer la pression atmosphérique. Il communique avec un microcontrôleur type Arduino ou compatible via une liaison analogique. Ce circuit est livré avec un cordon et un connecteur à souder en fonction de l'utilisation.</p> <p>Caractéristiques :</p> <ul style="list-style-type: none"> ✓ Alimentation: 5Vcc ✓ Interface: analogique ✓ Plage de mesure: 0,15 à 1,15bar ✓ Température de travail: 0 à 85°C ✓ Pression maxi: 4bars ✓ Temps de réponse: 20ms ✓ Sensibilité: 45mV/kPa ✓ Offset: 0,2Vcc <p>Dimensions: 22 x 23 x 6mm</p>	 <p>Figure 1.36 : Capteur de pression Velleman MM103 [39]</p>

Tableau 1.13 : Description et caractéristiques de quelques modèles de capteurs de pression [39]

- **Capteur de météo** : nous présentons les deux modèles suivants:



Figure 1.37 : Capteur de Météo [39]

Description et caractéristiques du capteur de météo

Cet ensemble de capteurs météo comprend un anémomètre, un pluviomètre, une girouette ainsi que les accessoires de montage. Il vous permet de mesurer la vitesse du vent, de connaître sa direction et de mesurer la quantité de pluie. Tous les capteurs sont pourvus d'une sortie sur des interrupteurs ILS étanches et ne contiennent pas de carte électronique.

- Le pluviomètre donne une impulsion tous les 0,2794 mm de hauteur de pluie et se vide automatiquement.
- L'anémomètre délivre une impulsion par tour (une vitesse de 2,4 km/h donne une impulsion par seconde).
- La girouette peut donner la direction du vent sur 16 secteurs (grâce à une combinaison de résistances et 8 ILS).

Tous les capteurs sont équipés d'une prise RJ11 et peuvent se raccorder par exemple sur le shield météo **DEVI2081** (regarder le tableau 9 et la figure 22 mentionné auparavant).

Caractéristiques :

- Anémomètre: 1 impulsion = 2,4km/h
- Pluviomètre: 1 impulsion = 0.2794 mm de hauteur
- Girouette: 16 secteurs sur tension analogique (voir fiche technique)
- Dimensions: 724 x 203mm
- Kit à assembler (câblage très simple)

Tableau 1.14 : Description et caractéristiques du capteur de météo et ses composantes [39]

• **Capteur de mouvement**

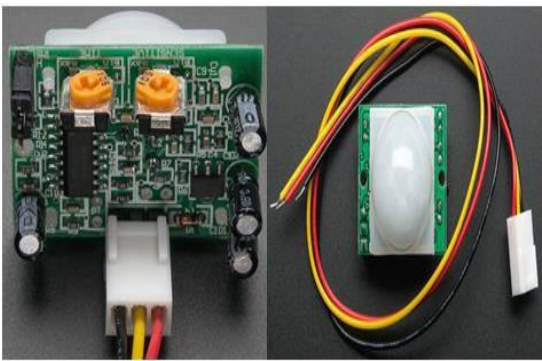

Description et caractéristiques	Image
<p><i>Les détecteur de mouvement PIR sont utilisé pour détecter les mouvements des humains et des animaux dans un rayon de 6 mètres (Ils peuvent détecter également les zombies, mais ce n'est pas garantie). Ce modèle dispose d'un réglage de seuil de détection avant déclenchement de 2 à 4 secondes ainsi que d'un réglage de sensibilité. Un câble avec connecteur de 30 cm est fourni afin de simplifier son câblage.</i></p> <p><i>Il fonctionne sur une alimentation entre 5V et 16V DC, voir même en 3V en soudant juste après le régulateur. Le signal de sortie est à 3,3V à l'état haut. La portée est d'environ 7m sur 120°.</i></p> <p>Caractéristiques :</p> <ul style="list-style-type: none"> ✓ Longueur : 24.03mm / 0.94in ✓ Largeur : 32.34mm / 1.27in ✓ Espacement trous de fixation : 28mm ✓ Diamètre du trou de vis : 2mm ✓ Hauteur (avec objectif): 24.66mm / 0.97in ✓ Poids : 5.87g / 0,2 oz 	 <p>Figure 1.38 : Capteur de mouvement PIR avec réglages [37]</p>
<p>Le module Grove capteur de mouvement va permettre de détecter le moindre mouvement à proximité afin d'activer la sortie SIG à l'état haut. Le champ de détection est paramétrable avec un potentiomètre (non inclus) jusqu'à 6m et le temps de réponse peut être ajusté entre 0.3s et 25s également avec un potentiomètre (non inclus). Il est nécessaire de souder des potentiomètres sur les empreintes pour disposer de ces fonctions. Un câble au standard Grove est fourni avec ce module.</p> <p>Caractéristiques :</p> <ul style="list-style-type: none"> ✓ Compatible de l'interface Grove ✓ Alimentation : 3V - 5V ✓ Angle de détection : 120° ✓ Distance de détection max : 6m ✓ Temps de réponse : de 0.3s à 25s ✓ Dimension : 20 mm x 40mm ✓ Précision : 1,5°C ✓ Dimensions: 22 x 23 x 6 mm 	 <p>Figure 1.39 : Capteur de mouvement Grove [37]</p>

Tableau 1.15 : Description et caractéristiques de quelques modèles du capteur de mouvement [37]

7.5 Les actionneurs

Un actionneur est un objet qui transforme l'énergie qui lui est fournie en un phénomène physique qui fournit un travail, modifie le comportement ou l'état d'un système. Nous pouvons classer les actionneurs selon les trois critères suivants [43]:

- **Energie utilisée :** pneumatique, hydraulique, électrique, mécanique, combustible fossile.
- **Phénomène physique utilisable :** déplacement ou freinage, chaleur ou Froid, lumière, son.
- **Principe mis en œuvre:** pneumatique (les propriétés de compression et de dilatation sont utilisées), hydraulique (la propriété d'incompressibilité du fluide est utilisée).

Le tableau suivant représente un des types d'actionneurs le plus utilisé :


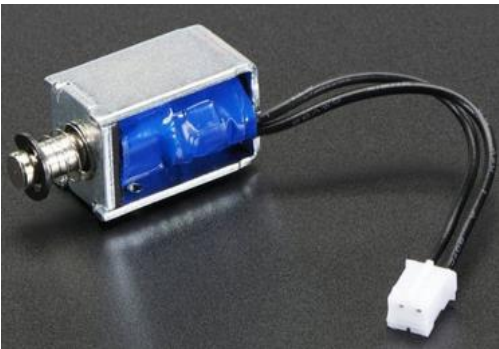
Description et caractéristiques	Image
<p>Cette électrovanne, qui est normalement fermée au repos, fonctionne en 12V grâce à un solénoïde (voir la figure 41) qui actionne l'ouverture du flux. Elle est idéale pour vos projets comme de l'arrosage automatique, un mur d'eau, ou tout autre projet nécessitant un contrôle de flux comme l'eau.</p> <p>Caractéristiques :</p> <ul style="list-style-type: none"> ✓ Tension de travail : 12VDC ✓ Courant max : 450mA ✓ Mode de fonctionnement t: normalement fermé ✓ Taille d'entrée / sortie : G1 / 2 " ✓ Matériaux : nylon / acier inoxydable / Polyoxyméthylène ✓ Type Valve : membrane de la vanne (exploité par servo) Filtre : filtre d'entrée en acier inoxydable ✓ Température de fonctionnement maximum: 120 ° C ✓ plage de pression de fonctionnement: de 0,02 à 0,8 MPa ✓ Durée de vie prévue: 200,000+ 	 <p>Figure 1.40 : Electrovanne G1/2 12V [37]</p>  <p>Figure 1.41 : Mini Solénoïde Push-Pull 5V [42]</p>

Tableau 1.16 : Description et caractéristiques de l'actionneur : Electrovanne G1/2 12V [37]

7.6 Les câbles

Un câble regroupe généralement plusieurs fils conducteurs au sein d'une même gaine qui va les protéger, ces câbles sont utilisés pour relier les différents composants et peut être utilisé pour le transport d'énergie électrique mais aussi pour la transmission de données.

Il existe ainsi plusieurs types et plusieurs modèles de câbles que nous pouvons retrouver sur le marché. Pour la construction de notre projet on va utiliser plusieurs câblages tels que : les câbles d'alimentation pour assurer l'alimentation des composants impliquées ou participées dans notre projet (capteurs, carte Arduino, les Leds, ...). Des câbles antenne pour relier les différentes antennes Wifi, Bluetooth, ..., des câbles USB pour connecter facilement les cartes Arduino à un ordinateur,

ainsi ils permettent de téléverser le code Arduino vers le microcontrôleur. Le tableau suivant présente quelques modèles pour ces genres de câbles :

Description et caractéristiques	Image
<p>Ce câble permet d'alimenter simplement votre kit Arduino (Duemilanove, UNO ou encore MEGA 2560) à l'aide d'un pile 9V type6LR61.</p> <p>Le câble dispose d'un connecteur type clip pile d'un côté et d'une fiche Power jack 2.1mm.</p> <p>Idéal pour donner toute l'autonomie énergétique à votre montage à moindre frais.</p>	 <p>Figure 1.42 : Câble alimentation arduino pour piles 9V [37]</p>
<p>Il existe deux standards pour alimenter une carte en 5V, L'USB et la fiche en DC Jack 2.1mm. Cet adaptateur est donc idéal pour alimenter une carte avec un connecteur DC Jack 2.1mm à partir d'une prise USB en 5V DC.</p> <p>Le câble mesure 1m, il est en 22 AWG et il dispose d'une fiche USB A d'un côté et d'un connecteur DC Jack 2.1mm de l'autre avec le +5V DC relié au centre et la masse à l'extérieur.</p> <p>Caractéristiques :</p> <ul style="list-style-type: none"> ✓ Poids :23g ✓ Longueur : 3.3 feet /1m ✓ Taille de câble : 22AWG 	 <p>Figure 1.43 : Câble USB vers fiche DC JACK 2.1mm – 1m [37]</p>
<p>Ce câble est RG178 et il mesure 15 cm de long (5.9") hors connecteur RP-SMA. Il dispose en une extrémité d'un connecteur RP-SMA montable sur panneau, et souvent utilisé pour les routeurs WIFI et antennes. Les GPS et les connections cellulaires utilisent plutôt le standard SMA.</p>	 <p>Figure 1.44 : Câble antenne d'adaptation RP-SMA vers uFL--u.FL-IPX-IPEX-RF [37]</p>
<p>Ce câble USB est nécessaire pour connecter facilement en USB les cartes Arduino suivantes :</p> <ul style="list-style-type: none"> • ArduinoDuemilanove • ArduinoUNO • Arduino MEGA2560 • ArduinoADK • Mini-USBAdaptateur 	 <p>Figure 1.45 : Câble USB [37]</p>

Tableau 1.17 : Description et caractéristiques de quelques modèles de câbles utilisés [37]

7.7 La communication dans un système d'IoT

7.7.1 Technologies de communication

Certains objets fixes peuvent être connectés par des réseaux filaires, la croissance de l'internet des objets devrait majoritairement être portée par l'utilisation de technologies sans fil et mobiles.

Les technologies de connectivité sans fil sont nombreuses et variées, et l'usage de l'une ou l'autre est souvent avant tout décidé par la portée du réseau envisagé. Certains cas d'usage nécessitent également l'association de technologies sans fil et filaires pour relier les équipements à

Des réseaux privés étendus ou à Internet.

En bref, nous distinguons dans les méthodes de transmission de données entre les objets connectés, deux solutions : filaire et sans fil.

- **Connectivité filaire** : Le réseau filaire est un réseau qui comme son nom l'indique est un réseau que l'on utilise grâce à une connexion avec fil. Ce réseau utilise des câbles Ethernet pour relier des ordinateurs et des périphériques grâce à un routeur ou à un commutateur. C'est grâce aux réseaux filaires que l'on obtient les débits les plus élevés et les plus stables. Nous donnons comme exemples : Ethernet, DSL¹², ADSL¹³, SDSL¹⁴, le câble, Fibre Optique (FTTH, FTTB, FTTC...), CPL¹⁵, etc.[44]
- **Connectivité sans fil** : Les réseaux sans fil permettent de relier des ordinateurs et d'autres appareils informatiques sans avoir à installer de câblage, ce qui représente plus de confort et fait économiser de l'argent au niveau des infrastructures. En revanche, ce genre de réseaux a généralement moins de sécurité car, sans protection efficace, l'introduction d'intrus est probable [45].

Selon le type de couverture, le réseau sans fil peut être classé comme WPAN¹⁶, WLAN¹⁷, WMAN¹⁸ ou WAN¹⁹. Le réseau WPAN²⁰ est commun concernant la technologie **Bluetooth** tandis que les systèmes **WiFi**²¹ établissent généralement des réseaux WLAN. Les technologies à base de **WiMAX**²², à leur tour, permettent d'établir des réseaux WMAN tandis que les réseaux

¹²Digital Subscriber Line

¹³Asymmetric Digital Subscriber Line ¹⁴Symmetric Digital Subscriber Line ¹⁵Courant Porteur en Ligne

¹⁶Wireless Personal Area Network

¹⁷Wireless Metropolitan Area Network

¹⁸Wireless Wide Area Network

¹⁹Wide Area Network

²⁰ Wireless Personal Area Network

²¹ Wireless Fidelity ou 802.11x ou WLAN

²² Worldwide Interoperability for Microwave Access standard ou 802.16x

WAN sont utilisés avec des communications GSM²³, GPRS²⁴ (2G) ou UMTS²⁵ (3G) (voir la figure (43) [45]).



Figure 1.46 : Classement des réseaux sans fil [45]

Il existe de nombreuses options de technologies sans fil utilisées dans l'IoT, notamment: *Bluetooth, Zigbee, Z-wave, 6LowPAN, Thread, Wi-Fi*, outre la technologie cellulaire tel que: *NFC²⁶, Sigfox, Neul, LoRaWAN*.

Nous présentons quelques-unes d'eux [46]:

- **Bluetooth :** la technologie Bluetooth est un acteur incontournable pour les télécoms de courte portée. Elle devrait apporter une contribution majeure pour les produits portables notamment, facilitant encore une fois la connexion à l'IoT même par l'intermédiaire d'un Smartphone. Une nouvelle technologie BLE²⁷, qui répond désormais au nom de Bluetooth Smart, est un protocole important pour les applications IoT. Offrant une portée similaire à celle du Bluetooth, elle a été conçue pour réduire considérablement la consommation électrique.

²³ Global System for Mobile Communication ou en français Groupe Spécial Mobile

²⁴ General Packet Radio Service

²⁵ Universal Mobile Telecommunication System

²⁶ Near Field Communication

²⁷ Bluetooth Low-Energy

- **Versions** : 4.0 , 4.2 (prise en charge des protocoles Internet permettra aux capteurs Bluetooth Smart d'accéder directement à Internet via la connectivité 6LoWPAN²⁸).

- **Norme** : la spécification fondamentale de Bluetooth4.2

- **Fréquence** : 2,4 GHz(ISM).

- **Portée** : 50-150 m(Smart/BLE)

- **Vitesses de transmission** : 1 Mbit/s(Smart/BLE)

➤ **Zigbee** : comme le Bluetooth, ZigBee dispose d'une importante base d'exploitation installée, surtout en milieu industriel. Parmi les profils ZigBee disponibles, ZigBee PRO et ZigBee Remote Control (RF4CE) reposent sur le protocole IEEE802.15.4 ; fonctionnant à 2,4 GHz, cette technologie de réseau sans fil standard de l'industrie cible les applications nécessitant des échanges de données relativement peu fréquents à de faibles vitesses de transmission sur un espace restreint et dans une portée de 100 m (résidence ou bâtiment, parexemple).

- **Norme** : ZigBee 3.0 basé surIEEE802.15.4

- **Fréquence** : 2,4GHz

- **Portée** : 10-100m

- **Vitesses de transmission** : 250Kbit/s

➤ **Thread** : Thread est un tout nouveau protocole de réseau IPv6 basé sur IP, destiné à l'environnement domotique. Basé sur 6LowPAN, et similaire à cette technologie, Lancé à la mi-2014 par le Thread Group (Samsung, Google et ARM) qui ont réalisé une alliance afin de créer le protocole Thread. Le but est de pouvoir connecter les différents équipements de plusieurs constructeurs principalement dans ladomotique.

- **Norme** : Thread, basée sur IEEE802.15.4 et6LowPAN

- **Fréquence** : 2,4 GHz(ISM)

²⁸ IPv6 Low Power Wireless Arena Network

- **Wi-Fi** : la connectivité Wi-Fi s'impose souvent comme le choix évident pour beaucoup de développeurs. Elle ne nécessite pas de longues explications, sauf pour rappeler l'évidence, à savoir la vaste infrastructure existante, le transfert de données rapide et la possibilité de gérer de grandes quantités de données. À l'heure actuelle, la 802.11n s'impose comme la norme Wi-Fi la plus utilisée dans le contexte privé et professionnel. Cette norme offre un débit élevé, de l'ordre de centaines de mégabits par seconde, idéal pour les transferts de fichiers, mais peut-être trop énergivore pour la plupart des applications IoT.

- **Norme** : basée sur 802.11n (actuellement la norme la plus utilisée pour un usage privé).

- **Fréquence** : bandes de 2,4 GHz et 5GHz

- **Portée** : environ 50m

- **Vitesses de transmission** : 600 Mbit/s maximum, mais les vitesses habituelles sont plus proches de 150 Mbit/s, en fonction de la fréquence de canal utilisée et du nombre d'antennes (la dernière norme 802.11-ac devrait permettre des vitesses pouvant atteindre 500 Mbit/s à 1Gbit/s).

➤ **NFC** : NFC (Near Field Communication) est une technologie favorisant des interactions bidirectionnelles simples et sûres entre deux dispositifs électroniques (les Smartphones en particulier), pour permettre aux consommateurs d'effectuer des transactions par paiement sans contact, d'accéder à des contenus numériques et de se connecter à des dispositifs électroniques. Son action principale consiste à étendre les fonctionnalités de la technologie de carte sans contact, pour permettre aux dispositifs de partager des informations à une distance inférieure à 4cm.

- **Norme** :ISO/CEI18000-3

- **Fréquence** : 13,56MHz(ISM)

- **Portée** : 10cm

- **Vitesses de transmission** : 100–420Kbit/s

Le tableau récapitulatif ci-dessous (voir Figure 44) compare les quatre principaux protocoles utilisés dans l'internet des objets. Il démontre que ces typologies présentent différents avantages et inconvénients, et leur utilisation dépend du contexte dans lequel il se trouve.























	WiFi	ZigBee (802.15.4)	Bluetooth	NFC
Network topology	 Star	 Mesh	 Point-to-point	 Point-to-point
Range	 30-100 m	 10-20 m	 10 m	 < 0.1 m
Discovery	 Broadcast	 Broadcast	 Broadcast	 Response to field
Power	 High	 Low	 Classic: Mid  LE/Smart: Low	 Tag: Zero  Reader: Very low
Privacy	 Low	 Mid	 Mid	 High

Figure 1.47 : *Extrait d'un tableau comparatif de protocoles [47]*

7.7.2 Protocoles de communication

Pour répondre à la problématique du nombre grandissant d'objets connectés qui vont faire leur apparition sur la toile dans les années à venir (selon une étude Gartner : c'est près de 26 milliards d'objets connectés qui seront sur Internet d'ici 2020), et pour le but de faire dialoguer et d'automatiser leur comportement du quotidien pour qu'ils fournissent des services ou envoient des informations sur leur état ou leur environnement afin d'en extraire de la valeur [48]. l'IoT s'est doté des standards [49]:

- **HTTP** : signifie Hypertext Transfer Protocol (protocole de transfert hypertexte). Ce protocole définit la communication entre un client (exemple: navigateur) et un serveur sur le World Wide Web(WWW).
- **MQTT**: signifie Message Queuing Telemetry Transport. Il utilise le principe de « Publisher / Subscriber » pour connecter les systèmes entre eux.
- **AMQP** : signifie Advanced Message Queuing Protocol. Un protocole pour les systèmes de messagerie orientés messages(MOM).
- **STOMP** : signifie Simple Text Oriented Messaging Protocol. Est un protocole textuel au-dessus de TCP conçu pour permettre l'interaction avec un middleware orienté messages.

Dans ce qui suit, au chapitre 3 donnerons une explication plus détaillée.

7.8 Les LEDs

Une diode électroluminescente plus connue sous l'appellation Del ou Led (*light-emitting diode*) désigne un dispositif ou composant optoélectronique²⁹ [51] qui émet de la lumière lorsqu'un courant électrique la traverse [52].

Les LEDs sont apparues en 1962 comme des composants électroniques pratiques qui émettaient seulement une lumière rouge de faible intensité. Les versions modernes peuvent être très lumineuses et sont disponibles dans un large spectre de couleurs dans le visible et même dans les spectres ultraviolets et infrarouges. Ils sont principalement utilisés dans divers appareils électroniques comme les montres, les lampes de poche, les téléphones portables, les écrans et bien d'autres parce qu'ils ont une durée de vie plus longue et sont plus économes en énergie que les ampoules ordinaires, ils les ont surtout remplacés, en particulier dans les ménages [52].

Les LEDs ont quelques avantages comme : sa facilité de montage sur un circuit imprimé, sa faible consommation, sa résistance mécanique, sa petite taille, sa longue durée de vie, peuvent fonctionner avec une batterie, etc. [51]



Figure 1.49 : les LEDs [53]

²⁹ Fusion des domaines de l'optique et de l'électronique, dont l'un des objectifs est de pouvoir raccorder circuits intégrés et fibres optiques. Le but ultime de cette discipline reste de réaliser des circuits optiques pour remplacer le

silicium. La résistance

La résistance considérée parmi les composants les plus utilisés dans le monde de l'électronique, est un dipôle qui s'oppose au passage du courant électrique qui la traverse, plus une résistance possède une valeur élevée et plus celle-ci résiste à la circulation du courant. Elle est notamment utilisée pour protéger les diodes électroluminescentes (DEL) qui ne supportent qu'une tension faible. Sur un schéma électronique, son symbole est le suivant [54]:

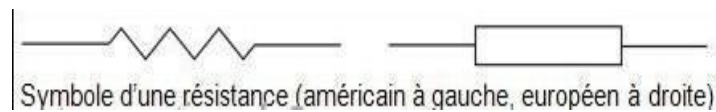


Figure 1.50 : Symbole d'une résistance électrique [54]

La résistance a les caractéristiques suivantes :

- ✓ notée par le symbole : **R**.
- ✓ La résistance **R** d'un conducteur ohmique est le quotient de la tension **U** entre ses bornes par l'intensité du courant **I** qui le traverse : $R = U \text{ (voltage)} / I \text{ (l'ampérage)}$.
- ✓ L'unité de la résistance électrique est le **Ohm** de symbole Ω .
- ✓ L'appareil de mesure de la résistance électrique est l'**ohmmètre**.

Le plus souvent, la résistance se présente avec des bagues de couleurs (anneaux) autour de celle-ci. Chaque couleur correspond à un chiffre. La correspondance entre les chiffres et les couleurs des anneaux constitue ce qu'on appelle le code des couleurs et permet de déterminer la valeur d'une résistance ainsi que sa tolérance. Il existe trois types de résistances : les résistances à 4, 5 et 6 anneaux (couleurs) [55].

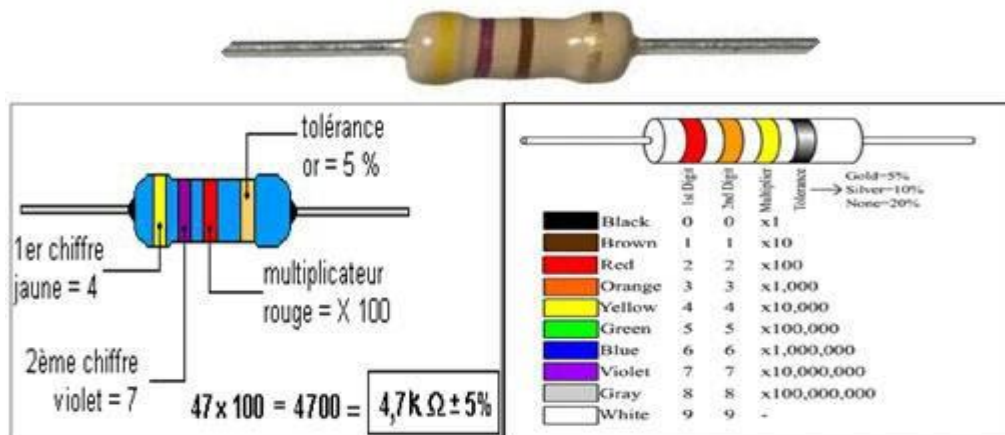


Figure 1.51 : Résistance à 4 anneaux [55]

8 Les plates-formes d'internet des objets

Avec l'explosion de l'internet des objets, de nombreux projets IoT ont vu le jour et on assiste à l'émergence de nombreux développements et solutions IoT. L'internet des objets constitue un ensemble d'objets connectés qui utilisent le réseau comme support pour échanger des données. Ces objets sont divers et peuvent être des cartes (ex : Arduino), des logiciels ou des capteurs.

L'existence de ce vaste réseau nécessite une ou plusieurs plateformes qui servent de support.

De plus, avec des cartes microprocesseurs comme Arduino ou Raspberry Pi devenus accessibles et bon marché, nombreux ont voulu dans le développement de projets IoT. C'est dans cette lancée que de nombreuses entreprises ont créé des plateformes IoT pour les développeurs ou tout simplement des amateurs qui sont intéressés par des projets IoT [56].

Le marché des plateformes IoT est vaste comme un océan! Aujourd'hui, l'écosystème IoT détient plus de 300 entreprises de la plateforme IoT [57], Alors qu'il est devenu d'autant plus nécessaire de connaître correctement les noms des meilleures plates-formes IoT sur le marché. Ce qui est important pour nous, c'est de jeter un œil exhaustive sur chacun d'entre eux, de les explorer et de mettre les mains sur les points et les caractéristiques les plus importants afin de pouvoir choisir la plate-forme correspondant à notre projet et nos moyens matériels modestes.

8.1 Définition d'une plate-forme d'IoT

Une plateforme IoT joue un rôle crucial pour accélérer et réduire le coût de développement de produits et applications IoT. Elle permet non seulement de connecter plusieurs objets et de gérer leur diversité mais aussi de manier des protocoles de communication complexes et variés. Ces plateformes procurent aussi toute l'infrastructure nécessaire pour permettent aux développeurs de mètrent en place des projets IoT facilement et rapidement : serveurs, base de données, management des objets.

Enfin, elles assurent la partie analyse des données en fournissant des bibliothèques d'algorithmes, des requêtes, des tableaux de bords et des outils préconstruits [58].

8.2 Composants d'une plateforme IoT moderne

Nous distinguons 8 composants dans une plateforme IoT moderne (voir la figure 52) [58]:

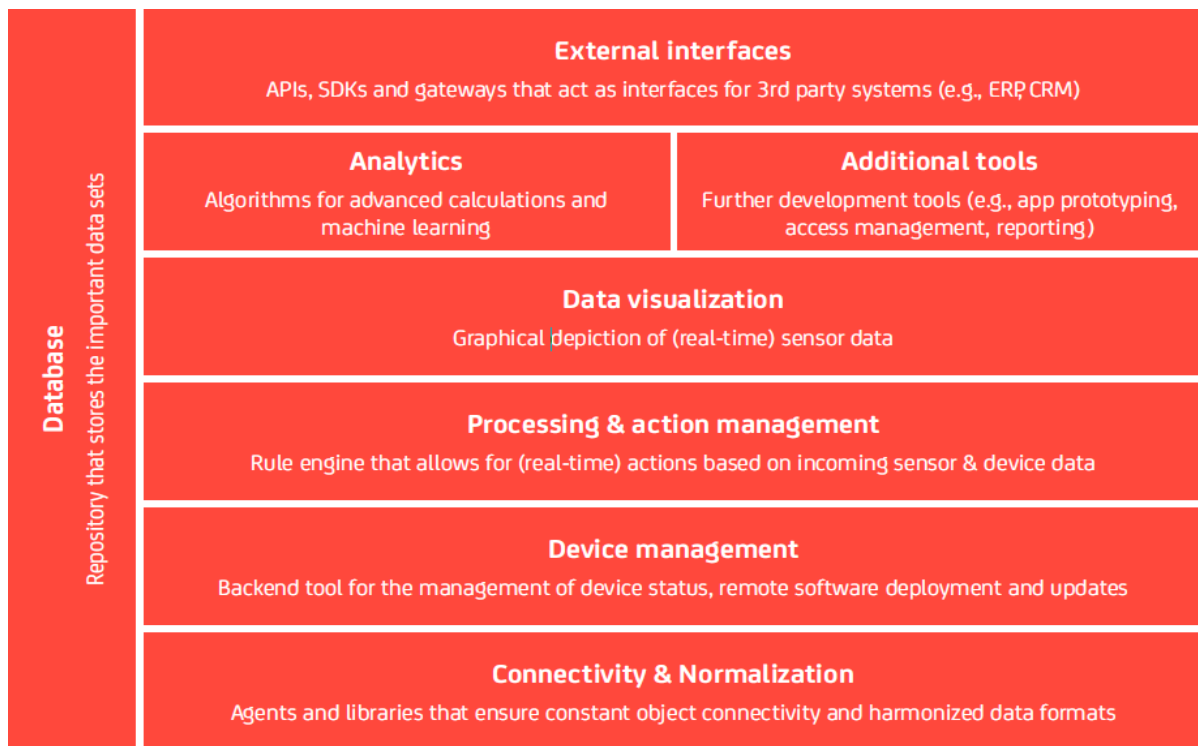


Figure 1.52 : Composants d'une plateforme IoT moderne [58]

- **Connectivité et normalisation** : apporte différents protocoles et différents formats de données en un seul "software" interface assurant ainsi la transmission correcte des données et l'interaction avec tous les dispositifs.
- **Gestion du matériel** : assure que les objets « connectés » s'exécute correctement en gérant la mise à jour des logiciels et applications tournant sur le dispositif ou sur les passerelles.
- **Base de données** : stockage évolutif des données du matériel et prise en compte des exigences pour des base de données basées sur le cloud hybrides (volume de données, diversité, véracité).
- **Gestion des actions et processus** : exploitation des données avec des règles, événements et actions permettant ainsi l'exécution d'actions "smart" actions selon les données des capteurs.
- **Analytiques** : analyse des données allant du Clustering de données aux machines learning et analyse prédictive pour sélectionner les valeurs les plus pertinentes du flux de données.

- **Visualisation** : permettre à l'homme d'apercevoir des patterns et d'observer des tendances et facilement interpréter les données grâce à des diagrammes et des modèles.
- **Outils additionnels** : facilite le prototypage et le test pour les développeurs IoT, fournit aussi des IoT use cases pour créer des applications adaptés à la plateforme.
- **Interfaces externes** : intégration avec des systèmes tiers et d'autres plateformes IoT via des API, SDK et passerelles.

8.3 Différentes plates-formes d'IoT

Comme nous l'avons mentionné auparavant, il existe une multitude de plates-formes d'IoT dans le marché mondial dont les plus célèbres sont[55]:

- Amazon Web Services IoT Platform.
- IBM BLUEMIX.
- IBM WATSON.
- CISCO IoT CLOUD CONNECT.
- Thing Worx.
- Microsoft Azure.
- Google Cloud Platform.
- Thing Speak.
- Sales force IoT Cloud.
- Oracle Integrated Cloud.
- Predix.

Dans ce qui suit, nous présentons un bref aperçu de certaines de ces plates-formes IoT :

- **Amazon Web Services IoT Platform** : Amazon Web Services (AWS) est l'un des géants technologiques les plus dynamiques au monde. AWS figure toujours dans la liste des meilleures plates-formes IoT. Elle fournit un SDK de périphérique, une passerelle de périphérique sécurisée, un registre pour la reconnaissance des périphériques, des ombres de périphérique (une version virtuelle) et un moteur de règles pour évaluer les messages entrants. En termes simples, la plate-forme fournit un lieu pour connecter et gérer les capteurs - sur les voitures, les turbines, les grilles de capteurs et les ampoules par exemple en utilisant le cloud public AWS³⁰ pour stocker, traiter et analyser les données transmises par ces appareils [59].

³⁰ Amazon Web Services

➤ **Microsoft Azure** : Tout comme AWS, Microsoft Azure est un lecteur cloud majeur. Azure, a un système de support intéressant pour les modèles non traditionnels. Azure en font une plate-forme IoT idéale. De plus, il fournit un environnement idéal pour le développement de Linux. Grâce à l'implémentation Hadoop, HDInsight, Apache Storm, Azure Stream Analytics et la plate-forme Azure IoT sont capables de traiter en temps réel les données du périphérique [60].

Fonctionnant aux côtés des services « Microsoft Azure Cloud », Azure IoT Hub propose un moteur de règles, un registre d'identités, la surveillance des informations et l'observation des périphériques. La plate-forme IoT intègre des produits existants tels que IoT Hub, Stream Analytics, des hubs de notification, Power BI et un apprentissage machine pré-emballé pour traiter et analyser de grandes quantités de données de périphériques en temps quasi réel [59].

➤ **Google Cloud Platform** : Google cherche à mettre en place sa plate-forme IoT, qui se situe au-dessus de Google Cloud Platform, à l'exception de ses rivaux qui se concentrent sur l'intelligence. Cela signifie que vous pouvez connecter, gérer et importer des données IoT sur la plateforme Google avant d'exécuter des analyses avancées, comme des requêtes ad hoc avec Google BigQuery, l'apprentissage automatique avec Cloud Machine Learning Engine, visualiser des données dans Google Data Studio ou même déclencher des changements automatiques aux périphériques basés sur des événements en temps réel à l'aide des flux de travail des fonctions cloud[59].

➤ **IBM WATSON** : Avec IBM WATSON, le BIG BLUE a placé son pied dans le monde en pleine explosion de l'Internet des objets. Dans l'ère cognitive, personne ne peut battre IBM dans les capacités cognitives [59]. IBM WATSON en tant que plate-forme IoT est déjà devenu très populaire parmi les développeurs. Watson donne un aperçu approfondi de la grande quantité de données non structurées grâce au traitement du langage naturel avec l'utilisation du moteur de synthèse de la parole et de l'apprentissage automatique. L'avantage des plates-formes IBM Bluemix et IBM Watson IoT est qu'elles permettent aux développeurs de déployer leurs propres applications IoT [60].

Saleforce IoT Platform : Une plate-forme unique qui collecte toutes les données des appareils connectés à l'équipe de Salesforce qui, à son tour, donne un aperçu plus précis des données capturées et suggère les actions des clients en temps réel. C'est une plate-forme IoT en tant que service (PaaS³¹) qui permet aux développeurs de créer des applications

³¹ Plateformes as a Service

supplémentaires, qui peuvent être intégrées à l'équipe Salesforce. Il propose des plateformes cloud IoT pour les solutions CRM. Mais ses services ne se limitent pas aux solutions CRM. Ils ont leur propre magasin d'applications pour télécharger des applications similaires à Google Play Store [60].

➤ **CISCO IoT Cloud Connect** : Selon Cisco, le nombre d'appareils IoT passera à 50 milliards d'ici 2020! Avec une vue d'oiseau, Cisco a largement ciblé l'acquisition d'une part importante des 50 milliards de dispositifs projetés. Cisco IoT Cloud Connect est une plateforme de périphériques IoT basée sur le cloud de Cisco pour fournir des solutions IoT aux opérateurs mobiles. Ses principales caractéristiques sont [60]:

- ✓ Connectivité globale de l'Internet des objets.
- ✓ Automatise entièrement la livraison du service IoT.
- ✓ Connectivité Data & Voice.
- ✓ Contrôle de session IP.
- ✓ Gestion du cycle de vie de la carte SIM.
- ✓ Facturation et rapports personnalisés.
- ✓ Le coût d'entrée est trèsbas.
- ✓ Structure de coûts détaillée pour une rentabilité plus élevée.
- ✓ Permet aux opérateurs mobiles de faire des affaires avec un revenu moyen par connexion inférieur (ARPC).
- ✓ Visibilité en temps réel et mises à jour du réseau à tous lesniveaux.

8.4 Comparaison de quelques plateformes IoT

Le tableau suivant représente une comparaison de quelques plateformes d'IoT selon certains critères tel que : stockage de données, intégration de services, visualisation données, SDK et API, gestion d'évènement et tarification.

Plateforme	Stockage de données	Intégration de services	Visualisation données	DK, API	Gestion d'évènements	Tarification
AWS IoT	Oui	Oui	Avec Elstics Search & kibana	Oui	Oui	Gratuit 1 an (avec des conditions d'utilisation exp nbr apareils, msgs,...) après paiement selon l'utilisation. Exige d'une carte crédit au moins contient 1\$
Microsoft Azure IoT	Oui	Oui	Oui	Oui	Oui, avec IoT hub	Essai gratuit, coût par service Exige d'une carte crédit au moins contient 1\$
IBM IoT WATSON	Oui	Oui	Oui	Oui	Oui, avec IoT hub	Essai 1 mois gratuit, cout par service n'exige pas d'une carte crédit au moins contient 1\$
Carriots	Oui	Oui	Oui	Oui	Oui	2€ par mois par machine
Temboo	Oui	Oui	Oui	Oui	Oui	49\$ par mois par application

Tableau 1.18 : *Comparaison entre quelques plateformes IoT [56]*

9 Les défis d'IoT

Plusieurs obstacles pourraient toutefois ralentir la progression de l'IoT, notamment le - déploiement du protocole IPv6, l'alimentation des capteurs et la définition de normes [6].

➤ **Le déploiement du protocole IPv6:** Nous avons atteint le nombre maximal d'adresses IPv4 en février 2010. Si cela n'a pas eu d'incidence visible pour le grand public, le développement de

l'IoT pourrait s'en trouver ralenti, puisque chacun des milliards de nouveaux capteurs potentiels devra avoir sa propre adresse IP. En outre, le protocole IPv6 facilite la gestion des réseaux grâce à des fonctions de configuration automatiques, et propose des fonctions de sécurité améliorées.

➤ **L'alimentation des capteurs** : Pour que l'IoT puisse dévoiler tout son potentiel, les capteurs devront être autosuffisants. Imaginez qu'il faille changer les piles de milliards d'appareils déployés aux quatre coins de la planète et même dans l'espace. Cela serait évidemment impossible. Nous devons donc trouver un moyen de générer de l'électricité en puisant dans l'environnement, par exemple en utilisant les vibrations, la lumière et les courants d'air.

➤ **Les normes** : D'énormes progrès ont déjà été accomplis dans le domaine des normes, mais le chemin à parcourir est encore long, notamment dans les domaines de la sécurité, de la confidentialité, de l'architecture et des communications. Tout comme l'IEEE, de nombreux organismes s'efforcent de relever ces défis en s'assurant que les paquets IPv6 peuvent être acheminés sur différents types de réseau.

10 Conclusion

Dans ce chapitre, nous avons présenté les différents concepts, composants et les principales caractéristiques liées à l'internet des objets. Nous avons aussi éclairé la majorité des points essentiels qui visent la réalisation de notre projet.

En outre, nous avons montré que cette nouvelle technologie a contribué de manière significative et efficace au développement de la vie humaine, car elle est en fait partie intégrante dans divers domaines.

Nous avons montré – en plus – que cette technologie a contribué d'une façon efficace au développement et à rendre la vie des hommes plus aisée, elle est devenue surtout pour les pays développés un outil dont on ne peut plus s'en passer du fait que les choses qu'on voit et qu'on utilise dans notre vie de tous les jours sont désormais douées d'une intelligence remarquable dont on peut citer: la montre intelligente, le réfrigérateur intelligent, le télé intelligent, la maison intelligente, la voiture intelligente, l'irrigation intelligente, la cité intelligente, l'usine intelligente, etc. Dans la dernière partie de ce chapitre nous avons signalé que notre monde connaît une prolifération étonnante des objets intelligents ce qui a poussé les sociétés des services de cloud à mettre à la disposition des intéressés des plateformes connues sous le nom de plateforme d'internet des objets (plateformes d'IoT) qui contiennent tout ce que l'internaute cible a besoin : prestations, analyses, statistiques, bibliothèques et des remèdes (solutions) qui facilitent aux programmeurs et aux développeurs l'opération de la communication et de sa maîtrise, ainsi qu'elles permettent d'accélérer et de réduire le coût de développement de produits et applications IoT.

CHAPITRE 2 : LA COMMUNICATION DANS IOT

Chapitre 2

La communication dans IoT

1. Introduction

En IoT les objets peuvent communiquer avec les autres (M2M) « machine to machine » ou recueillir des données et envoyé à l'infrastructure de serveur (M2S) « machine to server ». Cette infrastructure de serveur peut partager les données avec autre serveur (S2S) « server to server », ou les objets pour analyser les données. Comme le énorme nombre d'objets augmenté par l'année, trouver un bon protocole pour le réseau est toujours un grand défi.

Dans cette section, on va avoir une vue ensemble sur quelques protocole pour la communication entre les objets et serveur. L'analyse sur leurs points forts et les points faibles pourrait nous donner une solution possible pour notre objet.

2. Les protocoles de transfer des données dans l'IoT

2.1. Hypertext Transfer Protocol, un protocole bien connu

Hypertext Transfer Protocol (HTTP) [RFC 2616] est un des protocoles les plus omniprésents dans l'Internet. Il définit la façon dont le Web navigateurs, proxys Web et les serveurs Web doivent se comporter et d'interagir [62]. HTTP est évolutive, robuste et est un protocole omniprésent en ce moment. Mais les problèmes est l'en-tête du message est lourds.

Une étude de Google [63] montre que les en-têtes des requêtes aujourd'hui varient en taille d'environ 200 octets de plus de 2 Ko. Il n'est pas applicable pour les périphériques qui nécessitent très peu d'interaction. Ils consomment très peu d'énergie et souvent disposer d'une connectivité réseau pauvres. HTTP est trop lourd pour être un bon ajustement pour ces dispositifs. Les frais généraux HTTP ajoute également à l'IoT charges d'exploitation tandis que les coûts actuels de la connectivité sans fil sont déjà cher.

2.2. CoAP - Contrainte Application Protocol, le protocole pour le web des objets

Le Constrained Application Protocol (CoAP) [RFC 7252] est un web protocole spécialisé pour transfert pour une utilisation avec des nœuds contraints et réseaux contraints à l'internet des objets. Comme HTTP, CoAP est basée sur le succès de REST [64](de Representational State Transfer): serveurs fournir des ressources sous une URL, et les clients ont accès à ces ressources en utilisant des méthodes telles que GET, PUT, POST et DELETE.

CoAP est conçue pour interfacer aisément avec HTTP pour l'intégration avec le Web tout en répondant aux besoins spécialisés comme support multicast, très peu de frais généraux, et la simplicité pour environnements contraints. CoAP peut transporter différents types de charges utiles, et peut identifier le type de données sont utilisé.

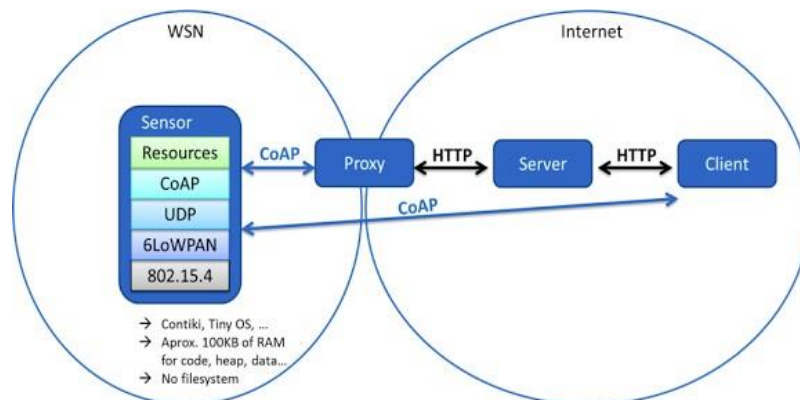


Figure 2.1: *Protocole CoAP activer les services Web en intégrant à l'architecture Web et HTTP[65]*

Sur ce schéma, lorsque le client veut accéder (lecture / écriture) le resouce, il peut utiliser une application envoyer demande via URI de la ressource disponible. Différent entre HTTP et CoAP est un est la base sur TCP, l'autre base sur UDP.

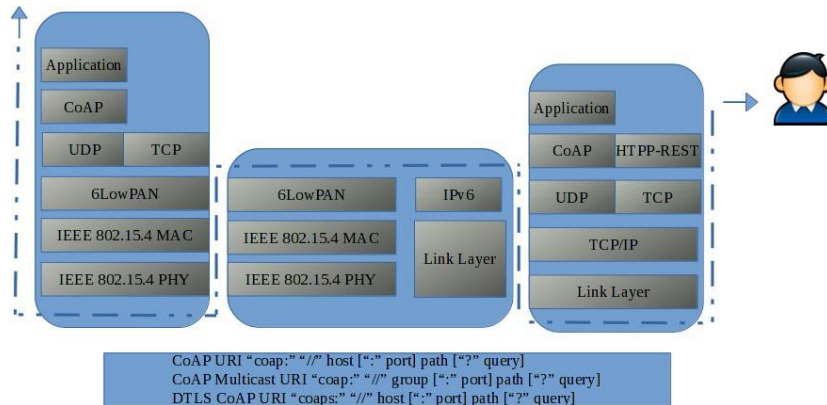


Figure 2.2: CoAP diagramme

Un exemple d'utiliser firefox **Copper** add-on avec l'URI

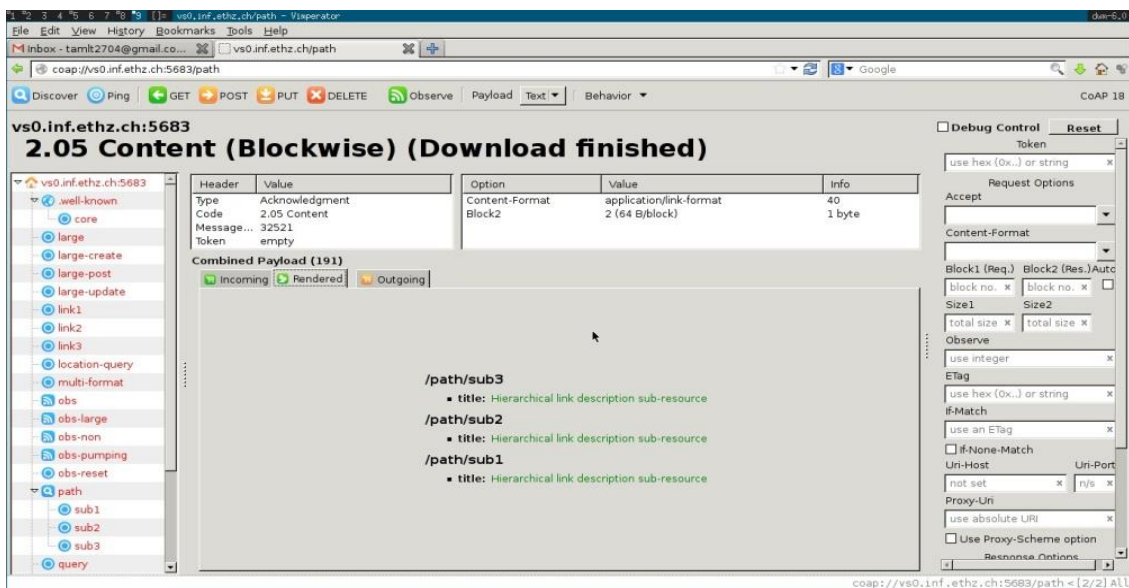


Figure 2.3: Un accès web demande l'utilisation de CoAP exemple à l'aide de ressources à distance via URI

Un autre élément intéressant avec le protocole CoAP est **ressource observant** que signifie lorsque le client CoAP inscrire son intérêt dans une ressource. Il va d'abord obtenir l'état actuel de la ressource (si disponible).

Après que le client recevra les notifications sur les modifications tant que le serveur peut déterminer l'intérêt continu du client dans la ressource.

L'image ci-dessous montre un exemple d'un client CoAP inscrire son intérêt dans une ressource et recevoir les notifications de trois: le premier moment de l'inscription à l'état actuel, puis deux sur les changements à l'état des ressources.

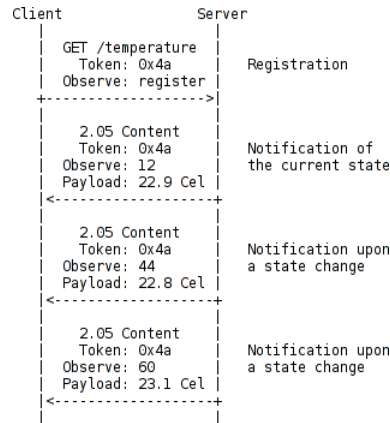


Figure 2.4: *Observer ressources dans CoAP*[66].

En comparaison avec le protocole HTTP, COAP est plus efficace, car il permet de réduire la taille de l'en-tête. Comme RFC 7252, un message vide ne contient que la tête de Octets.

Il est capable (conçu) pour travailler sur des microcontrôleurs avec aussi peu que 10 Kio de RAM et 100 KiB de l'espace de code [RFC 7228]. La sécurité est assurée par la DTLS (de Datagram Transport Layer Security) [RFC 4347] couche pour soutenir l'UDP fiable dans l'architecture abstraite.

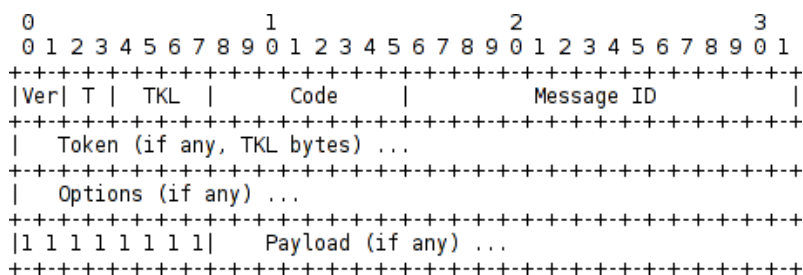


Figure 2.5: *un message CoAP avec l'en-tête seulement 4 octets (en-tête, option, donnée). la source RFC 7252*

CoAP a été largement mis en œuvre jusqu'à présent. Il existe des bibliothèques disponibles pour C, C ++, Java et Python. Nous pouvons également trouver les bibliothèques CoAP dans Systèmes d'exploitation embarqués ouverts avec le soutien 6LoWPAN pour capteurs (par exemple Contiki OS, TinyOS).

En conclusion, nous voyons que le protocole est conçu pour CoAP réseau de contrainte, son schéma est capable pour les objets qui ont des faible performance avec connexion

à faible coût mais ils peut assurer que la base de la sécurité sur DTSL. Le message de transmission fiable pour tenir le client au courant des modifications sur le serveur tant que le registre des clients de la ressource intéressante et serveur peut déterminer laquelle des intérêts du client dans la ressource. Performance, fiabilité et largement mis en œuvre en font un bon candidat pour le protocole de l'IoT.

2.3. Message Queuing Telemetry Transport

Message Queuing Telemetry Transport Protocol (MQTT) est une publication / abonnement protocole de messagerie conçu pour les communications M2M légers. Il a été créé par IBM & Eurotech et remis à Eclipse projet "Paho" M2M (standard OASIS en 2014).

Un problème commun à tous les protocoles IoT est la sécurité. MQTT est basé sur TCP, et utiliser la connexion SSL / TLS. Il fournit également la sécurité dans le cadre à l'aide de nom d'utilisateur et mot de passe et le cryptage de la charge utile.

Une autre caractéristiques de l'avance de MQTT est de garder en vie un message (courtier peut détecter connexion du client), retain (message publié est maintenue sur courtier un nouvel abonné sur le sujet reçoit le "dernier-savoir" bon message).

Même scénario avec CoAP (client communiquer avec le serveur de l'infrastructure), mais sur le scénario MQTT, capteurs (appareils) est de plus d'initiative, car ils peuvent décider quand il est nécessaire de publier des données. Petite tête est également un avantage. Plus le niveau de sécurité QoS flexible et fiable assurer. Il n'est pas difficile de trouver une mise en œuvre de MQTT, il ya certains projets comme HiveMQ, mqtt ou mosquitto.

3. Quel est le meilleur protocole?

Il ya beaucoup de protocole pour l'IoT mais le meilleur protocole pour tous les scénarios ne sont pas existent. Le choix du protocole dépend au scénario. Certains protocoles ont plus de fonctionnalités que les autres, mais n'est pas capables pour le projet ne peut pas etre bon choix.

Un système complexe peut utiliser plusieurs protocoles de profiter de leurs avantages.

Plusieurs standard peut être utilisé pour évaluer protocole que la performance, robuste, scalaire, la sécurité, mais toujours besoin d'équilibre entre des exigences et des performances de l'appareil.

HTTP est robuste, omniprésent mais très lourd pour les appareils à faible rendement.

CoAP est conçu pour le réseau de contrainte avec bouteille, sécurité et facilement applicable à http application, mais le modèle REST lorsque les capteurs sont généralement serveur ne semble pas efficace comme ils ont besoin pour traiter la demande et la réponse. MQTT est mieux en comparaison avec CoAP, car son en-tête est plus petit mais fiable en raison de la QoS. Le cryptage des données assurer que les informations sont en sécurité. Publier / Souscrire (publish/subscribe) modèle permet à chaque utilisation de l'appareil seulement leur propre tâche (publier ou de souscription). Client ne dispose pas d'envoyer la demande à des capteurs pour récupérer les données. Ce système permet d'éviter problème potentiel.

4. Conclusion

Dans ce chapitre, nous avons présenté les différents protocoles de communication au niveau applicative dans IOT. Nous avons aussi éclairé la majorité des points essentiels des principes de fonctionnement.

Une comparaison entre ces protocoles on prise en comptes la projection de notre besoin du projet a réaliser.

Dans ce projet, nous choisissons MQTT grâce-aux performances et on faire une etude detail dans le prochane chapitre.

CHAPITRE 3 LE PROTOCOLE MQTT

Chapitre 3

Le protocol MQTT

1. Introduction

MQTT (Message Queue Telemetry Transport) est un protocole M2M (Machine-to-Machine) open source [67]. C'est un protocole de messagerie basé sur le publish/subscribe à la fois extrêmement simple et léger idéal pour l'Internet des objets. Il a été inventé par Andy Stanford-Clark (IBM) et Arlen Nipper (Arcom, maintenant Cirrus Link) en 1999, lorsque leur cas d'utilisation était de créer un protocole avec les objectifs suivants : [68]

- 1- Simple à mettre en œuvre.
- 2- Fournir une qualité de service de livraison de données.
- 3- Léger et faible consommation de la bande passante.

Par la suite, la version 3.1 a été libérée de droits en 2010, et le protocole est standardisé par l'OASIS en 2014 dans sa **version 3.1.1** : [69]. Il devient un standard ISO (ISO/IEC 20922) en 2016[70].

Aujourd'hui, avec l'essor de l'IOT, MQTT suscite un intérêt grandissant, dont on peut se rendre compte en observant l'évolution du nombre de recherches pour MQTT sur Google Trends (Figure 3-1).

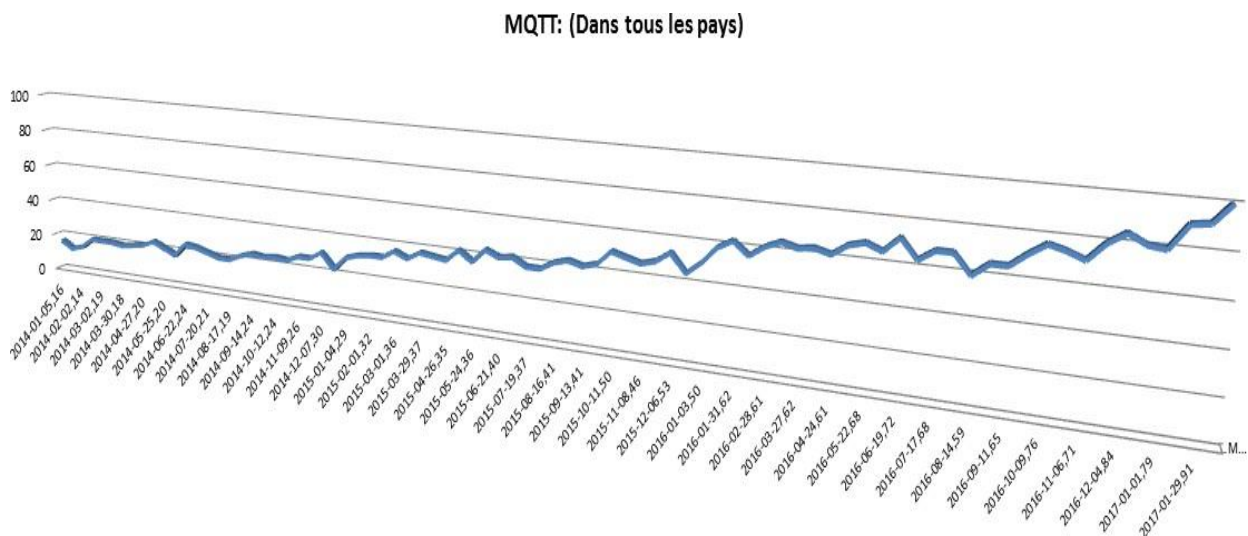


Figure 3-1 Google Trends pour MQTT (18/02/2017)

Les qualités de MQTT font de lui un excellent candidat pour les communications au sein de l'Internet des Objets [71], mais il a également été utilisé avec succès par Facebook pour être au cœur de leur système de messagerie à cause de sa performance [72], comme il est utilisé par amazon web services dans ses passerelles pour permettre la diffusion des données à plusieurs abonnés sur un sujet donné [73].

La norme MQTT décrit le comportement attendu des brokers et clients MQTT, ainsi que les détails du format binaire employé pour les échanges.

La suite de ce chapitre synthétisera l'essentiel de cette norme afin de mieux l'exploité ce protocole dans notre projet. Nous expliquerons en détail le fonctionnement du protocole MQTT. La partie suivante sera consacrée au format binaire défini par la norme pour les paquets MQTT.

2.Mode de fonctionnement

MQTT est un protocole de publication et de souscription de message. Les clients ne communiquent pas directement entre eux, ils publient des messages sur un broker (*appelé aussi courtier*), les messages sont composés d'un contenu et d'un sujet (topic) [74]. Le broker garde en mémoire le dernier message pour chaque sujet. Les clients qui sont intéressés par les messages d'un sujet peuvent les récupérer en se connectant au broker. L'avantage de cette solution est qu'elle permet à plusieurs clients de communiquer même s'ils ne sont jamais connectés en même temps au broker.

Ce patterne consiste à organiser les messages par sujets (topics) et à gérer leur distribution selon le principe d'abonnement. Ainsi, toute application peut publier ses messages sur les topics de son choix, tandis que les applications intéressées par les messages d'un topic donné peuvent s'abonner pour recevoir tous les nouveaux messages publiés sur ce topic.

2.1.Topologies MQTT

Le standard ne définit pas de topologie particulière d'utilisation, la plus simple étant celle avec un seul broker et un certain nombre de clients. D'autres organisations restent évidemment possible, avec en particulier les systèmes à base de bridge dans lesquels plusieurs brokers sont utilisés, chacun étant client des autres brokers, comme sur la Figure.3.2.

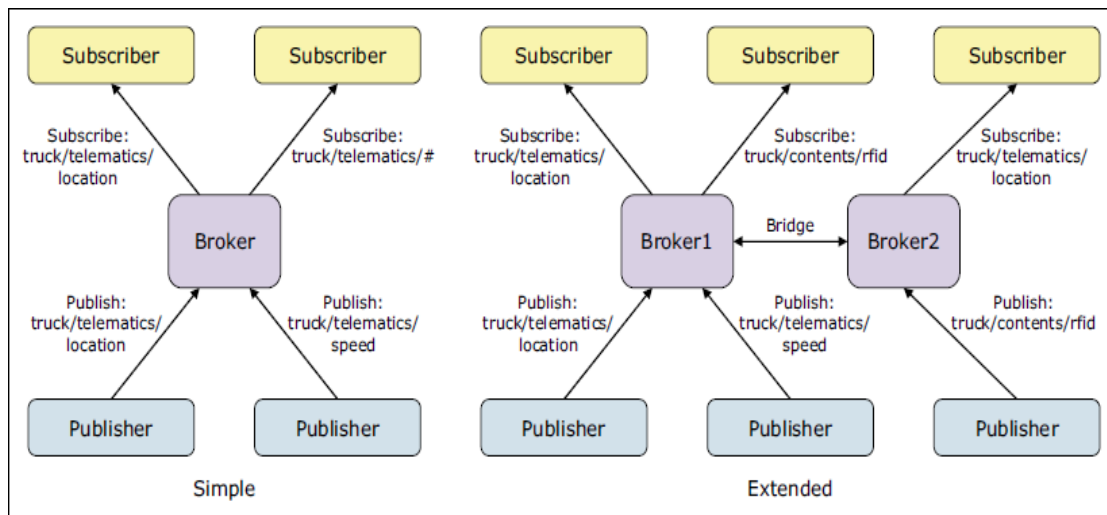


Figure .3-2 – Exemples de topologies MQTT [75]

2.2.Fonctionnement

2.2.1 Connexion et Déconnexion

MQTT se sert de connexions persistantes entre les clients et le broker, et exploite pour cela les protocoles de réseau garantissant un bon niveau de fiabilité comme TCP.

Avant de pouvoir envoyer des commandes de contrôle, un client doit au préalable s'enregistrer auprès du broker, ce qui se fait avec la commande **CONNECT**.

Divers **paramètres** de connexion peuvent alors être échangés, comme les identifiants du client ou encore le mode de persistance souhaité. Le broker doit confirmer au client que son **inscription** a bien été prise en compte, soit indiquer qu'une erreur est survenue en renvoyant un **CONNACK** accompagné d'un code de retour.

Il existe une commande **PINGREQ** permettant de faire savoir au broker que le client est toujours actif, le broker répondra de son côté avec un **PINGRESP** pour indiquer au client que la connexion est toujours active.

Lorsque le client veut se déconnecter, il envoie au préalable une commande **DISCONNECT** au broker pour lui faire part de son intention. Dans le cas contraire, le broker considérera la déconnexion comme anormale et agira en conséquence, [76] en envoyant le message de volonté **WILL** au nom du client à tous les abonnés. Un message que Le client peut spécifier et conservera l'indicateur dans la charge utile de message **CONNECT**.

2.2.2 Abonnements et Publications

Chaque message publié est nécessairement associé à un sujet, ce qui permet sa distribution aux abonnés. Les sujets peuvent être organisés en hiérarchie arborescente, ainsi les abonnements peuvent porter sur des motifs de filtrage que nous détaillerons plus loin.

La gestion des abonnements est très simple et consiste en trois commandes essentielles :

1- SUBSCRIBE Permet à un client de s'abonner à un topic (voir section **2-3**), une fois abonné il recevra par la suite toutes les publications concernant ce sujet. Un abonnement définit également un niveau de qualité de service, dont il sera question plus bas (section **2-4**).

La bonne réception de cette commande est confirmée par le broker par un **SUBACK** portant le même identifiant de paquet.

2- UNSUBSCRIBE Donne la possibilité d'annuler un abonnement, et ainsi ne plus recevoir les publications ultérieures. La bonne réception de cette commande est confirmée par le broker par un **UNSUBACK** portant le même identifiant de paquet.

3- PUBLISH Initié par un client, permet de publier un message qui sera transmis par le broker aux abonnés éventuels. La même commande sera envoyée par le broker aux abonnés pour délivrer le message.

Si la qualité de service requise est supérieure à zéro, des messages seront échangés pour confirmer la prise en charge de la publication.

2.3 Topic et motifs de filtrage

Un sujet est une chaîne UTF-8, qui est utilisé par le broker pour filtrer les messages pour chaque client connecté. Un sujet est constitué d'un ou plusieurs niveaux de sujet. Chaque niveau de sujet est séparé par une barre oblique.

Voici quelques exemples de sujets:

sport/tennis/player1/score/wimbledon myhome / RdC / living / température

5ff4a2ce-e485-40f4-826c-b1a5d81be9b6 /Allemagne / Bavière / voiture /

2382340923453/LNj200

A noter que chaque sujet doit avoir au moins 1 caractères pour être valide et il peut également contenir des espaces. Aussi un sujet est sensible à la casse, ce qui rend **myhome / température** et **MyHome / Température** deux sujets individuels. En outre, la barre oblique seule est un sujet valable.

Un client peut souscrire à plus de sujets à la fois comme illustré dans la figure qui suit:

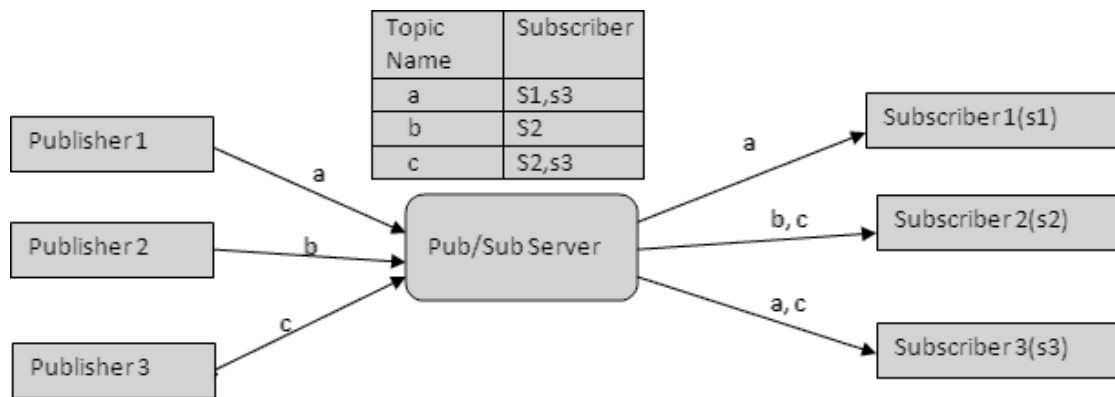


Figure 3-3: *Souscription au topic*

Afin de proposer un système de filtrage efficace sur les sujets, il est possible de définir une arborescence à l'aide du séparateur /.

Deux jokers sont réservés pour représenter un et plusieurs niveaux d'arborescence:

- + représente un niveau d'arborescence, ainsi T1/T2/T3 peut être mis en correspondance avec divers filtres tels que T1/T2/+, T1/+/T3 ou T1/+/+.
- # représente autant de niveaux que possible, et ne peut être utilisé qu'à la fin d'un motif de filtrage ; ainsi T1/# permettra de filtrer tous les sujets dont le premier niveau est T1, et # permet de recevoir tous les sujets publiés par le broker (A l'exception des sujets spéciaux commençant par\$).

2.4 Qualité de service [77]

Trois niveaux de qualité de service (QoS) sont définis pour la publication des messages:

QoS 0: Livraison *une fois maximum*.

Les messages sont envoyés en fonction des capacités maximum du réseau TCP/IP sous-jacent. Aucune réponse n'est attendue. Aucune sémantique de relance n'est définie dans le protocole. Par conséquent, le message ne parvient pas du tout ou une seule fois au broker de destination. Le niveau QoS 0 est également connu comme *fire and forget*.

QoS 1: Livraison *au moins une fois*.

L'arrivée d'un message QoS 1 au broker est reconnue. En cas d'échec identifiable de la liaison ou de l'unité d'envoi, ou bien après une certaine période de temps sans réception du message d'accusé de réception, l'expéditeur envoie à nouveau une copie du message. Par conséquent, le message est sûr d'arriver, mais il peut le faire plusieurs fois.

QoS 2: Livraison *exactement une fois*.

Pour le niveau QoS 2, d'autres flux de protocoles sont utilisés au-delà de QoS 1 pour que des messages ne soient pas envoyés en double à l'application de réception. Il s'agit du niveau de service le plus élevé qui sert lorsque des messages en double ne sont pas appropriés. Il existe évidemment des conséquences en matière de trafic réseau, mais cet impact est souvent acceptable sachant l'importance du contenu du message.

La qualité de service peut être sélectionnée message par message, ce qui permet la publication des messages d'importance mineure avec le niveau QoS 0 et la distribution des messages plus importants avec QoS 2.

Ces niveaux sont mis en œuvre par des échanges supplémentaires entre l'expéditeur et le récepteur, et plus la qualité demandée est élevée, plus il faudra d'échanges pour valider une publication comme illustré dans la figure.

Pour tous les niveaux supérieurs à zéro, un identifiant est associé au message pour permettre son suivi, MQTT prévoyant la possibilité d'avoir au plus 65535 messages en attente (l'identifiant de message tenant sur 16 bits)

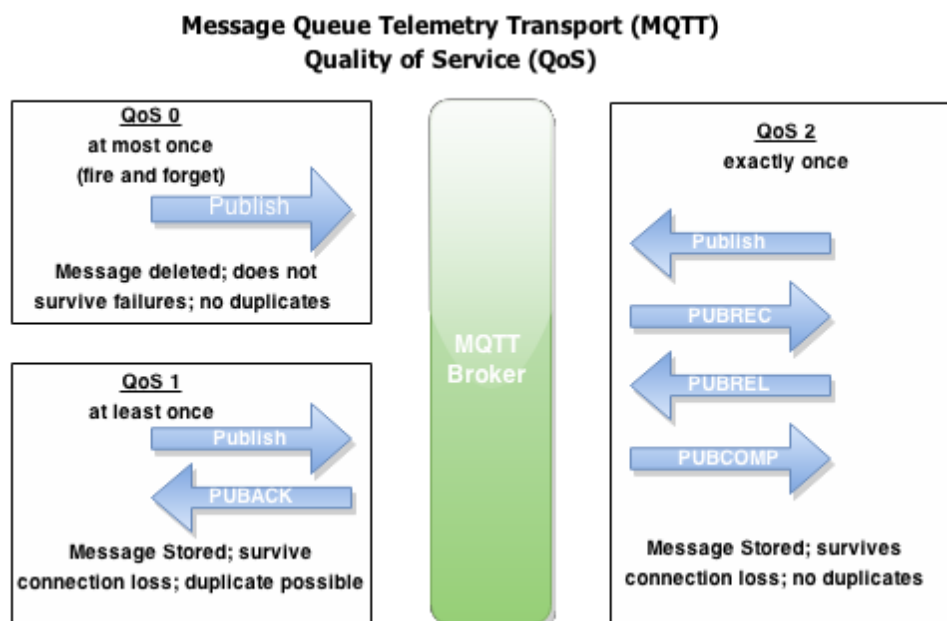


Figure 3-4: Niveaux de *Qualité de Service MQTT* [78]

2.5 Dernière volonté et testament (Last Will & Testament)[79]

Permet de notifier une déconnexion anormale, ils ont défini lors de la connexion.

❖ Dernière volonté (LastWill):

Dans le protocole MQTT parce qu'il n'y a pas de file d'attente le message n'est pas mis en cache. Pour éviter ce problème, les utilisateurs peuvent définir une (*retenue dans le drapeau*) lors de la publication. Ce drapeau invite le broker pour stocker le message et à l'offrir aux abonnés futurs dès leur inscription comme dernière publication connue pour un sujet donné.

❖ Testament:

Est une caractéristique importante de MQTT. Un client indique au broker au cours de la configuration de la connexion que: «En cas de ma mort, s'il vous plaît envoyer le message suivant pour ce sujet à mes abonnés». Si le broker détecte alors que ce client est plus en vie, il le fait comme il a été dit et publie le message pour les abonnés pour le sujet désiré.

2.6 Sécurité [80]

Trois concepts sont fondamentaux pour la sécurité MQTT : identification, authentification et autorisation. .

- **L'identification:** consiste à nommer le broker et le client auquel on donne des droits d'accès.
- **L'authentification:** cherche à prouver mutuellement l'identité du client et du broker.
- **L'autorisation:** consiste à gérer les droits du client.

Le broker MQTT s'identifie auprès du client avec son adresse IP et son certificat numérique.

Le client MQTT utilise le protocole SSL pour authentifier le certificat envoyé par le broker.

Un client authentifie un broker à l'aide du protocole SSL.

Un broker MQTT authentifie un client à l'aide du protocole SSL, d'un mot de passe, ou des deux.

L'autorisation ne fait pas partie du protocole MQTT. Elle est fournie par les brokersMQTT. Ce qui est autorisé ou non dépend de ce que fait le broker.

3 Description informelle du protocole ⁽¹⁾

Dans cette section, nous définissons le format binaire des paquets spécifié par la norme MQTT.

3.1 Représentations de données

3.1.1 Bits

Les bits dans un octet sont étiquetés de 7 à 0. Le bit numéro 7 est le bit le plus important, le bit le moins significatif est affecté au bit 0.

3.1.2 Valeurs de données entières

Les valeurs de données entières sont 16 bits: l'octet d'ordre supérieur précède l'octet d'ordre inférieur. Cela signifie qu'un mot de 16 bits est présenté sur le réseau comme octet le plus significatif (**MSB**), suivi de l'octet le moins important (**LSB**).

3.1.3 chaînes codées enUTF-8

Les chaines de caractères doivent toutes être encodées en utf8, et préfixées par leur longueur sur deux octets, ces chaines sont donc limitées à une longueur de

$$2^{16} - 1 = 65535 \text{ octets.}$$

3.2 Structure d'un paquet de contrôle MQTT

Le protocole MQTT fonctionne en échangeant une série de paquets de contrôle MQTT d'une manière définie. Cette section décrit le format de ces paquets.

Un paquet de contrôle MQTT comprend jusqu'à trois parties, toujours dans l'ordre suivant: En-tête fixe, En-tête variable, Charge utile,

3.2.1 Entête fixe

Tous les packets MQTT commencent par un format d'entête fixe, contenant le code de commande MQTT (sur 4 bits), un drapeau de 4 bits et la remaining length.

Bit	7	6	5	4	3	2	1	0
byte 1	MQTT Control Packet type				Flags specific to each MQTT Control Packet type			
byte 2	Remaining Length							

Figure 3-5: Entête fixe du protocole MQTT.

(1) Tous les tableaux de cette section sont extraits du ref -46.

3.2.1.1 Control Packet type:

Name	Value	Direction of flow	Description
Reserved	0	Forbidden	Reserved
CONNECT	1	Client to Server	Client request to connect to Server
CONNACK	2	Server to Client	Connect acknowledgment
PUBLISH	3	Client to Server/Server to Client	Publish message
PUBACK	4	Client to Server/Server to Client	Publish acknowledgment
PUBREC	5	Client to Server/Server to Client	Publish received (assured delivery part 1)
PUBREL	6	Client to Server/Server to Client	Publish release (assured delivery part 2)
PUBCOMP	7	Client to Server/Server to Client	Publish complete (assured delivery part 3)
SUBSCRIBE	8	Client to Server	Client subscribe request
SUBACK	9	Server to Client	Subscribe acknowledgment
UNSUBSCRIBE	10	Client to Server	Unsubscribe request
UNSUBACK	11	Server to Client	Unsubscribe acknowledgment
PINGREQ	12	Client to Server	PING request
PINGRESP	13	Server to Client	PING response
DISCONNECT	14	Client to Server	Client is disconnecting
Reserved	15	Forbidden	Reserved

Tableau 3.1 – *Type des paquets MQTT*

3.2.1.2 Remaining length: Une particularité du champ **remaining length** est qu'il est de longueur variable, et peut utiliser de 1 à 4 octets en fonction de la taille encodée c'est-à-dire : La taille maximale d'un paquet MQTT est donc de 256 Mo.

3.2.1.3 Flag: Les bits restants [3-0] de l'octet 1 de l'en-tête fixe contiennent des drapeaux spécifiques à chaque type de paquet de contrôle MQTT. Lorsqu'un bit de drapeau est marqué comme «Réservé» , il est réservé pour une utilisation future.

Control Packet	Fixed header flags	Bit 3	Bit 2	Bit 1	Bit 0
CONNECT	Reserved	0	0	0	0
CONNACK	Reserved	0	0	0	0
PUBLISH	Used in MQTT 3.1.1	DUP ¹	QoS ²	QoS ²	RETAIN ³
PUBACK	Reserved	0	0	0	0
PUBREC	Reserved	0	0	0	0
PUBREL	Reserved	0	0	1	0
PUBCOMP	Reserved	0	0	0	0
SUBSCRIBE	Reserved	0	0	1	0
SUBACK	Reserved	0	0	0	0
UNSUBSCRIBE	Reserved	0	0	1	0
UNSUBACK	Reserved	0	0	0	0
PINGREQ	Reserved	0	0	0	0
PINGRESP	Reserved	0	0	0	0
DISCONNECT	Reserved	0	0	0	0

Tableau 3-2: Description du drapeau dans l'entête fixe du protocole MQTT

Voir la section 3.2.4 pour une description des drapeaux DUP, QoS et RETAIN dans le paquetde contrôle PUBLISH.

3.2.2 CONNECT

L'entête de **CONNECT** dans la version 3.1.1 se présente sous la forme suivante :

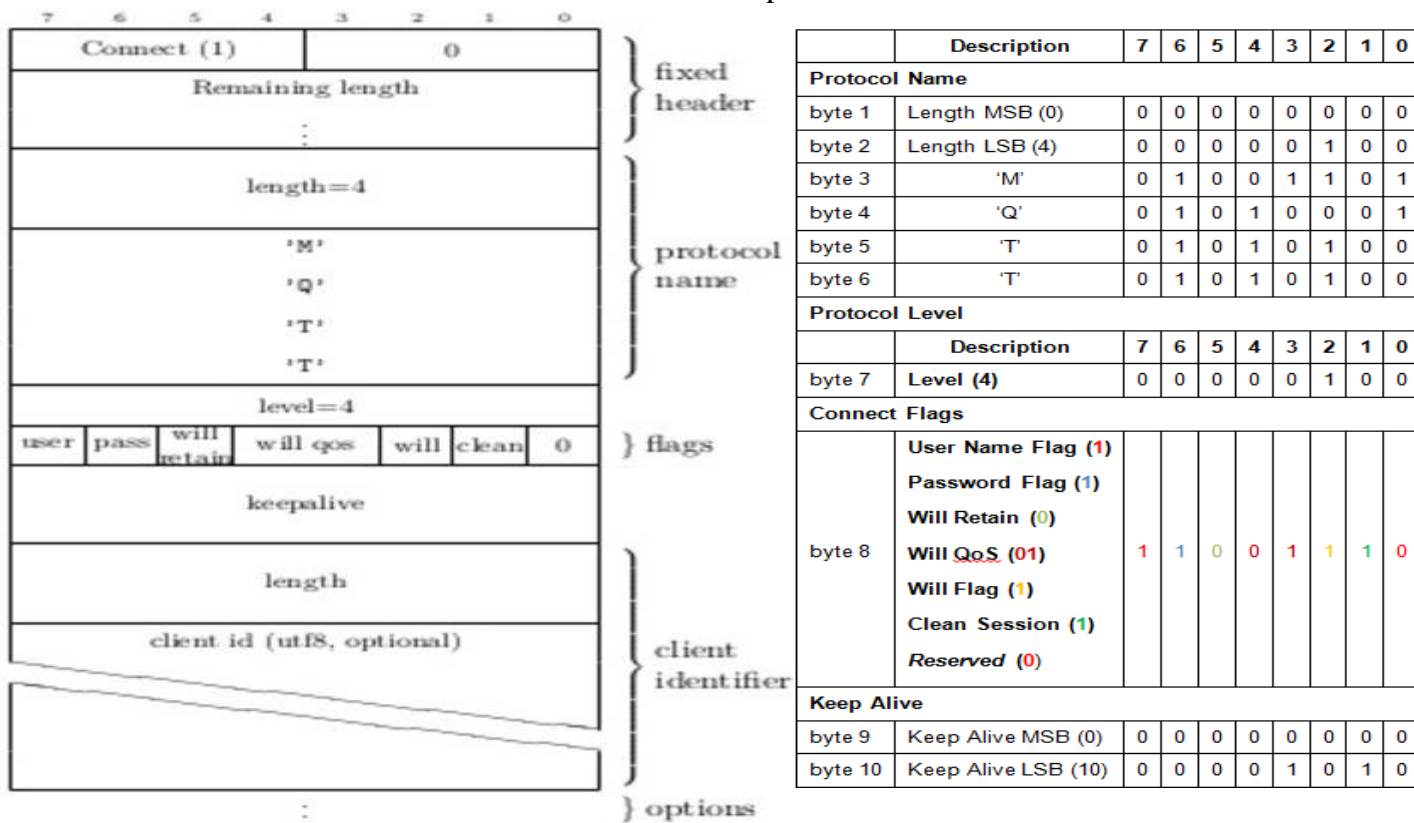


Figure .3-6: Format d'un paquet CONNECT

Tout d'abord le nom du protocole (préfixé par sa taille), puis le niveau du protocole, des drapeaux indiquant la présence ou non de paramètres optionnels ainsi que la persistance de la session (clean indique que la session est non persistante, et constitue en même temps une directive pour effacer une éventuelle session existante), et enfin la valeur du keepalive

3.2.2.1 Level

La valeur non signée de 8 bits qui représente **le niveau de révision du protocole** utilisé par le Client. La valeur du champ Level de protocole pour la version 3.1.1 du protocole est **4 (0x04)**.broker DOIT répondre au paquet CONNECT avec un code de retour CONNACK 0x01 (niveau de protocole inacceptable), puis déconnecter le client si le niveau de protocole n'est pas pris en charge par le broker

3.2.2.2 Connect Flags

L'octet Connect Flags contient un certain nombre de paramètres spécifiant le comportement de la connexion MQTT. Il indique également la présence ou l'absence de champs dans la charge utile.

- Le broker doit valider que le drapeau **Reserved** dans le paquet de Control **CONNECT**est mis à **zéro** et déconnecter le client s'il n'est pas zéro.

3.2.2.2.1 Nom de l'utilisateur Position:bit 7 de l'octet Connect Flags.

- Si le drapeau de nom d'utilisateur est défini sur 0, un nom d'utilisateur NE DOIT PAS être présent dans la charge utile.
- Si le drapeau de nom d'utilisateur est défini sur 1, un nom d'utilisateur DOIT être présent dans la charge utile.

3.2.2.2.2 Indicateur de mot de passe Position: bit 6 de l'octet Connect Flags.

- Si le drapeau du mot de passe est défini sur 0, un mot de passe NE DOIT PAS être présent dans la charge utile.
- Si le drapeau de mot de passe est défini sur 1, un mot de passe DOIT être présent dans la charge utile.
- Si le drapeau de nom d'utilisateur est défini sur 0, l'indicateur de mot de passe DOIT être réglé sur 0.

3.2.2.2.3 Will Retain

Position: bit 5 de l'octet Connect Flags.

Ce bit spécifie si le message de volonté doit être conservé lorsqu'il est publié.

- Si le drapeau Will est défini sur 0, alors le drapeau Will Retain doit être réglé sur 0.
- Si le drapeau de volonté WILL est défini sur 1:
 - o Si Will Retain est défini sur 0, le broker DOIT publier le message Will comme un message

non retenu.

- Si Will Retain est défini sur 1, le broker DOIT publier le message Will comme message conservé.

3.2.2.2.4 Will QoS

Position: bits 4 et 3 de l'octet Connect Flags

Ces deux bits spécifient le niveau QoS à utiliser lors de la publication du message Will. Si le drapeau de Will est défini sur 0, alors la QoS de will DOIT être réglée sur 0 (0x00).

Si le drapeau Will est défini sur 1, la valeur de Will QoS peut être 0 (0x00), 1 (0x01) ou 2 (0x02). Il NE DOIT PAS être 3.

3.2.2.2.5 Will (volonté)

Position: bit 2 de l'octet Connect Flags.

Si le drapeau Will est défini sur 1, cela indique que, si la demande de connexion est acceptée, un message de volonté DOIT être stocké sur le broker et associé à la connexion réseau. Le message de volonté DOIT être publié lorsque la connexion réseau est fermée par la suite, sauf si le message de volonté a été supprimé par le broker lors de la réception d'un paquet DISCONNECT.

Les situations dans lesquelles le message de volonté est publié comprennent, sans s'y limiter:

- Une erreur d'E / S ou une panne de réseau détectée par le broker.
- Le Client ne parvient pas à communiquer dans le temps de KeepAlive.
- Le client ferme la connexion réseau sans d'abord envoyer un paquet DISCONNECT.
- Le broker ferme la connexion réseau en raison d'une erreur de protocole.
- Si le drapeau de volonté est défini sur 1, les champs Will QoS et Keep Retain dans les Flags Connect seront utilisés par le broker et les champs Will Topic et Will Message DOIVENT être présents dans la charge utile.

Le message de volonté DOIT être supprimé de l'état de session stocké dans le broker une fois qu'il a été publié ou le broker a reçu un paquet DISCONNECT du client

- Si le drapeau de volonté est défini sur 0, les champs Will QoS et Keep Retain dans les drapeaux de connexion DOIVENT être mis à zéro et les champs Will Topic et Will Message NE DOIVENT PAS être présents dans la charge utile.
- Si le drapeau de volonté est défini sur 0, un message de volonté NE DOIT PAS être publié lorsque cette connexion réseau se termine.
- Le broker DEVRAIT publier Will Messages rapidement. Dans le cas d'un arrêt ou d'une panne du broker, le broker PEUT différer la publication des messages de volonté jusqu'à un redémarrage ultérieur. Si cela se produit, il peut y avoir un délai entre le moment où le broker a éprouvé une panne et un message de volonté en cours de publication.

3.2.2.2.6 clean Session

Position: bit 1 de l'octet Connect Flags. Ce bit spécifie la gestion de l'état Session.

Le drapeau clean session indique au broker, que le client souhaite établir une session persistante ou non. Le client et le broker peuvent stocker l'état de session pour permettre une messagerie fiable de continuer à travers une séquence de connexions réseau. Ce bit est utilisé pour contrôler la durée de vie de l'état Session.

- Si Clean Session est défini sur 0, le broker DOIT reprendre les communications avec le client en fonction de l'état de la session en cours (identifié par l'identifiant du client). S'il n'y a pas de session associée à l'identifiant client, le broker DOIT créer une nouvelle session. Le client et le broker DOIVENT enregistrer la session après que le client et le broker soient déconnectés. Après la déconnexion d'une session qui avait CleanSession réglé sur 0, le broker DOIT stocker plus de messages QoS 1 et QoS 2 qui correspondent aux souscriptions que le client avait au moment de la déconnexion dans le cadre de l'état Session. Il PEUT également stocker des messages QoS 0 répondant aux mêmes critères.
- Si Clean Session est défini sur 1, le Client et le Broker DOIVENT renoncer à toute Session précédente et en créer une nouvelle. Cette session dure aussi longtemps que la connexion réseau. Les données d'état associées à cette session NE DOIVENT PAS être réutilisées lors d'une session ultérieure.

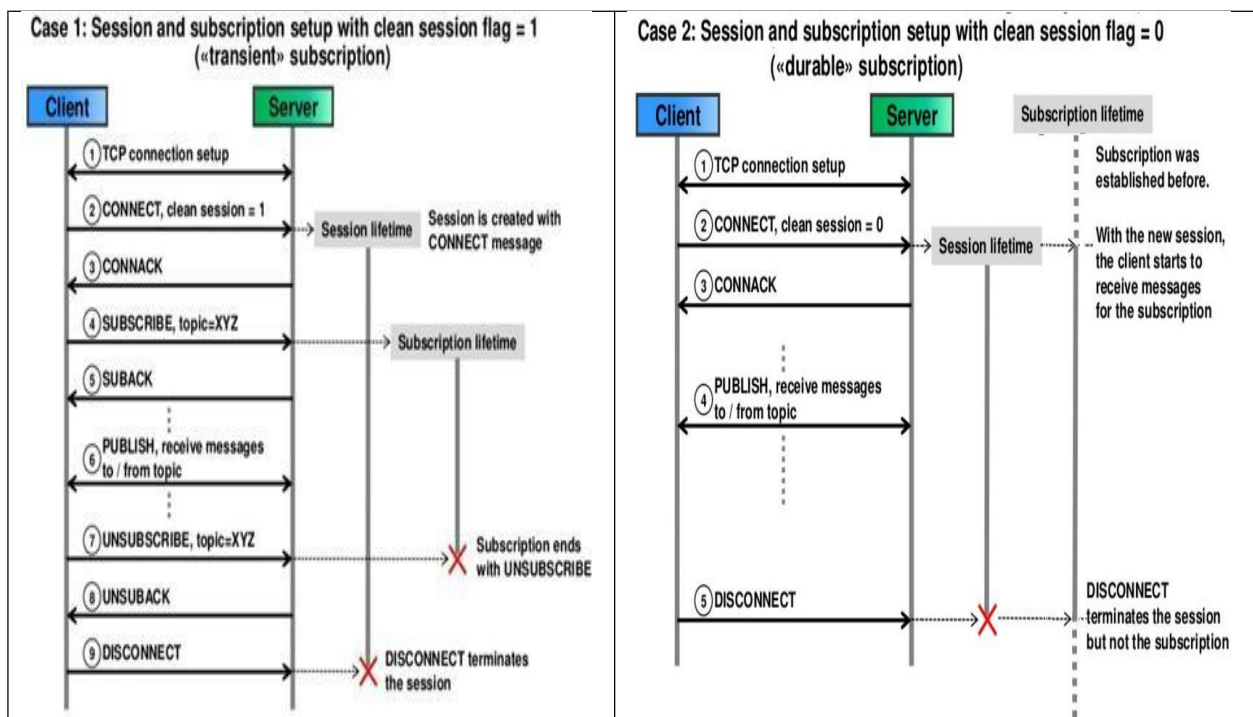


Figure 3-7: Exemple d'utilisation du drapeau clean session [81]

3.2.2.3 Keep Alive

Keep Alive est un intervalle de temps mesuré en secondes. Exprimé en tant que mot de 16 bits, c'est l'intervalle de temps maximum qui peut s'écouler entre le point auquel le client finit de transmettre un paquet de contrôle et le point qu'il commence à envoyer le prochain. Il revient au client de s'assurer que l'intervalle entre les paquets de contrôle envoyés ne dépasse pas la valeur Keep Alive. En l'absence d'envoi d'autres paquets de contrôle, le client **DOIT** envoyer un paquet PINGREQ.

Le client peut envoyer PINGREQ à tout moment, indépendamment de la valeur Keep Alive, et utiliser le PINGRESP pour déterminer si le réseau et le broker fonctionnent.

Si la valeur Keep Alive est non nulle et que le broker ne reçoit pas un paquet de contrôle du client en une fois et demi la période de temps de Keep Alive, il **DOIT ANNULER** la connexion réseau au client comme si le réseau avait échoué.

Si un client ne reçoit pas un paquet PINGRESP dans un délai raisonnable après avoir envoyé un PINGREQ, il **DOIT** fermer la connexion réseau au broker.

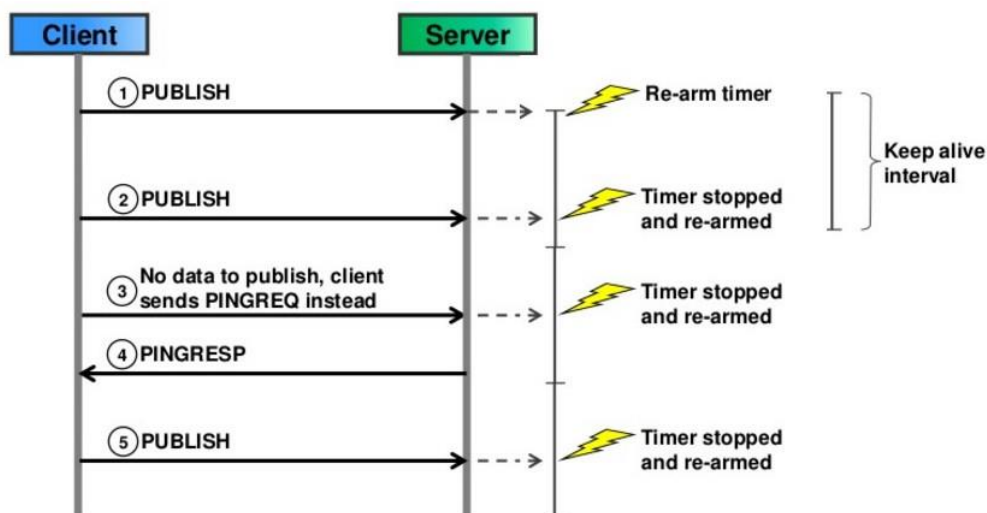



Figure 3-8: Exemple d'utilisation du drapeau Keep alive [81]

3.2.2.4 Payload (Charge utile) : La charge utile du paquet CONNECT contient un ou plusieurs champs préfixés en longueur, dont la présence est déterminée par les indicateurs dans l'en-tête de la variable. Ces champs, le cas échéant, **DOIVENT** apparaître dans l'ordre: Identifiant du client, Nom d'utilisateur, Mot de passe, Sujet (Will topic), WillMessage,

Exemple: de paquet MQTT- CONNECT [59]

MQTT-Packet: CONNECT 	
contains:	Example
clientId	"client-1"
cleanSession	true
username (optional)	"hans"
password (optional)	"letmein"
lastWillTopic (optional)	"/hans/will"
lastWillQos (optional)	2
lastWillMessage (optional)	"unexpected exit"
keepAlive	60

3.2.3 CONNACK

L'entête du CONNACK est constitué d'un bit (SP) pour indiquer si le broker possède déjà une session pour ce client, dans le cas où le client demande à restaurer une session existante, et d'un code de retour sur un octet (seuls les codes de 0 à 5 étant actuellement définis comme illustré dans le tableau3-3).

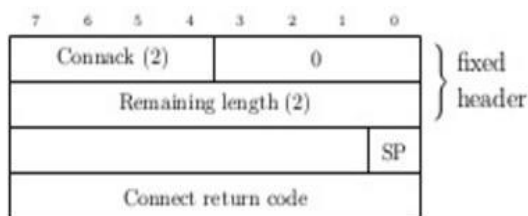


Figure 3-9: Format d'un paquet CONNACK

3.2.3.1: Les valeurs du code de retour

Valeur	Réponse du code de retour	Description
0	0x00 Connexion acceptée	Connexion acceptée
1	0x01 Connexion rejetée, Version protocole inacceptable	Le broker ne prend pas en charge le niveau du protocole MQTT demandé par le client
2	0x02 Connexion refusée, identifiant rejeté	L'identifiant du client est correct UTF-8 mais pas autorisé par le broker
3	0x03 Connexion refusée, broker indisponible	La connexion réseau a été effectuée mais le service MQTT n'est pas disponible
4	0x04 Connexion refusée, mauvais nom d'utilisateur ou mot de passe	Les données dans le nom d'utilisateur ou le mot de passe sont mal formées
5	0x05 Connexion Refusée, non autorisée	Le Client n'est pas autorisé à se connecter
6-255		Réservé pour une utilisation future

Tableau 3-3: Les valeurs du code de retour d'un paquet CONNACK.

2.3.4 PUBLISH

Un paquet PUBLISH est comme illustrer sur la figure suivante:

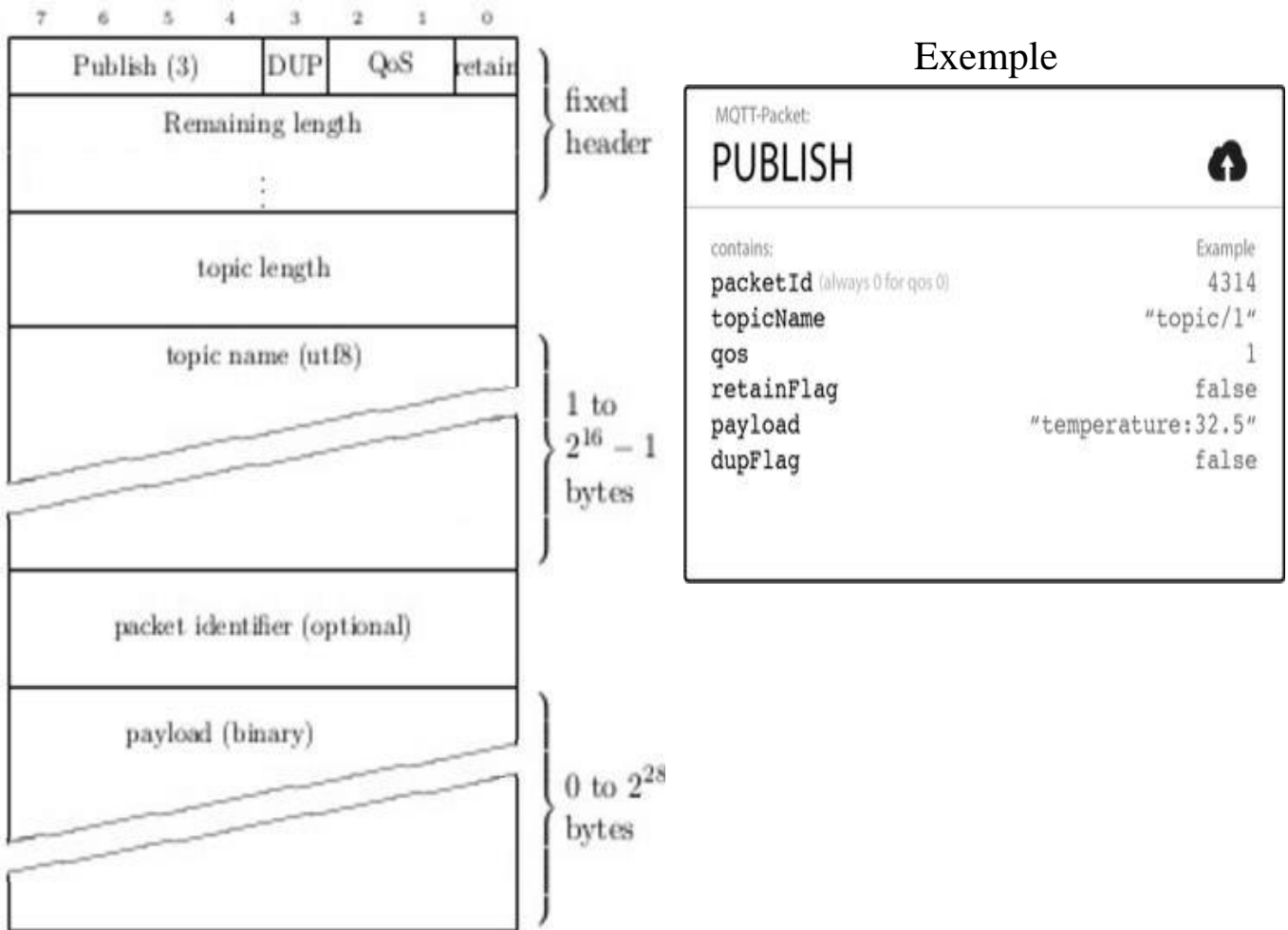


Figure 3-10: *Format d'un paquet PUBLISH*

Un paquet PUBLISH contient les valeurs:

4.2.3.2 Fixedheader

a) DUP

- Si l'indicateur DUP est réglé sur 0, cela indique que c'est la première fois que le Client ou le Broker a tenté d'envoyer ce paquet MQTT PUBLISH.
- Si l'indicateur DUP est défini sur 1, cela indique que cela pourrait être la remise à neuf d'une tentative antérieure d'envoi du paquet.

b) QoS: Ce champ indique le niveau d'assurance pour la livraison d'un message d'application.

- Si un broker ou un client reçoit un paquet PUBLISH qui a deux bits QoS réglés sur 1, il **DOIT** fermer la connexion réseau.

c) RETENIR

- Si l'indicateur RETAIN est réglé sur 1, dans un paquet PUBLISH envoyé par un client à un broker, le broker DOIT stocker le message d'application et sa QoS, afin qu'il puisse être envoyé aux futurs abonnés dont les abonnements correspondent à son nom de sujet.

- Si le broker reçoit un message QoS 0 avec l'indicateur RETAIN réglé sur 1, il DOIT rejeter tout message précédemment retenu pour ce sujet. Il DOIT stocker le nouveau message QoS 0 comme nouveau message retenu pour ce sujet.

d) Champ de longueur restante

Il s'agit de la longueur de l'en-tête variable plus la longueur de la charge utile.

4.2.3.3.En-tête variable

L'en-tête variable contient les champs suivants dans l'ordre: Nom du sujet, Identificateur de paquets.

a) Nom du sujet (topic name)

Le nom du sujet identifie le canal d'information auquel les données de la charge utile sont publiées.

Le nom du sujet DOIT être présent comme premier champ dans l'en-tête variable de paquet PUBLISH. Il DOIT être une chaîne codée UTF-8 telle que définie dans la section 2.3.

b) Identificateur de paquet

Le champ Identificateur de paquets n'est présent que dans les paquets PUBLISH où le niveau QoS est 1 ou 2.

3.2.4 PUBACK, PUBREC, PUBREL, PUBCOMP

Tous ces paquets ont la même structure, seules les valeurs du code de commande et du champ **Reserved** peuvent varier.(voir tableau 3-1).

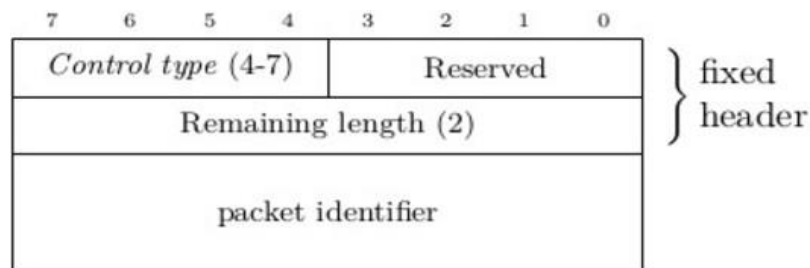


Figure 3-11: – *Format d'un paquet PUB [ACK/REC/REL/COMP]*

3.2.5 SUBSCRIBE

Un paquet SUBSCRIBE envoyé par un client comprend un identifiant pour la réponse, puis une **liste** de souscriptions d’au moins un élément. La structure d’une souscription est composée d’un filtre (longueur et chaine de caractères) et de la qualité de service demandée.

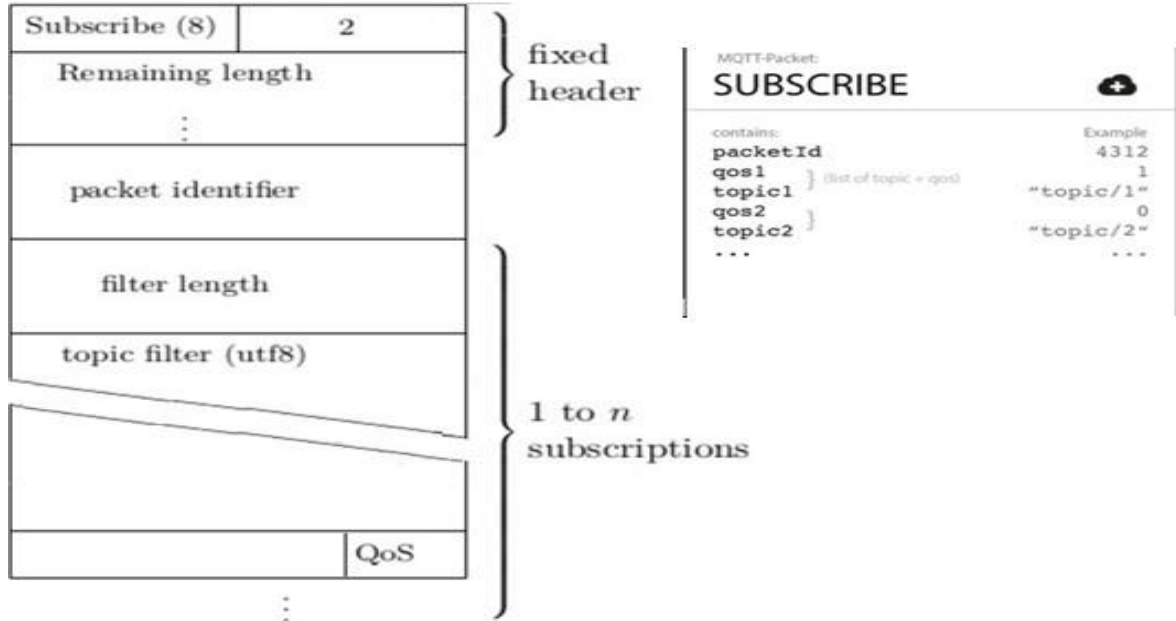


Figure 3-12: Format d’un paquet SUBSCRIBE

Les bits 3,2,1 et 0 de l’en-tête fixe du paquet de contrôle SUBSCRIBE sont réservés et DOIVENT être réglés respectivement à 0,0,1 et 0. Le broker DOIT traiter toute autre valeur comme mal formée et fermer la connexion réseau

Exemple de paquet SUBSCRIBE :

Topic Name	"a/b"
Requested QoS	0x01
Topic Name	"c/d"
Requested QoS	0x02

	Description	7	6	5	4	3	2	1	0
Topic Filter									
byte 1	Length MSB (0)	0	0	0	0	0	0	0	0
byte 2	Length LSB (3)	0	0	0	0	0	0	1	1
byte 3	'a' (0x61)	0	1	1	0	0	0	0	1
byte 4	'/' (0x2F)	0	0	1	0	1	1	1	1
byte 5	'b' (0x62)	0	1	1	0	0	0	1	0
Requested QoS									
byte 6	Requested QoS(1)	0	0	0	0	0	0	0	1
Topic Filter									
byte 7	Length MSB (0)	0	0	0	0	0	0	0	0
byte 8	Length LSB (3)	0	0	0	0	0	0	1	1
byte 9	'c' (0x63)	0	1	1	0	0	0	1	1
byte 10	'/' (0x2F)	0	0	1	0	1	1	1	1
byte 11	'd' (0x64)	0	1	1	0	0	1	0	0
Requested QoS									
byte 12	Requested QoS(2)	0	0	0	0	0	0	1	0

3.2.6 SUBACK

Un paquet SUBACK est envoyé par le broker au client pour confirmer la réception et le traitement d'un paquet SUBSCRIBE.

Dans un paquet SUBACK, le broker répond avec le même identifiant de paquet, puis retourne les codes de retour dans le même ordre que les souscriptions du paquet SUBSCRIBE correspondant.

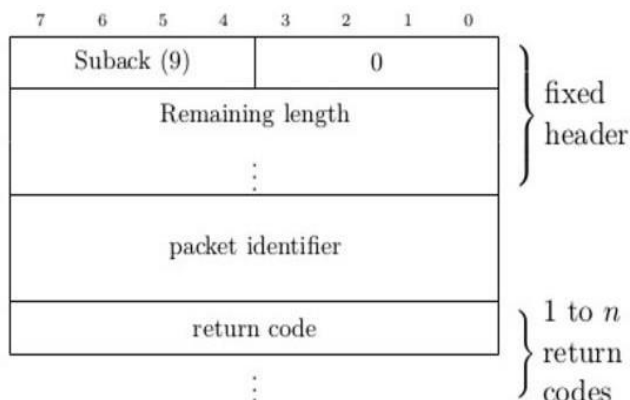


Figure 3-13 – *Format d'un paquet SUBACK*

Un paquet SUBACK contient une liste de codes de retour, qui spécifient le niveau de QoS maximum qui a été accordé dans chaque abonnement qui a été demandé par le SUBSCRIBE.

Ces codes de retour correspondent à la qualité de service garantie par le broker (qui peut être inférieure à celle initialement demandée), ou le code d'erreur 0x80 en cas de problème.

Description de code de retour		7	6	5	4	3	2	1	0
0x00 Success - Maximum QoS 0	0	0	0	0	0	0	0	0	0
0x01 Success - Maximum QoS 1	1	0	0	0	0	0	0	0	1
0x02 Success - Maximum QoS 2	2	0	0	0	0	0	0	1	0
0x80 Failure	128	1	0	0	0	0	0	0	0

Tableau 3-4: *Code de retour de paquet SUBACK*

3.2.7 UNSUBSCRIBE

Un paquet UNSUBSCRIBE est envoyé par le client au broker, pour se désabonner des sujets, il ressemble à SUBSCRIBE, à ceci près que le client ne spécifié pas de qualité de service.

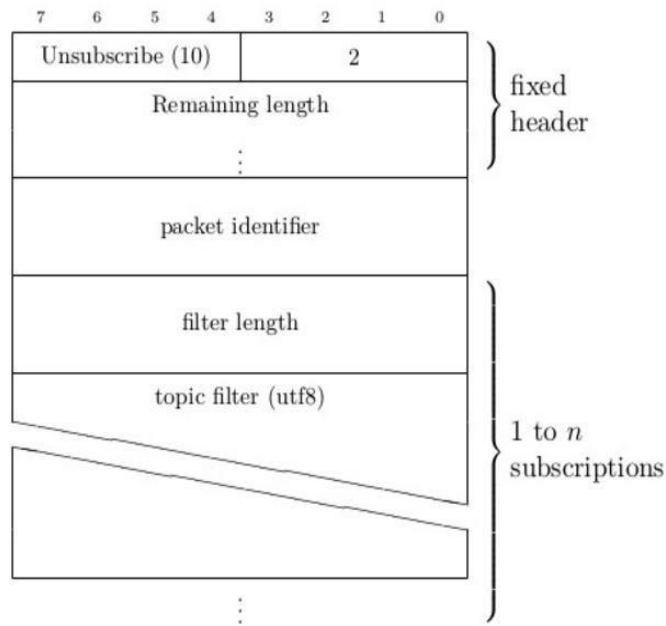


Figure 3-14: *Format d'un paquet UNSUBSCRIBE*

3.2.8 UNSUBACK

UNSUBACK sert de confirmation pour UNSUBSCRIBE, et contient simplement l'identifiant de paquet de la requête correspondante.

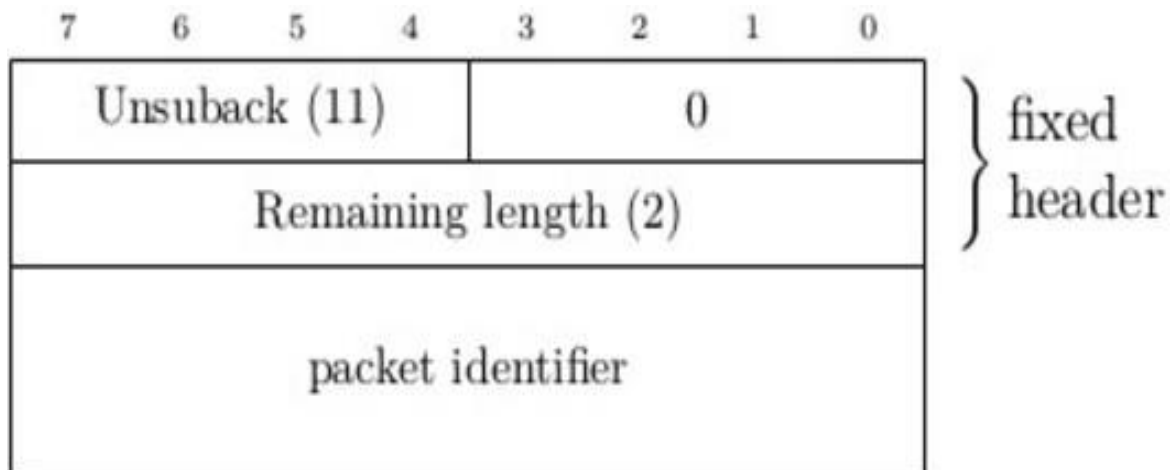


Figure3-15: *Format d'un paquet UNSUBACK*

3.2.9 PINGREQ, PINGRESP

Ces deux paquets sont très simples, et sont constitués uniquement de l'entête fixe du protocole, tenant ainsi sur deux octets.

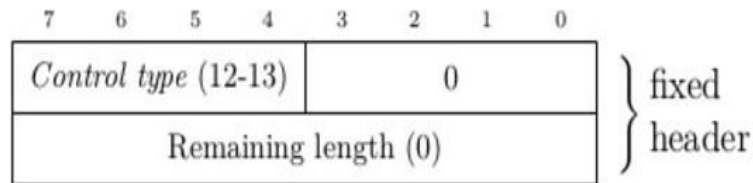


Figure 3-16 – Format d'un paquet PING [REQ/RESP]

*- Le paquet **PINGREQ** est envoyé d'un client au broker. Il peut être utilisé pour:

- 1- Indiquez au broker que le client est en vie en l'absence de tout autre paquet de contrôle envoyé par le client au broker.
- 2- Demandez au Broker de répondre pour confirmer qu'il est en vie.
- 3- Exercer le réseau pour indiquer que la connexion réseau est active.

*- Un paquet **PINGRESP** est envoyé par le broker au client en réponse à un paquet PINGREQ. Cela indique que le broker est en vie.

Ces paquets sont utilisés dans le traitement **Keep Alive**, voir la section **3-2-2-3**

3.2.10 DISCONNECT

Le paquet DISCONNECT est le paquet de contrôle final envoyé du client au broker. Cela indique que le client a l'intention de se déconnecter, et comporte uniquement l'entête fixe.

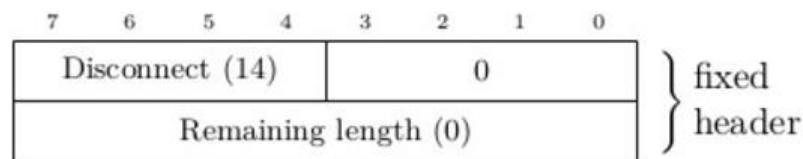


Figure 3-17 – Format d'un paquet DISCONNECT

Sur réception de DISCONNECT le Broker DOIT rejeter tout message de volonté (Will message) associé à la connexion actuelle sans le publier.

3.3 La synchronisation des requêtes envoyées au broker [83]

Le tableau 3.5 décrit les méthodes du client MQTT qui envoient une demande au broker. À moins que le client d'application fixe un délai d'attente indéfinie, le client attend indéfiniment jamais pour le broker. Si le client se bloque, il est soit un problème de programmation d'application, ou un défaut dans le client MQTT.

Méthode	Synchronisation	intervalle délai d'attente
Connect	Temps d'attente pour une connexion à établir avec le broker	La valeur par défaut de 30 secondes, ou tel que défini par un paramètre, lance alors une exception
Disconnect	Temps d'attente pour le client MQTT pour terminer les travaux qu'il doit faire, et pour la session TCP / IP pour déconnecter	
Subscribe UnSubscribe	Attend la fin de la méthode abonner ou désabonner	
Publish	Retourne immédiatement au fil d'application après avoir passé la demande au client MQTT.	Aucun

Tableau 3-5: *Comportement de synchronisation des méthodes qui donnent lieu à des requêtes au broker*

3.4 L'identifiant de client MQTT

Le protocole MQTT définit un «identifiant client» (ID client) qui identifie de manière unique un client dans un réseau. En termes simples, lors de la connexion à un broker, un client doit spécifier une chaîne unique qui n'est pas utilisée actuellement et ne sera pas utilisée par un autre client qui se connectera au broker MQTT. Il existe plusieurs façons de choisir un identifiant client. Voici quelques exemples :

- Un capteur installé dans un emplacement particulier peut utiliser le code d'emplacement comme ID du client.
- Un appareil mobile possédant une capacité de réseau peut choisir l'adresse MAC ou un identifiant d'appareil unique comme ID de client mais MQTT restreint la longueur d'ID de client à 23 caractères. Par conséquent, il y aura une situation où l'identification du client doit être raccourcie.

En raccourcissant l'identifiant du client, il faut s'assurer que l'ID client n'est pas identique à toute autre ID client utilisée dans le réseau. Afin de garder l'identifiant court et unique, il faut

présenter un mécanisme de génération d'identifiant fiable. Par exemple, créer un identifiant client à partir du périphérique MAC 48 bits adresse. Si la taille de la transmission n'est pas un problème critique, et utiliser les 17 octets restants pour rendre l'adresse plus facile à administrer, comme un texte lisible par l'utilisateur dans l'identifiant.

Maintenant, essayons de comprendre les implications de deux clients obtenant le même identifiant client.

Le broker MQTT surveille les messages en attente d'être envoyés à un client en fonction de l'identifiant du client. Ainsi, si un client utilise QoS 1 ou QoS 2 et abonné à n'importe quel sujet et déconnecté du broker, le broker sauvera les messages arrivés pour le client pendant qu'il était déconnecté. Une fois que le client se reconnecte, le broker enverra ces messages au client. Si un autre appareil MQTT utilise le même ID de client et se connecte au broker, le broker enverra les messages qu'il avait sauvegardés.

Un autre scénario lié aux identifiants clients est la connexion en double. Disons qu'un périphérique particulier à l'aide de l'ID client DeviceA est connecté au courtier. Si un autre client vient avec le même ID de client DeviceA, le broker peut décider s'il doit permettre au nouveau client de se connecter et de déconnecter le client existant ou de garder l'ancienne connexion en vie et interdire le nouveau client. Il s'agit d'une fonctionnalité facultative d'un broker MQTT.

3.5 Callbacks [60]

Le modèle de programmation client MQTT utilise des threads en profondeur. Les threads découplent une application client MQTT, autant qu'ils le peuvent, des retards dans la transmission de messages vers et depuis le broker. Les publications, les jetons de livraison et les événements perdus de connexion sont livrés aux méthodes d'une classe de rappel qui implémente `MqttCallback`.

L'interface `MqttCallback` comporte trois méthodes de rappel

3.5.1 ConnectionLost

- `Connection Lost` est appelé lorsqu'une erreur de communication entraîne la sortie de la connexion. Il est également appelé si le broker abandonne la connexion à la suite d'une erreur sur le broker une fois la connexion établie. Les erreurs de broker sont enregistrées dans le journal des erreurs du gestionnaire de files d'attente. Le broker supprime la connexion au client et le client appelle `MqttCallback.connectionLost`.

- Les seules erreurs distantes jetées comme exceptions sur le même thread que l'application client sont des exceptions de `MqttClient.connect`. Les erreurs détectées par le broker après que la connexion est établie sont un rapport à la `MqttCallback.connectionLost` méthode de rappel comme `Throwables`.

- Les erreurs typiques du broker qui résultent en `connectionLost` sont des erreurs d'autorisation. Par exemple, le broker de télémétrie tente de publier sur un sujet pour le compte d'un client qui n'est pas autorisé à publier sur le sujet. Tout ce qui se traduit par une `MQCC_FAIL` Le code d'état retourné au broker de télémétrie peut entraîner la chute de la connexion.

3.5.2 DeliveryComplete

- `DeliveryComplete` est appelé par le client MQTT pour passer un jeton de livraison à la demande client, En utilisant le jeton de livraison, le rappel peut accéder au message publié avec la méthode `token.getMessage`.

- Lorsque le retour d'appel de l'application renvoie le contrôle au client MQTT après avoir été appelé par la méthode `deliveryComplete`, la livraison est terminée. Jusqu'à ce que la livraison soit terminée, les messages avec QoS 1 ou 2 Sont conservés par la classe de persistance.

- L'appel à `DeliveryComplete` est un point de synchronisation entre l'application et la classe de persistance. La méthode `DeliveryComplete` n'est jamais appelée deux fois pour le même message.

- Lorsque le retour d'appel de l'application revient de `DeliveryComplete` au client MQTT, le client appelle `MqttClientPersistence.remove` pour les messages avec QoS1 ou 2. `MqttClientPersistence.remove` supprime la copie stockée localement du message publié.

- Du point de vue du traitement des transactions, l'appel à `deliveryComplete` est une transaction monophasée qui engage la livraison. Si le traitement échoue pendant le rappel, lors du redémarrage du client, `MqttClientPersistence.remove` est appelé à nouveau pour supprimer la copie locale du message publié. Le rappel n'est pas encore appelé. Si vous utilisez le rappel pour stocker un journal des messages livrés, vous ne pouvez pas synchroniser le journal avec le client MQTT. Si vous souhaitez stocker un journal de manière fiable, mettez à jour le journal dans la classe `MqttClientPersistence`.

- Le jeton de livraison et le message sont référencés par le thread d'application principal et le client MQTT. Le client MQTT désélectionne l'objet `MqttMessage` lorsque la livraison est terminée et l'objet token de livraison lorsque le client se déconnecte. L'objet `MqttMessage` peut être récupéré lorsque la livraison est terminée si l'application cliente la détermine. Le jeton de livraison peut être récupérée après la déconnexion de la session.

- Vous pouvez obtenir `MqttDeliveryToken` et `MqttMessage` attributs après un message a été publié. Si vous tentez de définir des attributs `MqttMessage` après la publication du message, le résultat est indéfini.

- Le client MQTT continue de traiter les remerciements de la livraison si le client se reconnecte à la session précédente avec le même `ClientIdentifier`. L'application client MQTT doit configurer `MqttClient.CleanSession` pour faux Pour la session précédente, et la configurer faux Dans la nouvelle session. Le client MQTT crée de nouveaux jetons de livraison et des objets de message

dans la nouvelle session pour les livraisons en attente. Il récupère les objets à l'aide de la classe `MqttClientPersistence`. Le rappel d'application est appelé dans la nouvelle session pour toutes les livraisons initiées dans la session précédente et complétées dans cette session.

- Le rappel de l'application est appelé après la connexion du client de l'application, lorsqu'une livraison en attente est terminée. Avant que le client de l'application ne se connecte, il peut récupérer les livraisons en attente en utilisant la méthode `MqttClient.get Pending Delivery Tokens`.

3.5.3 MessageArrived

- ✓ `MessageArrived` s'appelle lorsqu'une publication arrive pour le client correspondant à un sujet d'abonnement. Le sujet est le sujet de publication, pas le filtre d'abonnement. Les deux peuvent être différents si le filtre contient des caractères génériques.

- ✓ Si le sujet correspond à plusieurs abonnements créés par le client, le client reçoit plusieurs copies de la publication. Si un client publie sur un sujet auquel il s'inscrit également, il reçoit une copie de sa propre publication.

- ✓ Si un message est envoyé avec une QoS de 1 ou 2, Le message est mémorisé par la **classe `MqttClientPersistence`** avant que le client MQTT n'appelle message. **`MessageArrived`** se comporte comme `deliveryComplete`: il est appelé uniquement une fois pour une publication, et la copie locale de la publication est supprimée par **`MqttClientPersistence.remove`** lorsque **`messageArrived`** retourne au client MQTT.

4 Conclusion

Nous avons vu que MQTT est un protocole tout à la fois minimaliste et très simple à mettre en œuvre (en particulier du côté client). Malgré tout, il reste suffisamment souple et extensible pour satisfaire aux besoins d'un bon nombre d'applications, et est donc particulièrement adapté au contexte de l'IoT. C'est la raison pour laquelle nous avons choisi ce protocole pour l'implantation et la réalisation d'une application de contrôle de température et humidité dans IoT.

DEUSIÈME PARTIE : CONTRIBUTION

CHAPITRE 4 : CONCEPTION ET RÉALISATION DE SYSTÈME DE CONTRÔLE A BASE DE MQTT

Chapitre 4

Conception et réalisation de système de control à base de MQTT

1. Introduction

Nous entamons l'implémentation de notre projet, qui consiste en *l'implémentation du protocole MQTT pour le contrôle de la température et de l'humidité dans un système à base de IoT*. Pour ce faire, nous allons suivre une démarche génie logiciel en procédant par l'analyse des besoins, la conception, la réalisation et le déploiement du système d'IoT. Pour concrétiser la communication entre les différents composants du système, nous allons utiliser quelques protocoles de communication dédiés aux systèmes d'IoT. Ce chapitre commence par la présentation de la phase de l'analyse des besoins. Cette phase vise à étudier et analyser les exigences fonctionnelles et non-fonctionnelles ainsi que la désignation du matériel nécessaire pour la mise en œuvre d'un tel système. Ainsi, un aperçu sur le fonctionnement global du système est donné par la suite décrivant le « quoi » réaliser. Notre projet s'inscrit dans le cadre des systèmes d'IoT (Internet of Things), et il s'intéresse particulièrement à surveiller deux paramètres : température et humidité et de prendre en compte les valeurs transmises par les différents dispositifs situés dans l'environnement (capteur d'humidité, capteur de température), et de visualiser ces données à distance sur un smart phone.

En fin de la réalisation, on essayera de répondre à certaines questions :

- Comment utiliser l'internet des objets ?
- Comment implémenter le protocole MQTT ?
- Comment faire la communication d'IoT en utilisant le MQTT ?

2. Analyse des besoins du système

2.1 Exigences Fonctionnelles

Ce sont les exigences liées au fonctionnement du système à développer. Le système de contrôle à distance de la température et de l'humidité devra être capable d'établir les fonctionnalités suivantes:

- ✓ L'implémentation de protocole MQTT pour la communication (broker, publisher, subscriber).
- ✓ Acquérir les mesures des grandeurs physiques issues des capteurs utilisés.
- ✓ Permettre la consultation des données.
- ✓ Contrôler, à distance, la température et l'humidité.
- ✓ Visualisation des données sur un smart phone

2.2 Exigences non fonctionnelles

- ✓ Sécurité des données ;
- ✓ Ergonomie des interfaces graphiques ;
- ✓ Compatibilité avec n'importe quel système d'exploitation ;
- ✓ L'extensibilité ;

2.3 Vision de la solution proposée

Notre solution prend en compte les valeurs transmises par les différents dispositifs situés dans l'environnement cible pour le contrôle à distance (capteur d'humidité, capteur de température), pour effectuer le suivi de ces données en temps réel d'une façon fiable et adéquat. La solution à proposer est basée sur le paradigme des systèmes d'IoT. En outre, l'implémentation de protocole de transfert MQTT.

Une telle solution nécessite les composants matériels et logiciels suivant :

2.3.1 Besoin matériels

- ✓ Carte arduino Uno (Microcontrôleur).
- ✓ Module Wifi model: ESP8266.
- ✓ Carte d'essai (Bread-Board).
- ✓ Fils de connexion.
- ✓ Serveur Machine sous Windows (broker).
- ✓ Réseaux wifi / point d'accès.
- ✓ Smart phone (subscriber).
- ✓ Des capteurs tel que : capteur de température, capteur d'humidité (publisher).

2.3.2 Besoin Software

- ✓ Android Studio ;
- ✓ Arduino IDE sous Windows;
- ✓ Protocole de communication : wifi, TCP/IP, MQTT, etc.

3 Spécification des fonctionnalités du système

3.1 Identification des acteurs :

Nous distinguons les acteurs suivants :

- ✓ **Utilisateur** : Il représente toutes les personnes contribuant au suivi de la température et de l'humidité on utilisant des smart phone.

- ✓ **Objet connecté** : Représente la source de données du système. Son rôle est d'exécuter des actions et d'envoyer les données en fonction de la manière dont il a été programmé.
- ✓ **Broker MQTT** : son rôle a été expliqué précédemment au chapitre 3.

3.2 Identification des cas d'utilisation :

Il est à rappeler qu'un cas d'utilisation représente une unité discrète d'interaction entre un utilisateur (humain ou machine) et un système. La vue statique du comportement fonctionnel du système est représentée par le diagramme de cas d'utilisation de la Figure 4.1.

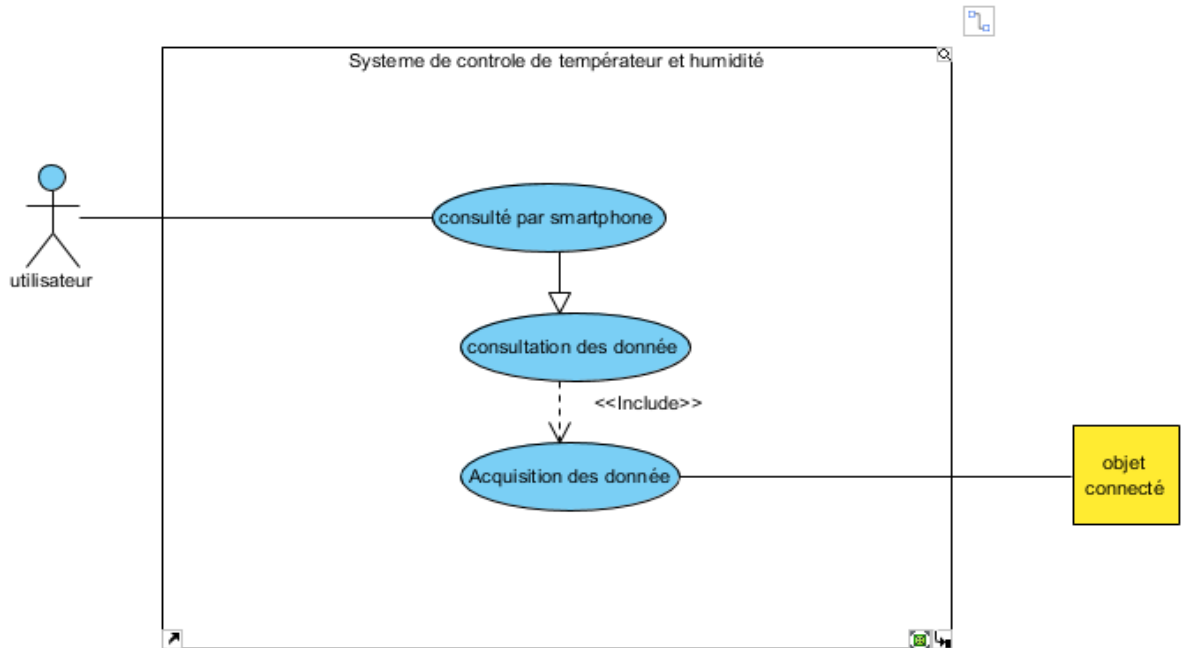


Figure 4.1: Diagramme de cas d'utilisation général

3.2.1 Description textuelle des cas d'utilisation

Le diagramme de cas d'utilisation général représente toutes les interactions possibles entre l'acteur et le système et les actions que le système doit réaliser en vue de produire les résultats attendus par les acteurs. Voilà une description textuelle pour chacune de ces cas d'utilisation :

3.2.1.1 Diagramme de cas d'utilisation « Consultation des données »

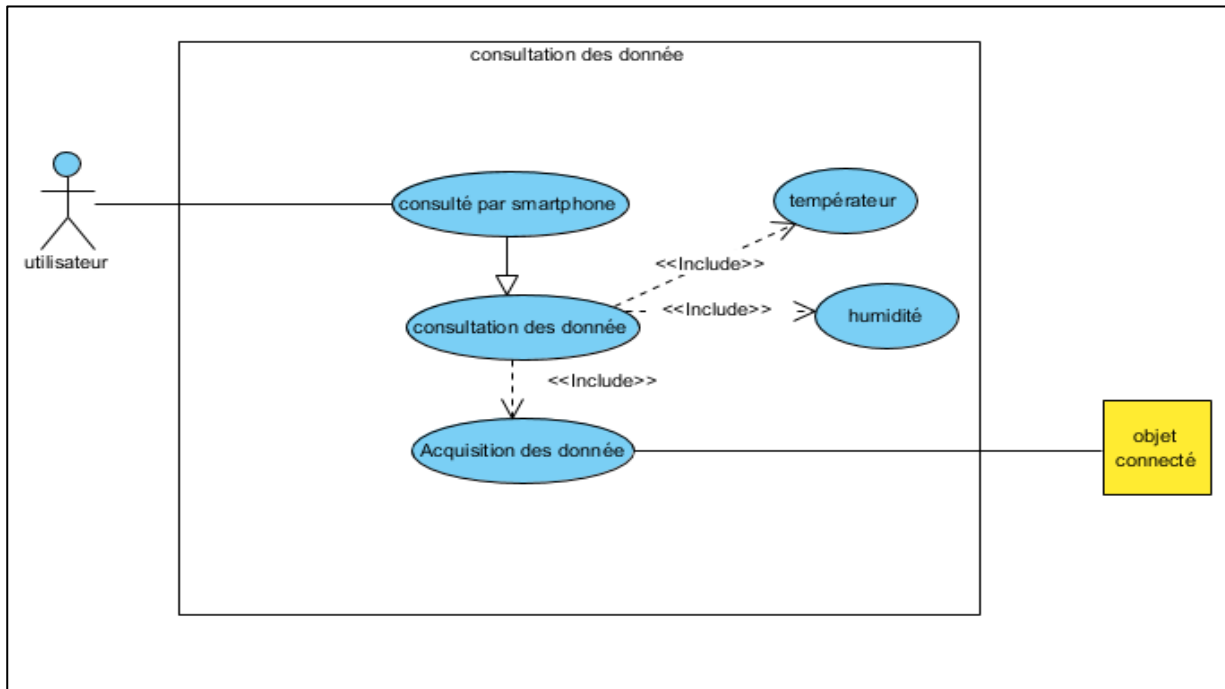


Figure 4.2 : Diagramme de cas d'utilisation « Consultation des données »

Cas d'utilisation	Consultation des données
Acteurs principaux	Utilisateur
Acteurs secondaires	/
Objectif	l'utilisateur peut utiliser une application smartphone pour consulter les données suivantes : Température, Humidité
Prés conditions	<ul style="list-style-type: none"> • L'objet connecté est bien placé et bien installé dans son environnement. • L'existence d'un réseau internet sans fil. • Le serveur (Broker) fonctionne très bien et toujours en états d'écoute.
Poste conditions	/
Scénario nominale	<ol style="list-style-type: none"> 1. l'utilisateur lance son application smartphone. 2. L'application demande la connexion au broker MQTT. 3. l'utilisateur consulte les données.
Scénario alternative	<ul style="list-style-type: none"> • Si le wifi n'est pas activer alors l'application demande l'activation du wifi

Tableau 4.1 : Fiche descriptive du cas d'utilisation : " Consultation des données "

3.2.1.2 Diagramme de cas d'utilisation « Acquisition des données »

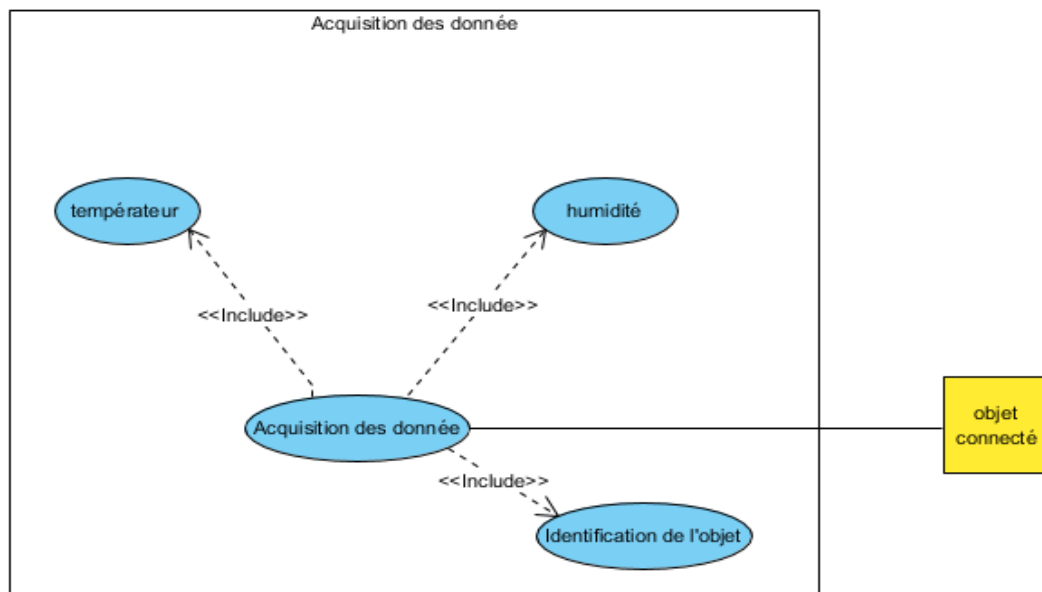


Figure 4.3 : Diagramme de cas d'utilisation « Acquisition des données »

Cas d'utilisation	Acquisition des données
Acteurs principaux	
Acteurs secondaires	Serveur (Broker MQTT), Objet connecté
Objectif	<ul style="list-style-type: none"> Le serveur recevoir des données transmises par l'objet connecté d'une façon continue ou périodique. L'objet connecté adopté aux différents capteurs qui sont liés à lui, pour capter les évènements de son environnement et les envois après ça au serveur.
Prés conditions	<ul style="list-style-type: none"> L'objet connecté est bien placé et bien installé dans l'environnement. L'existence d'un réseau internet sans fil. Le serveur fonctionne très bien et toujours en états d'écoute.
Poste conditions	/
Scénario nominale	<ol style="list-style-type: none"> L'objet connecté collecte les données captés depuis son environnement et les envois au serveur périodiquement ou d'une façon continue. Le serveur recevoir les données.
Scénario alternative	<ul style="list-style-type: none"> L'objet connecté tombe en panne a cause des raisons inconnues, cela nécessite une intervention immédiate par le propriétaire pour détecter le problème qui a causé l'en panne et le résoudre.

Tableau 4.2 : Fiche descriptive du cas d'utilisation : " Acquisition des données "

3.3 Diagrammes de séquences

Le diagramme de séquence est une représentation graphique qui permet de montrer les interactions entre les acteurs et le système en ordre chronologique et dans le cadre d'un scénario d'un diagramme des cas d'utilisation. Nous présentons ci-dessous l'ensemble des diagrammes de séquence de notre système qui correspondent aux cas d'utilisation présentés précédemment.

3.3.1 Diagrammes de séquence : Acquisition des données

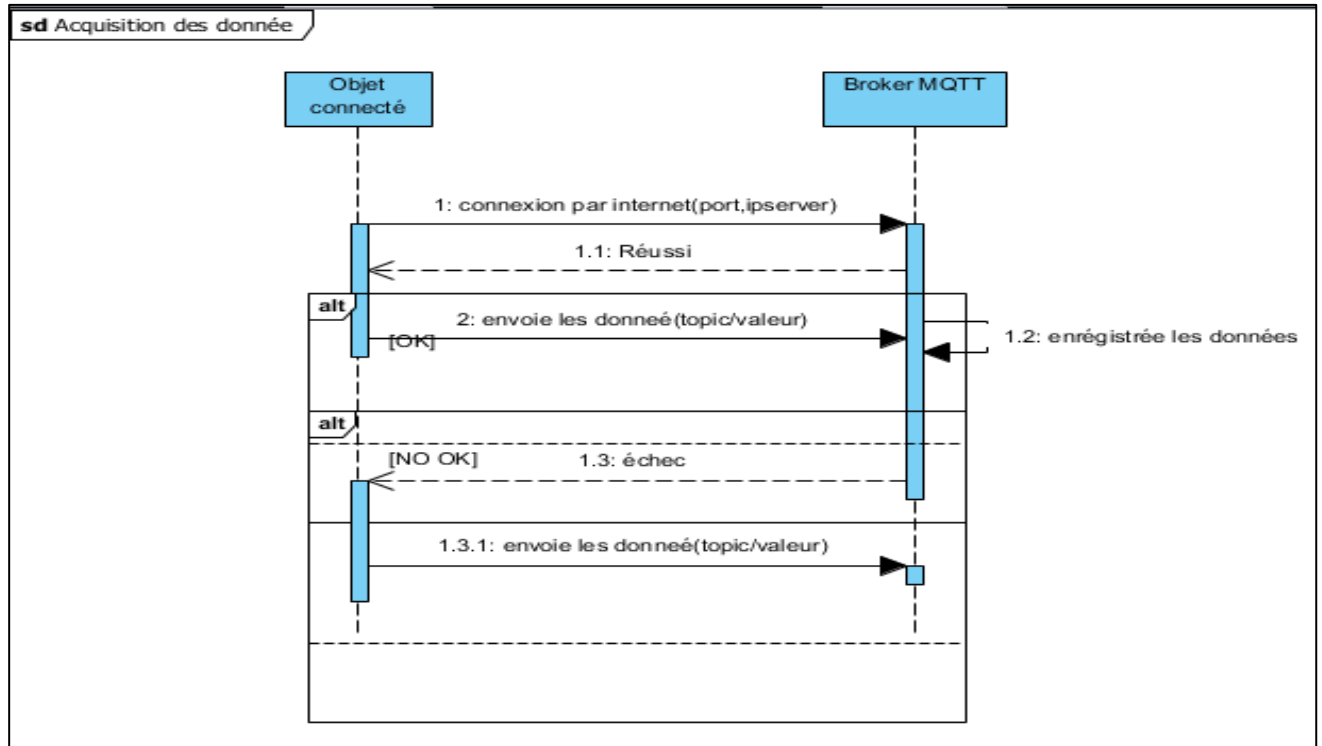


Figure 4.4 : Diagramme de séquence : Acquisition des données

3.3.2 Diagrammes de séquence pour le cas d'utilisation : Consultation des données.

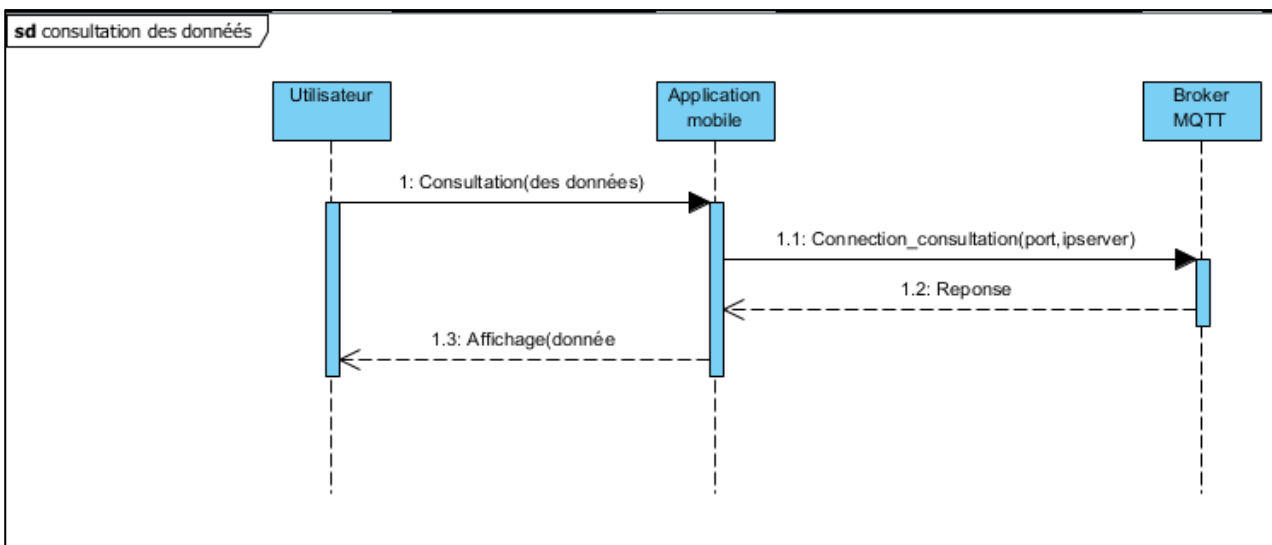


Figure 4.5 : Diagramme de séquence : Consultation des données.

A partir de l'exploitation de protocole MQTT (publication, souscription) de message. Les clients publient des messages sur un broker (*appelé aussi courtier*), les messages sont composés d'un contenu et d'un sujet (topic) [51]. Dans notre projet, les objets connectés (capteur de température, capteur de humidité) jouent le rôle d'un client de publication et les topics sont (temp, hum). Le broker (pc serveur) garde en mémoire le dernier message pour chaque sujet (topic). Les clients (smart phone) qui sont intéressés par les messages d'un sujet peuvent les récupérer en se connectant au broker.

4 La conception :

Après avoir cerné les périmètres du système à réaliser et étudié tous les besoins nécessaires pour sa mise en œuvre, nous passons à la deuxième phase, phase de conception. Dans cette partie, nous allons présenter les différentes architectures proposées (Matérielle et logicielle) ainsi que la manière dont ces architectures vont être réalisées.

4.1 Architecture matérielle

La figure ci-dessous représente l'architecture matérielle de notre système :

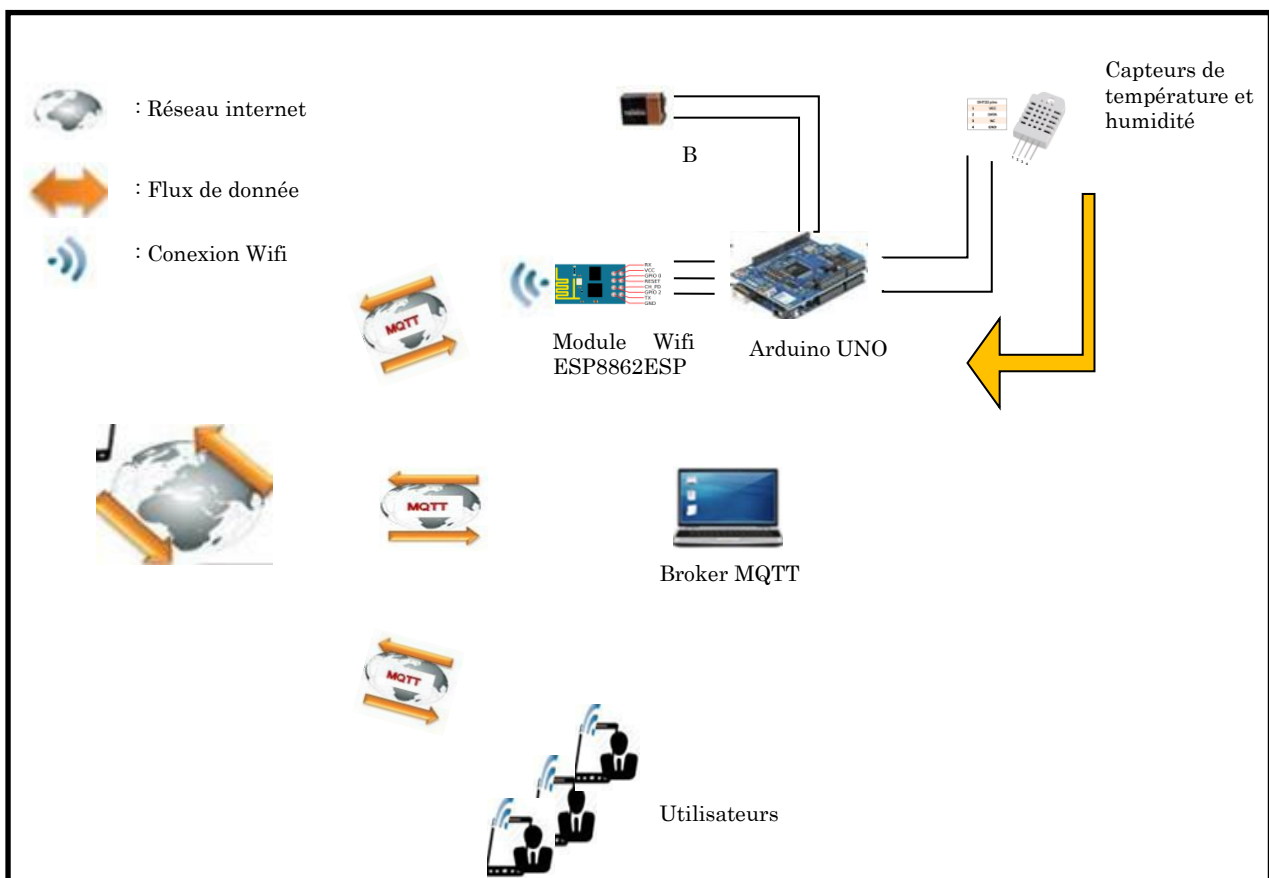


Figure 4.6 : Architecture matérielle du système

4.2 Motivation des choix

Nous avons présenté auparavant l'architecture matérielle du système de contrôle de température et humidité à distance en utilisant le protocole MQTT. Cette architecture est composée de la liste des dispositifs nécessaires pour la réalisation de notre système IoT. Nous allons dans la section suivante motiver le choix de quelque composant matériel :

4.2.1 Le choix de la carte Arduino

Par rapport aux autres cartes microcontrôleurs, les cartes Arduino sont relativement peu coûteuses ; La plupart des systèmes à microcontrôleurs sont limités à Windows, mais arduino est multiplateforme (Le logiciel Arduino, écrit en Java, agit sous les systèmes d'exploitation Windows, Macintosh et Linux) ;

- ✓ Facile à utiliser pour les débutants ;
- ✓ L'environnement de programmation Arduino est publié sous licence open source ;
- ✓ La disponibilité des cartes arduino dans le marché et son faible coût ;
- ✓ L'UNO est compatible avec la plupart des shields conçus pour l'Arduino ;
- ✓ L'auto-alimentation par le port USB (5V) mais dispose également d'une autre alimentation supportant des tensions jusqu'à 7-12V ;

4.2.2 Le choix de Wifi

Nous avons opté dans le l'architecture matérielle de notre système IoT d'un Wifi de type ESP8266 compatible avec la carte arduino UNO pour le but d'échanger le flux de données entre l'objet connecté et le serveur (Broker MQTT) via le réseau internet.

- ✓ Les modules Wifi sont peu coûteuses comparativement aux autres modules de connectivité.
- ✓ Programmation clair et simple soit pour la programmation matérielle ou logicielle.
- ✓ La connectivité par Wifi est adaptée à notre solution matérielle.

4.3 Description technique des dispositifs impliqués dans l'architecture matérielle

Remarque : La description technique pour les dispositifs a déjà été répertoriée lors du premier chapitre.

4.4 Architecture logicielle

La figure suivante représente l'architecture logicielle que nous avons proposée pour notre système :

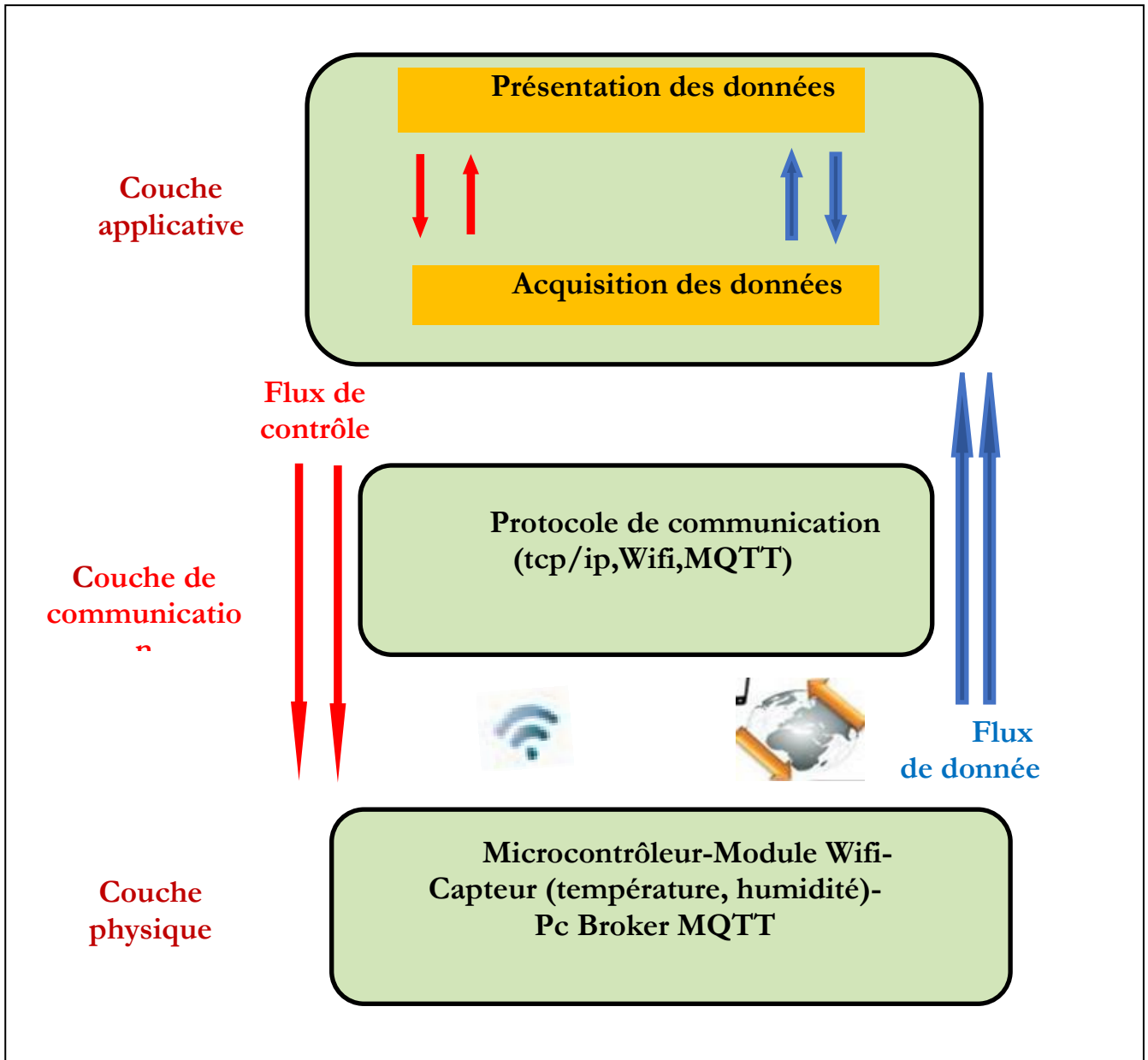


Figure 4.7 : Architecture logicielle du système

4.4.1 Description de l'architecture logicielle

- **La couche physique** : comporte l'ensemble du matériel utilisé dans le système, dans la présente couche nous avons les traitements suivants :
 - Collection des données à partir de l'environnement.
 - Les capteurs transforment les grandeurs physiques en des grandeurs logiques; ensuite le microcontrôleur transforme les signaux à une chaîne binaire.
 - Le module Wifi transfère les données codées vers la couche Applicatif (Acquisition des données).
- **La couche de communication** : représente les protocoles utilisés pour assurer la communication entre les différentes parties de notre système :
 - Protocole TCP/IP : pour la communication entre l'application mobile et le serveur.
 - Protocole Wifi : pour assurer la communication d'objet connecté via l'internet.
 - Protocole MQTT : utilisé pour la communication entre les objets connectés les publisher (les capteurs) et subscriber (smartphone).
- **La couche applicative** : comporte les trois couches suivantes :
 - La couche Acquisition des données : assure la réception de données.
 - La couche Présentation des données : assure l'affichage et la représentation des données sur les smartphone.

4.5 Diagramme de classe

Nous avons réalisé le diagramme de classe pour permettre de représenter une vue statique de notre système et de montrer la structure figée de notre solution logicielle. La figure ci-dessous représente le diagramme de classe de notre système de contrôle de température et humidité à distance.

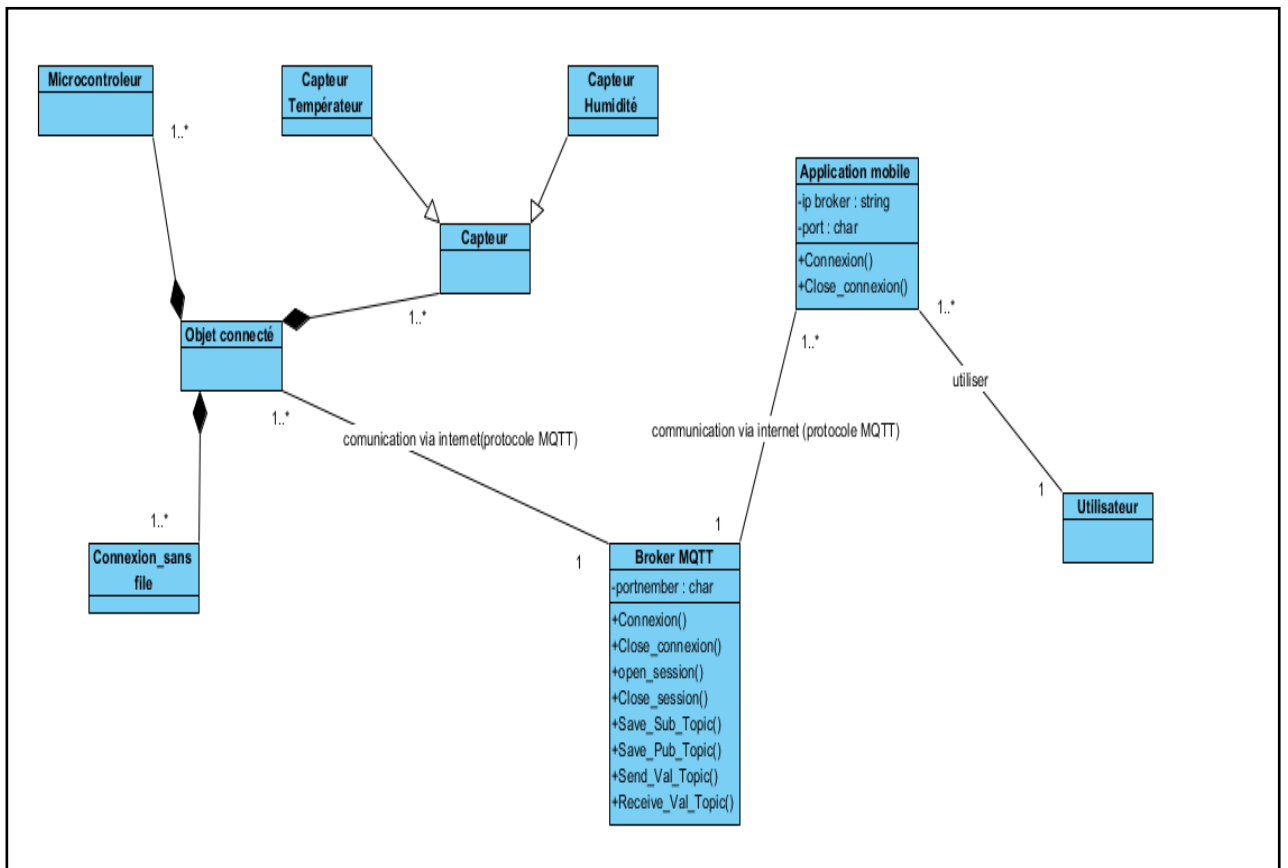


Figure 4.7 : diagramme de classe

Classe	Description
Utilisateur	Représente l'utilisateur du système qui peut via une application mobile consulté les différentes données transmises par l'objet connecté.
Application mobile	- Utiliser par un utilisateur pour pouvoir consulter à distance via le réseau internet les différentes données transmises par l'objet connecté.
Broker MQTT	Joue un rôle d'intermédiaire entre l'objet connecté les capteur (publisher) et les smartphone (subscriber) - Elle reçoit les flots de données transmises par l'objet connecté via la connexion internet et elle les renvois ensuite au smartphone via le réseau internet.
Objet connecté	- Se Compose de différents dispositifs nécessaires (capteurs, Module Wifi, microcontrôleur). Il collecte les données depuis son environnement et les transmises après ça au serveur d'une façon continue ou périodique.

Tableau 4.3 : Description textuelle du diagramme de classe

4.6 Diagramme d'activité

Les diagrammes d'activités permettent de mettre l'accent sur les traitements. Ils sont donc particulièrement adaptés à la modélisation du cheminement de flots de contrôle et de flots de données. Ils permettent ainsi de représenter graphiquement le comportement d'une méthode ou le déroulement d'un cas d'utilisation. Nous représentons ci-dessous les diagrammes d'activités correspondants à chaque cas d'utilisation de notre système :

4.6.1 Diagramme d'activité : Consultation des données

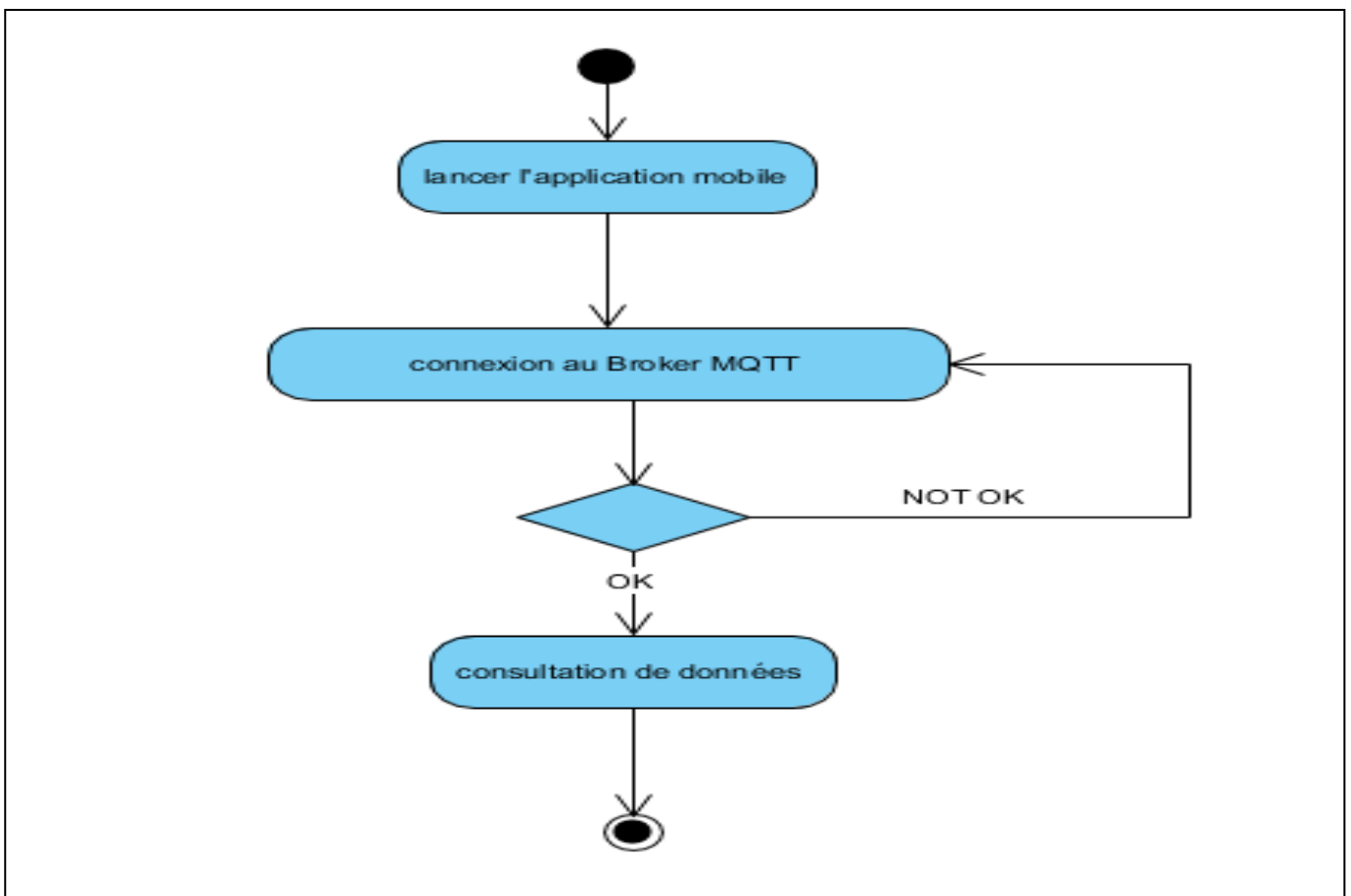


Figure 4.8 : Diagramme d'activité " Consultation des données "

4.6.2 Diagramme d'activité : Acquisition des données

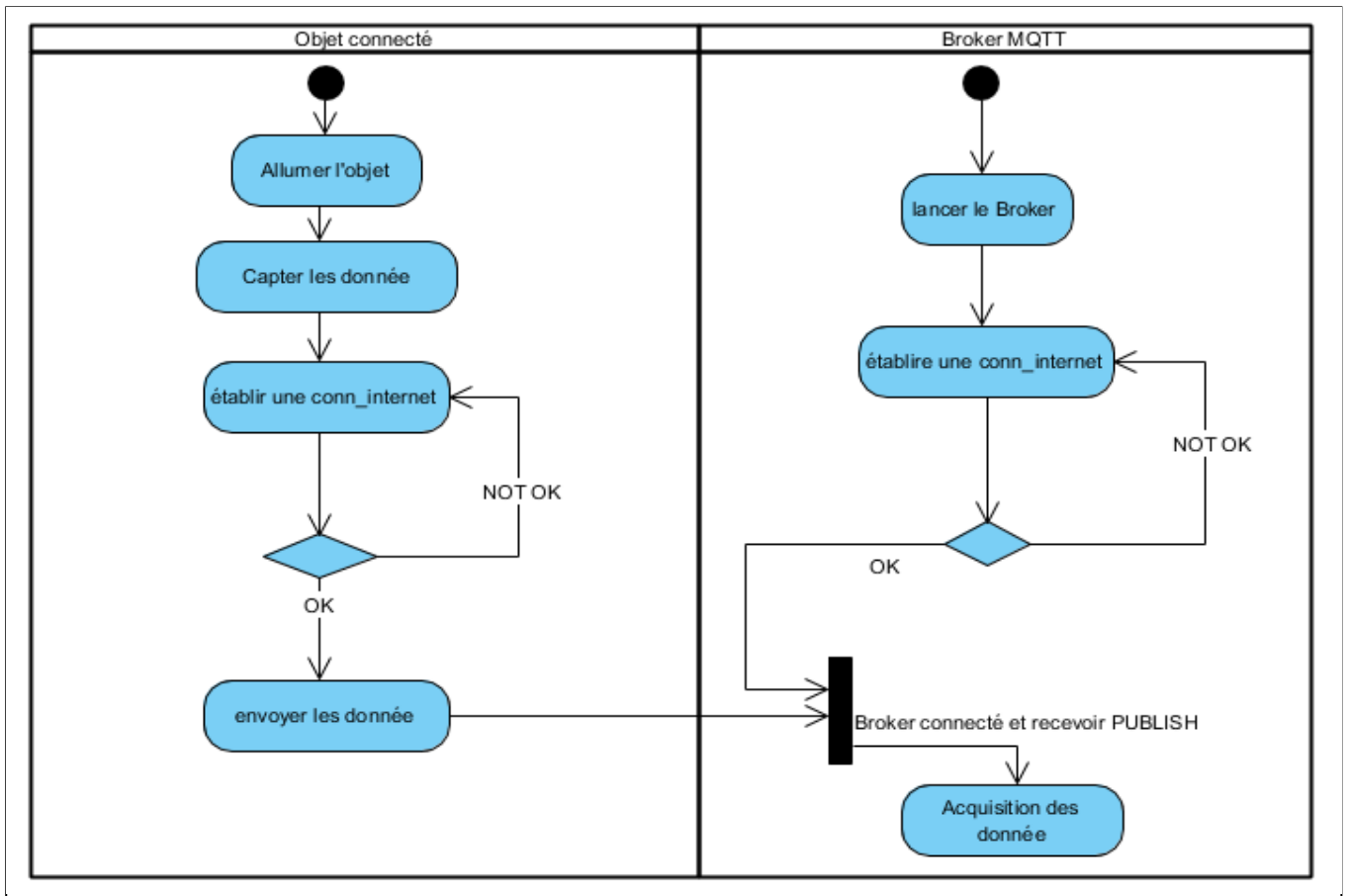


Figure 4.9 : Diagramme d'activité " Acquisition des données "

La conception est une étape très importante qui précède l'implémentation de tout système. Nous rappelons que dans ce travail, nous avons adopté une démarche génie logiciel basée sur le langage UML, nous avons pu concevoir un système d'IoT en se basant sur les diagrammes UML à savoir le diagramme de cas d'utilisation, et le diagramme de séquence, classe, activités. Alors maintenant nous allons entamer la réalisation physique et l'implémentation des différentes couches de l'architecture retenue.

5 La réalisation :

Après avoir analysé les besoins et défini la méthodologie de conception, nous avons utilisé plusieurs outils et de nouvelles technologies pour le développement de ce projet et nous spécifieront l'environnement logiciel supportant notre application et nous allons expliquer l'utilité de chaque outil. Ainsi, nous allons décrire la manière de réalisation du projet.

5.1 Description des langages et outils utilisés

Nous présentons dans cette section, les différents langages de programmation et les outils utilisés qui sont nécessaires à la réalisation de l'application.

5.1.1 Les langages de programmation

C++ : pour la programmation du matériel.

Le langage C++ est un langage de programmation créé par Bjarne Stroustrup et normalisé en 1998. Le ++ est l'opérateur d'incrément (qui fait

+1) : le C++ est donc une amélioration de l'ancestral langage C.

Il apporte notamment la programmation orientée objet (qui est l'apport majeur), la gestion des exceptions, la gestion des références (remplaçant partiellement l'usage quelque peu délicat des pointeurs), la surcharge des opérateurs et les templates (liste non exhaustive).

Enfin, une rétrocompatibilité a été gardée ; les programmes en C compilent sans difficulté avec un compilateur C++. [106].

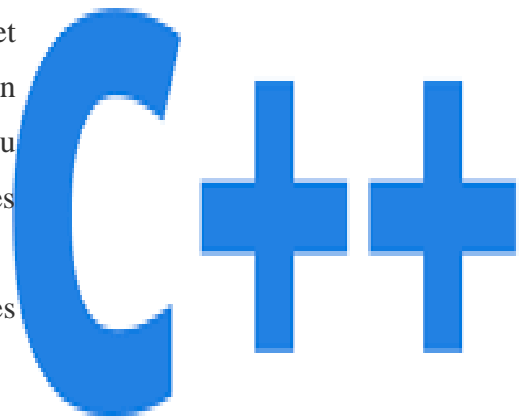


Figure 4.10 : *logo C++*

5.2 Plateforme de développement et outils logiciels

Dans cette partie on représente brièvement les outils de développement qu'on a utilisé durant le développement de notre système.

5.2.1 Plateformes utilisés

✓ **Android** : est le nom du système d'exploitation mobile le plus populaire au monde appartenant à la société américaine Google.

Android est un système d'exploitation ouvert dont le code source est librement accessible (contrairement aux systèmes de Apple ou Microsoft) ce qui permet à n'importe quel fabricant de l'intégrer dans son système gratuitement. Ce modèle est opposé au modèle de Apple, et explique en grande partie la forte croissance que connaît Android [109].

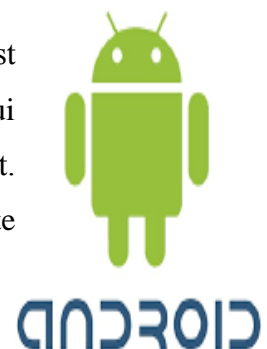


Figure 4.11: *Logo Android*

5.2.2 Logiciels utilisés

✓ Android Studio

Android Studio est un environnement de développement pour développer des applications Android. Il est basé sur IntelliJ IDEA. (IntelliJ IDEA est un IDE Java commercial développé par JetBrains). Android Studio permet principalement d'éditer les fichiers Java et les fichiers de configuration d'une application Android. Il propose entre autres des outils pour gérer le développement d'applications multilingues et permet de visualiser la mise en page des écrans sur des écrans de résolutions variées Simultanément [110].



Figure 109 : Logo Android Studio

Arduino IDE

L'environnement de développement intégré Arduino - ou Arduino Software (IDE) - contient un éditeur de texte pour écrire du code, une zone de message, une console de texte, une barre d'outils avec des boutons pour les fonctions communes et une série de menus. Il se connecte au matériel Arduino et Genuino pour télécharger des programmes et communiquer avec eux. [113]. L'IDE Arduino permet [114] :

- d'éditer un programme : des croquis (sketch en Anglais),
- de compiler ce programme dans le langage « machine » de l'Arduino,
- de téléverser le programme dans la mémoire de l'Arduino,
- de communiquer avec la carte Arduino grâce au terminal.



Fiquer4.12 : logo Arduino IDE

5.2.3 Description de la réalisation physique de l'objet

Notre projet consiste a l'implentation de protocole MQTT dans la base l'internet des objets « contrôle du température et humidité ».Alors les objet connecté de notre système contient un capteur(température et humidité), un module Wifi ESP8266, un microcontrolleur Arduino UNO,point d'accée Internet,smart phone,un pc (server) sous SE Windows.

Chapitre 4 : Conception et réalisation de système de control à base de MQTT

- Un capteur DHT22 AM2302 : pour mesurer la température et l'humidité de l'air.
- Un module Wifi ESP8266 pour envoyer les données mesurant par l'objet connecté. au serveur via l'internet.

- Microcontrolleur arduino UNO pour alimenté et transformée les données mesurer par le capteur DHT22 vers le module Wifi ESP8266.

- Modem 4G LTE de Algerie telecom pour assurer l'accée al l'internet.

- Pc sous windows pour instaler le broker (implimentation de broker MQTT mosquitto 1.5.4).

- Smart phone pour afficher les données (téléphone mobile ex :samsung g2).

5.3 Réalisation de l'objet connecté :

❖ **Branchement le capteur DHT22 et le module Wifi ESP8266 avec arduino UNO**

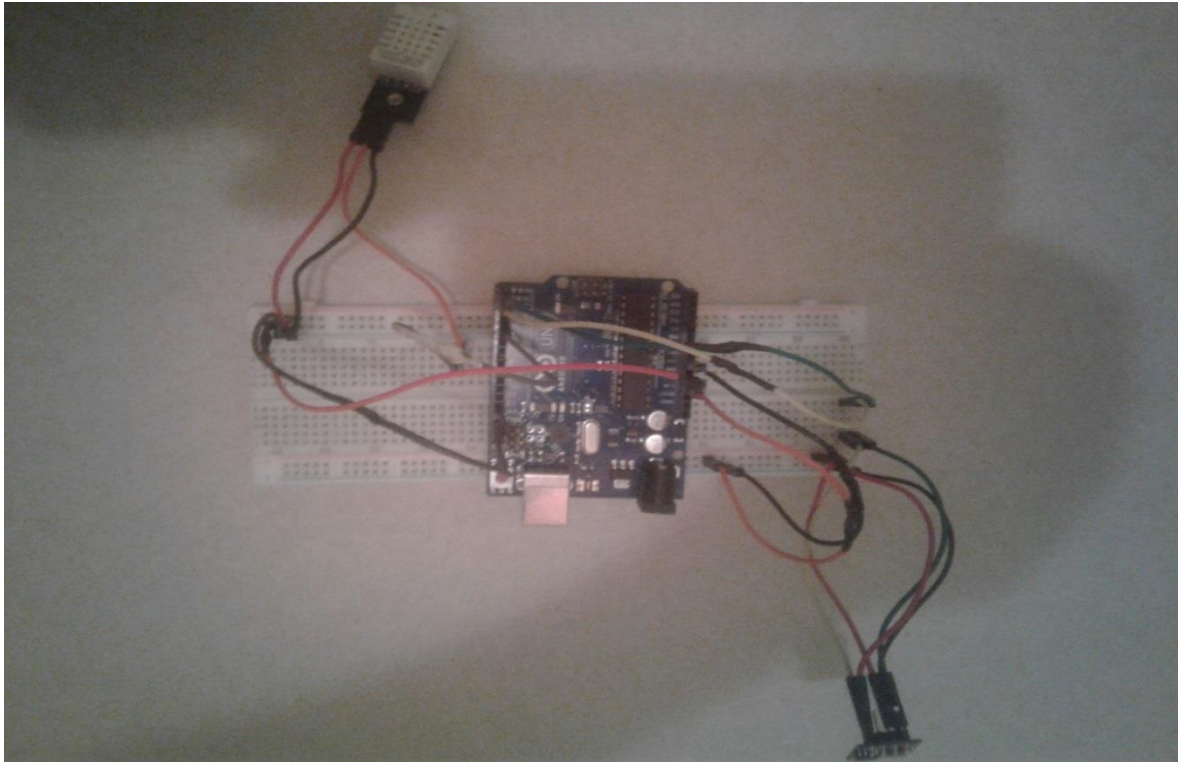


Figure 4.13 : *L'objet connecté en état éteint*

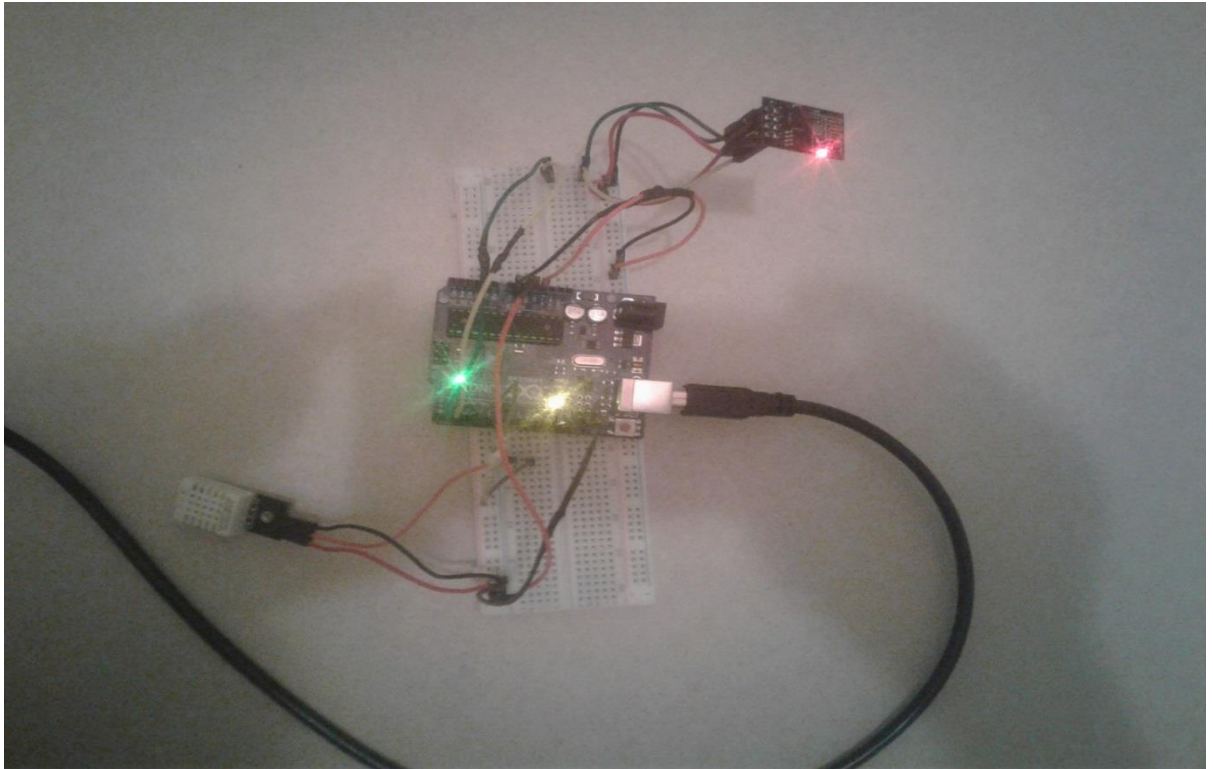


Figure 4.14 : L'objet connecté en état allumé

6 Les codes de diférant composant de système

Après avoir réalisé la partie hardware du système, nous passons à la programmation de l'objet ainsi qu'au le broker MQTT assurant la communication entre l'objet connecté et les smart phones.

6.1 Portion de code arduino pour l'objet connecté

✓ Code de connection au broker

```
PROJET | Arduino 1.8.8
Fichier Édition Croquis Outils Aide

PROJET

        status = WiFi.begin(WIFI_APP, WIFI_PASSWORD);
        delay(500);
    }

    //*****
    //***** connection au broker *****
    //*****
    //*****
    //*****
void reconnect() {
  // Loop until we're reconnected
  while (!client.connected()) {

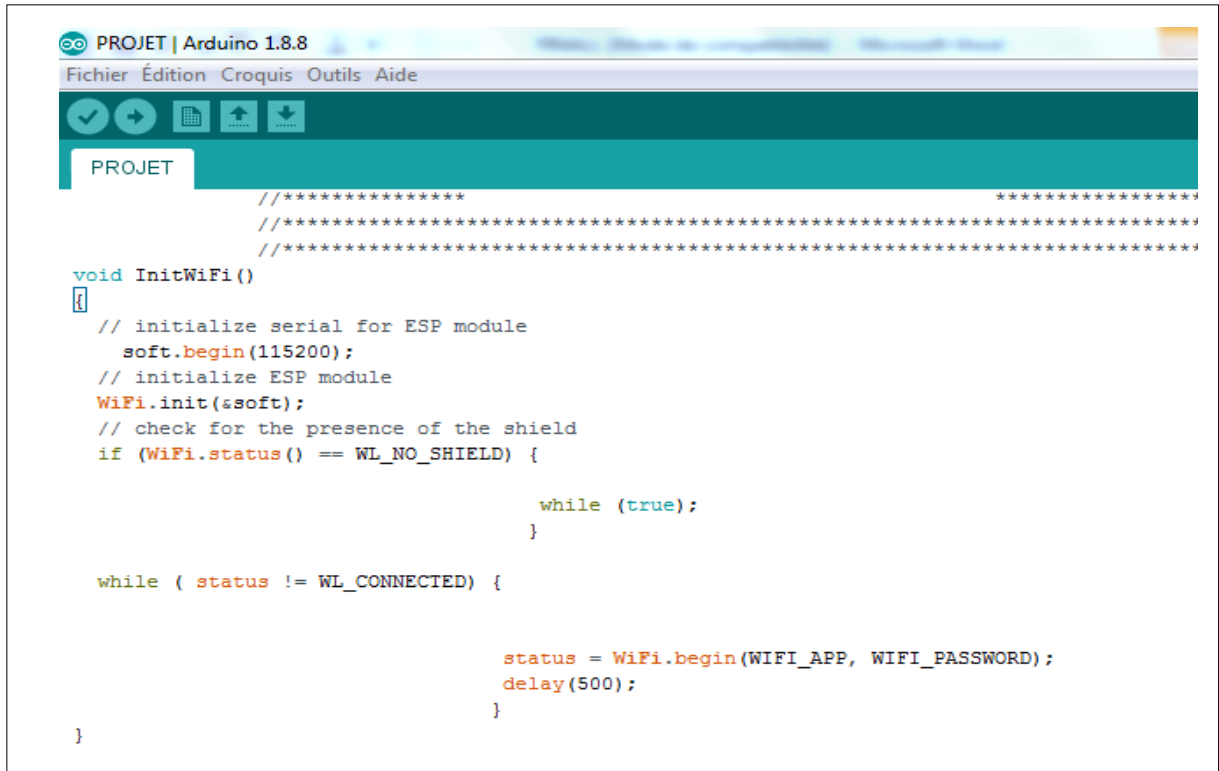
    if (client.connect("arduino",NULL, NULL)) {

    } else {

      delay(500);
    }
  }
}
```

Figure 4.15 : Code de connection au broker

✓ **Code d'initialisation module Wifi**



```
PROJET | Arduino 1.8.8
Fichier Édition Croquis Outils Aide

PROJET
//*****
//*****
//*****

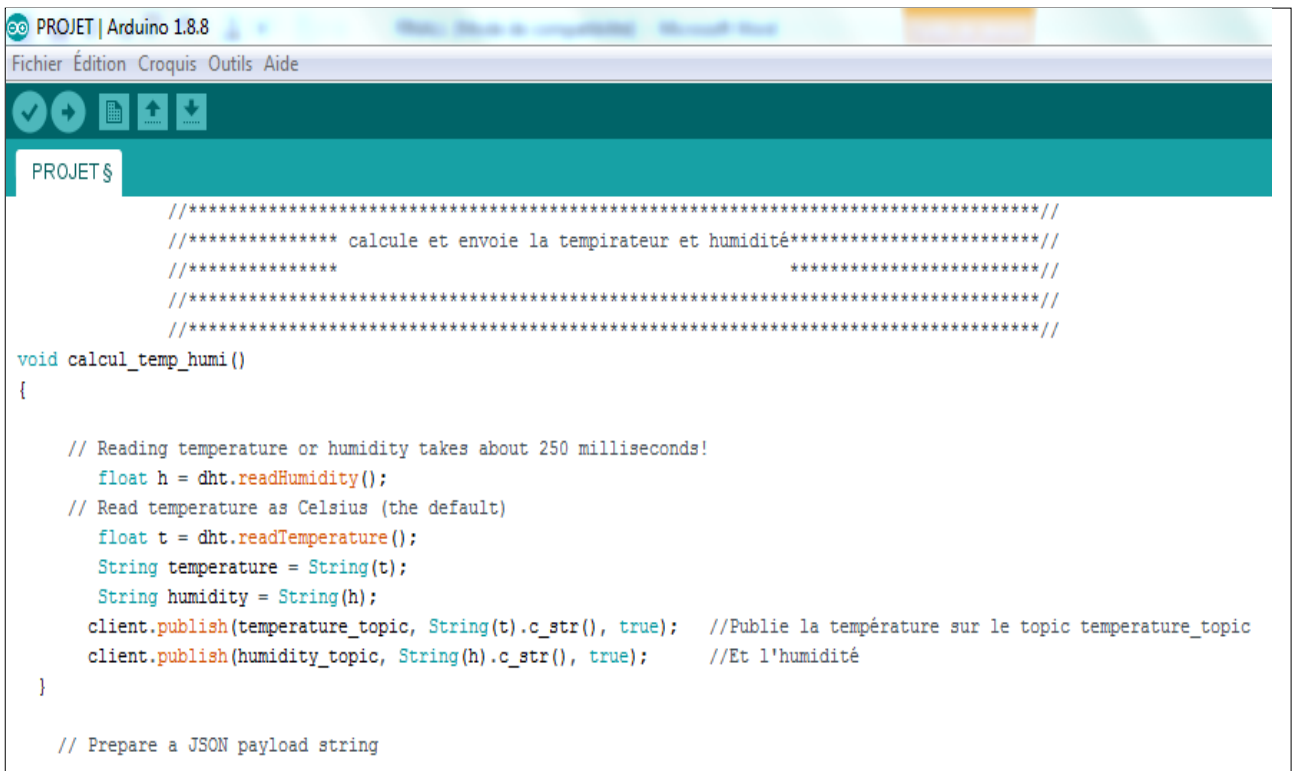
void InitWiFi()
{
  // initialize serial for ESP module
  soft.begin(115200);
  // initialize ESP module
  WiFi.init(soft);
  // check for the presence of the shield
  if (WiFi.status() == WL_NO_SHIELD) {
    while (true);
  }

  while ( status != WL_CONNECTED) {

    status = WiFi.begin(WIFI_APP, WIFI_PASSWORD);
    delay(500);
  }
}
```

Figure 4.16 : code d'initialisation module Wifi

✓ **Code de calcul et publier les topics**



```
PROJET | Arduino 1.8.8
Fichier Édition Croquis Outils Aide

PROJET$
//*****
//***** calcule et envoie la tempirateur et humidité*****
//*****
//*****
//*****

void calcul_temp_humi()
{
  // Reading temperature or humidity takes about 250 milliseconds!
  float h = dht.readHumidity();
  // Read temperature as Celsius (the default)
  float t = dht.readTemperature();
  String temperature = String(t);
  String humidity = String(h);
  client.publish(temperature_topic, String(t).c_str(), true); //Publie la température sur le topic temperature_topic
  client.publish(humidity_topic, String(h).c_str(), true); //Et l'humidité
}

// Prepare a JSON payload string
```

Figure 4.17: code de calcul et piblier les topics

6.2 Portion de code android pour les smart phones

On va dans un premier temps initialiser notre projet pour qu'il puisse communiquer avec le protocole MQTT. Pour cela, il faut :

Ajouter les dépendances dans le fichier gradle:

- Au tout début de votre fichier gradle :

```
repositories {  
    maven { url 'https://repo.eclipse.org/content/repositories/paho-snapshots/' }  
}
```

- Dans la partie dependencies de votre fichier gradle:

```
compile('org.eclipse.paho:org.eclipse.paho.android.service:1.0.3-SNAPSHOT') {  
    exclude module: 'support-v4'}  
compile 'org.eclipse.paho:org.eclipse.paho.client.mqttv3:1.0.3-SNAPSHOT'
```

Ajouter le service dans le Manifest entre les balises <Application> (si vous l'oubliez, vous n'aurez aucune erreur mais il vous sera impossible de vous connecter au broker et d'envoyer/recevoir des messages.

```
<service android:name="org.eclipse.paho.android.service.MqttService" ></service>
```

Ajouter les permissions dans le Manifest (en dessous de la balise fermante : </application>) :

```
<uses-permission android:name="android.permission.WAKE_LOCK" />
```

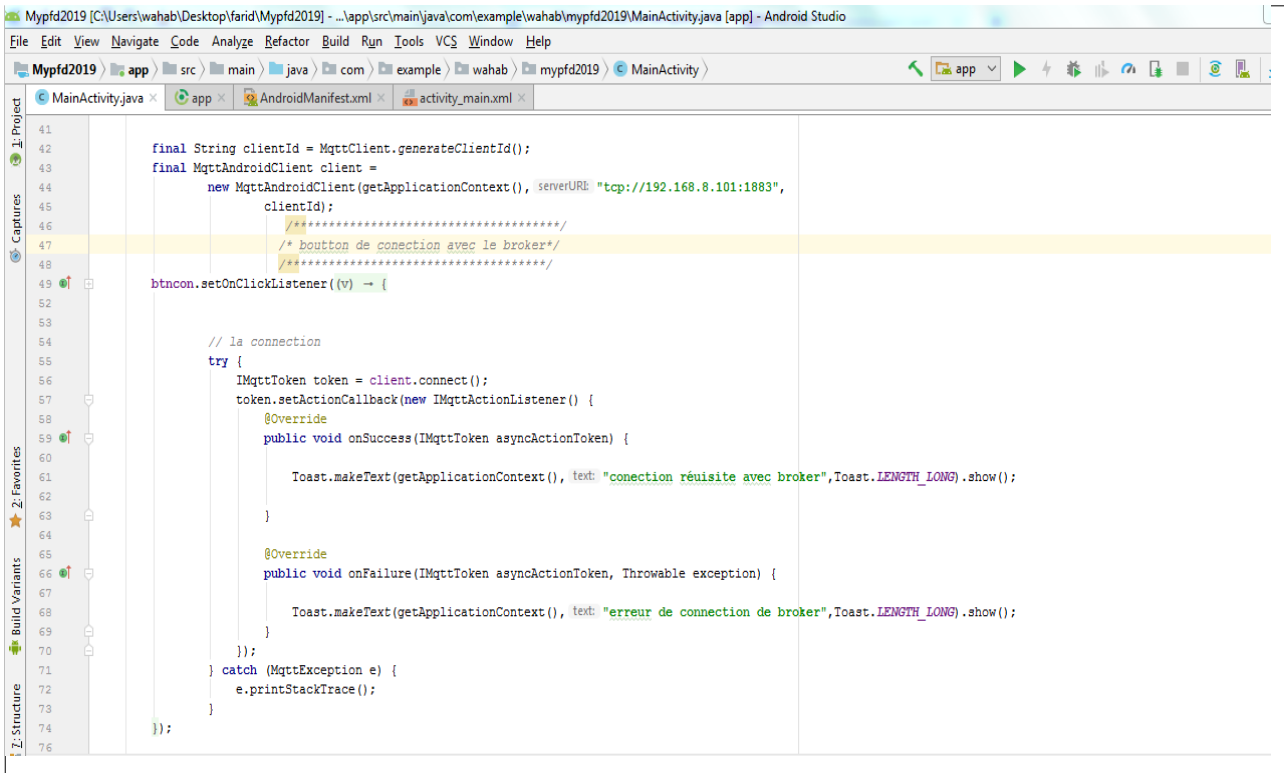
```
<uses-permission android:name="android.permission.INTERNET" />
```

```
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
```

```
<uses-permission android:name="android.permission.READ_PHONE_STATE" />
```

Notre projet est enfin prêt ! Nous allons maintenant créer la méthode qui va nous permettre de nous connecter au broker.

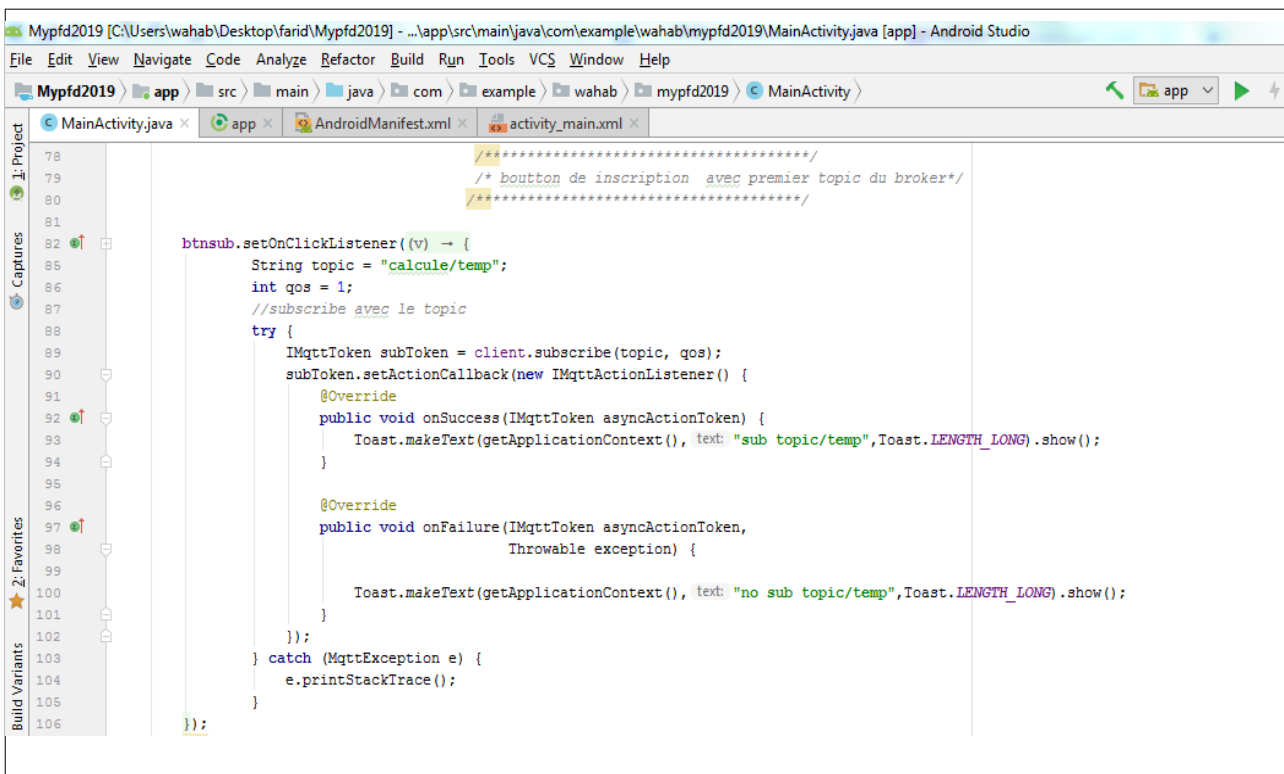
✓ Code de connexion au broker MQTT



```
41 final String clientId = MqttClient.generateClientId();
42 final MqttAndroidClient client =
43     new MqttAndroidClient(getApplicationContext(), serverURI: "tcp://192.168.8.101:1883",
44         clientId);
45
46
47 /* *****
48  * bouton de connexion avec le broker*
49  * *****
50
51 btncon.setOnClickListener((v) -> {
52
53
54     // la connexion
55     try {
56         IMqttToken token = client.connect();
57         token.setActionCallback(new IMqttActionListener() {
58             @Override
59             public void onSuccess(IMqttToken asyncActionToken) {
60
61                 Toast.makeText(getApplicationContext(), text: "conection réussite avec broker", Toast.LENGTH_LONG).show();
62
63             }
64
65             @Override
66             public void onFailure(IMqttToken asyncActionToken, Throwable exception) {
67
68                 Toast.makeText(getApplicationContext(), text: "erreur de connexion de broker", Toast.LENGTH_LONG).show();
69
70             }
71         });
72     } catch (MqttException e) {
73         e.printStackTrace();
74     }
75
76 });
```

Figure 4.18 : code de connexion au broker

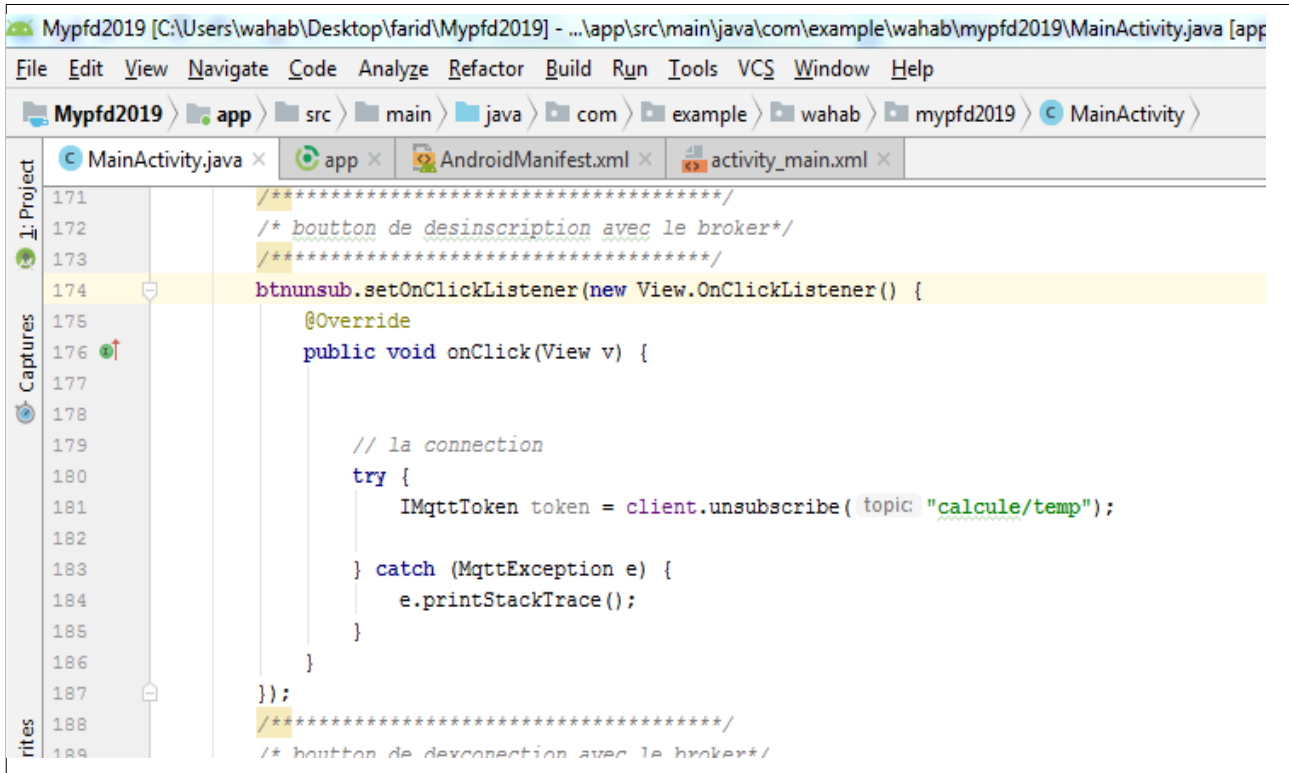
✓ Code d'inscription au topics



```
78
79 /* *****
80  * bouton de inscription avec premier topic du broker*
81  * *****
82
83 btnsub.setOnClickListener((v) -> {
84     String topic = "calcul/temp";
85     int qos = 1;
86     //subscribe avec le topic
87     try {
88         IMqttToken subToken = client.subscribe(topic, qos);
89         subToken.setActionCallback(new IMqttActionListener() {
90             @Override
91             public void onSuccess(IMqttToken asyncActionToken) {
92                 Toast.makeText(getApplicationContext(), text: "sub topic/temp", Toast.LENGTH_LONG).show();
93             }
94
95             @Override
96             public void onFailure(IMqttToken asyncActionToken,
97                 Throwable exception) {
98
99                 Toast.makeText(getApplicationContext(), text: "no sub topic/temp", Toast.LENGTH_LONG).show();
100
101             }
102         });
103     } catch (MqttException e) {
104         e.printStackTrace();
105     }
106 });
```

Figure 4.19 : code d'inscription au topic

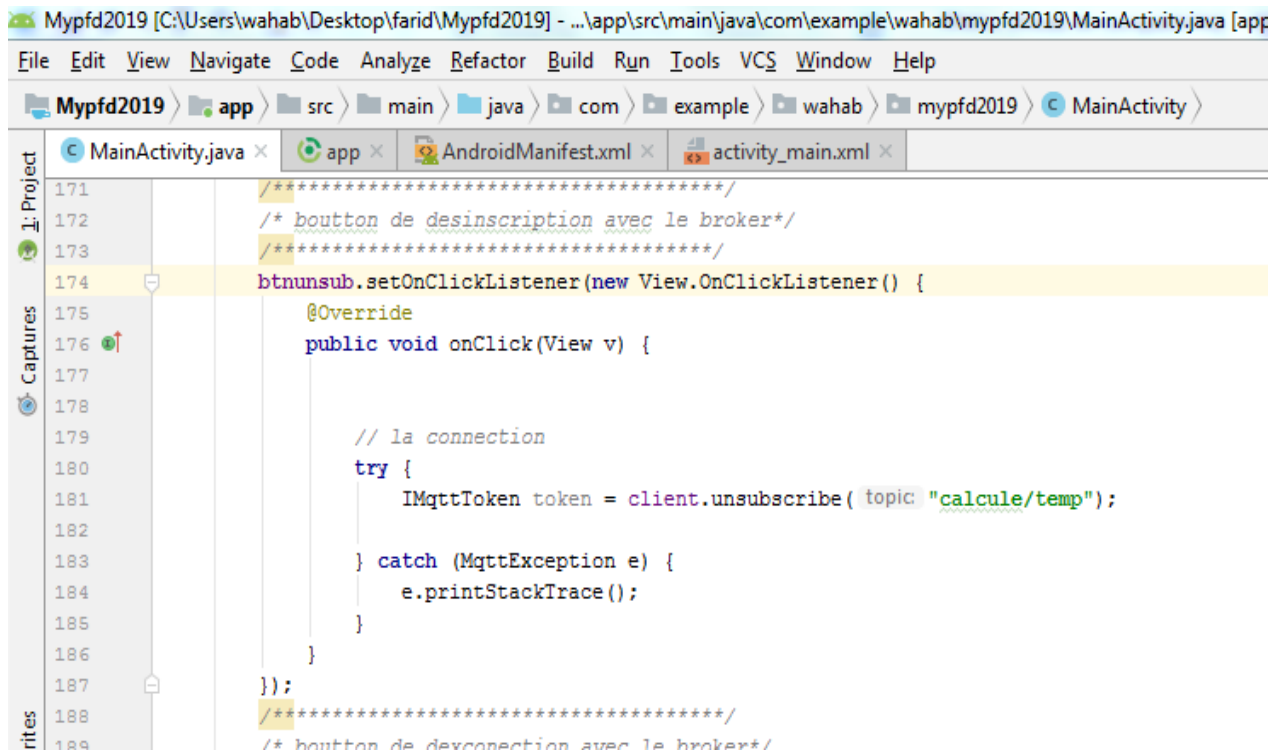
✓ **Cod de desinscription au topic**



```
171 //*****  
172 /* bouton de desinscription avec le broker*/  
173 //*****  
174 btnunsub.setOnClickListener(new View.OnClickListener() {  
175     @Override  
176     public void onClick(View v) {  
177  
178  
179         // la connection  
180         try {  
181             IMqttToken token = client.unsubscribe( topic "calcul/temp");  
182  
183         } catch (MqttException e) {  
184             e.printStackTrace();  
185         }  
186     }  
187 });  
188 //*****  
189 /* bouton de deconnection avec le broker*/
```

Figure 4.20 : code de désinscription au topic

✓ **Code déconnexion au broker**



```
171 //*****  
172 /* bouton de desinscription avec le broker*/  
173 //*****  
174 btnunsub.setOnClickListener(new View.OnClickListener() {  
175     @Override  
176     public void onClick(View v) {  
177  
178  
179         // la connection  
180         try {  
181             IMqttToken token = client.unsubscribe( topic "calcul/temp");  
182  
183         } catch (MqttException e) {  
184             e.printStackTrace();  
185         }  
186     }  
187 });  
188 //*****  
189 /* bouton de deconnection avec le broker*/
```

Figure 4.21 : code source de déconnexion au broker

✓ **Description de l'interface application mobile**

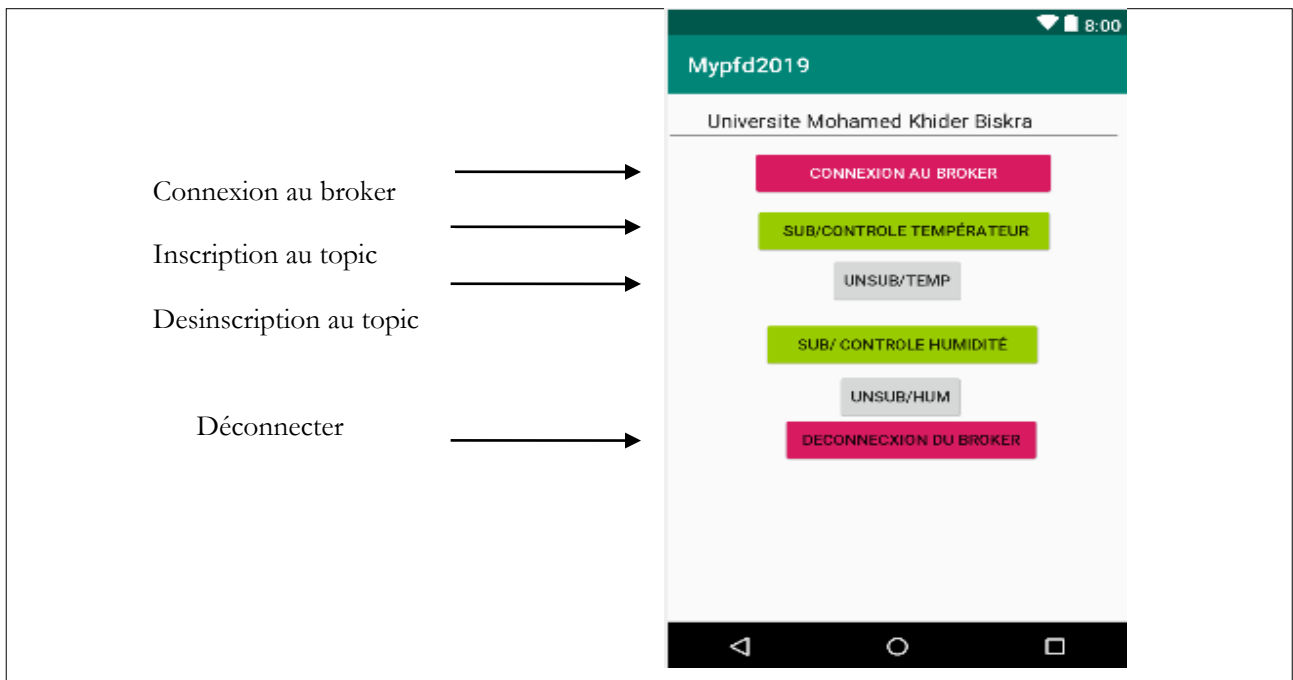


Figure 4.22 : *discription de l'interface de l'application mobile*

6.3 Discription coté de broker

Eclipse Mosquitto est un courtier de messages open source (licence EPL / EDL) qui implémente les versions de protocole MQTT 5.0, 3.1.1 et 3.1. Mosquitto est léger et convient à une utilisation sur tous les appareils, des ordinateurs et microcontroleur.. etc.

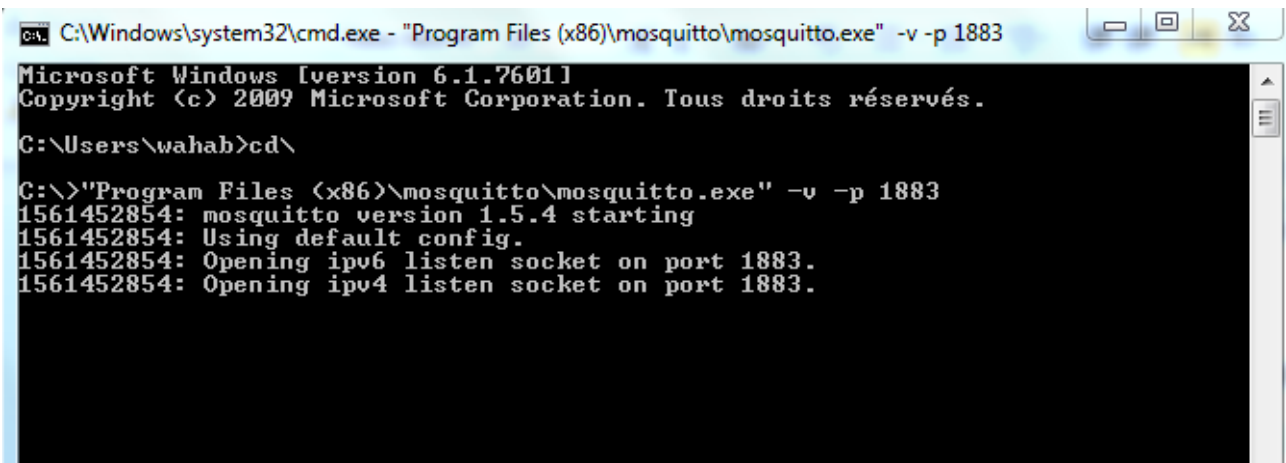
Le projet Mosquitto fournit également une bibliothèque C pour la mise en œuvre des clients MQTT, ainsi que les très populaires clients MQTT mosquitto_pub et mosquitto_sub. Mosquitto fait partie de la fondation Eclipse et est un projet de iot.eclipse.org .

Dans notre projet nous avons utiliser mosquitto 1.5.4 l'implimentation de MQTT 3.1.1.

Il suffit de télécharger l'exécutable sur le site officiel (<https://mosquitto.org/>) et de l'installer et incluse les fichier SSL pour le cryptage des données transferer .

Mosquitto utilise par défaut le port 1883 pour la communication externe, sa fenêtre principale est une fenêtre Invite de commande (CMD) (voir la figure 4.23).

- -v : mode verbose.
- -p 1883 : on va utiliser le port de l'ordinateur 1883.



```
C:\Windows\system32\cmd.exe - "Program Files (x86)\mosquitto\mosquitto.exe" -v -p 1883
Microsoft Windows [version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. Tous droits réservés.

C:\Users\wahab>cd\

C:\>"Program Files (x86)\mosquitto\mosquitto.exe" -v -p 1883
1561452854: mosquitto version 1.5.4 starting
1561452854: Using default config.
1561452854: Opening ipv6 listen socket on port 1883.
1561452854: Opening ipv4 listen socket on port 1883.
```

Figure 4.23 : *mosquitto broker démarer*

7 Résultat

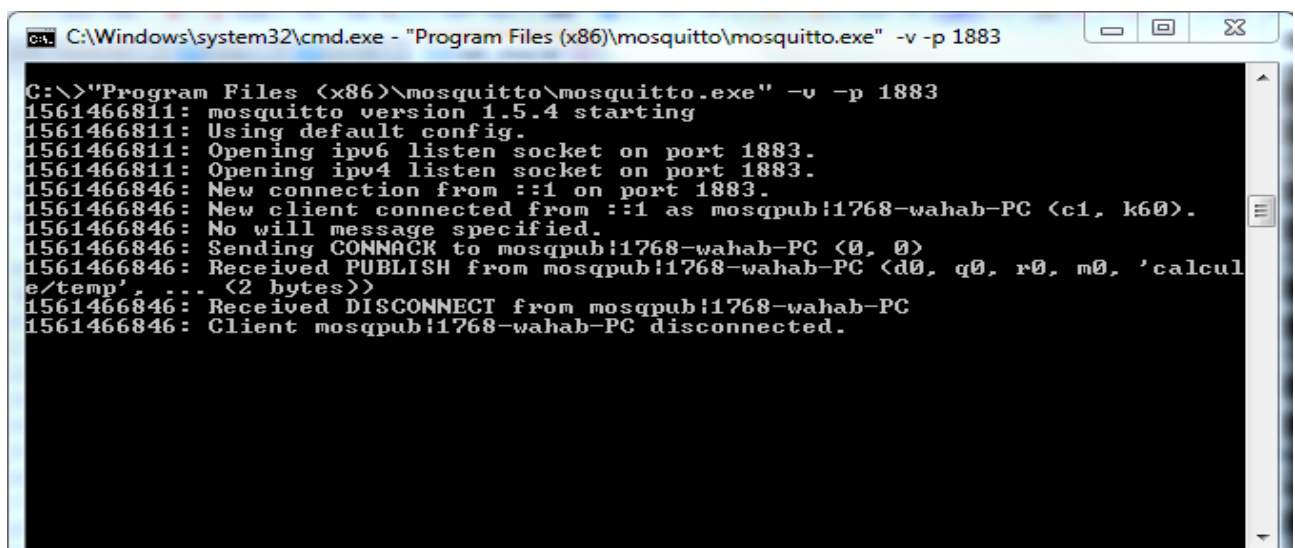
Nous allons à présent voir le résultat tant attendu après un travail acharné entre le monde d'Android, d'Arduino et le protocole MQTT géré par le broker qui tourne sur notre ordinateur.

- Lancez le broker sur notre ordinateur.
- Lancez l'application Android sur notre smart phone .
- Démarrez notre Arduino connecté à votre réseau internet.

En fin commencer de l'exploitation de l'application de contrôle de température et humidité.

7.1 Interface des l'application pendant l'exploitation de système

- ✓ **Acquisition des données** (*le broker recevoir une publication des donnéeé*)



```
C:\Windows\system32\cmd.exe - "Program Files (x86)\mosquitto\mosquitto.exe" -v -p 1883
C:\>"Program Files (x86)\mosquitto\mosquitto.exe" -v -p 1883
1561466811: mosquitto version 1.5.4 starting
1561466811: Using default config.
1561466811: Opening ipv6 listen socket on port 1883.
1561466811: Opening ipv4 listen socket on port 1883.
1561466846: New connection from ::1 on port 1883.
1561466846: New client connected from ::1 as mosqpub:1768-wahab-PC (c1, k60).
1561466846: No will message specified.
1561466846: Sending CONNACK to mosqpub:1768-wahab-PC (0, 0)
1561466846: Received PUBLISH from mosqpub:1768-wahab-PC (d0, q0, r0, m0, 'calcul
e/temp', ... (2 bytes))
1561466846: Received DISCONNECT from mosqpub:1768-wahab-PC
1561466846: Client mosqpub:1768-wahab-PC disconnected.
```

Figure 4.24 : *acquisition des données*

(le broker recevoir une inscriptio au topic et revoiyer les données vers le client)

```
C:\Windows\system32\cmd.exe - "Program Files (x86)\mosquitto\mosquitto.exe" -v -p 1883
1561466846: Received DISCONNECT from mosqpub!1768-wahab-PC
1561466846: Client mosqpub!1768-wahab-PC disconnected.
1561466920: New connection from ::1 on port 1883.
1561466920: New client connected from ::1 as mosqsub!3800-wahab-PC (c1, k60).
1561466920: No will message specified.
1561466920: Sending CONNACK to mosqsub!3800-wahab-PC (0, 0)
1561466920: Received SUBSCRIBE from mosqsub!3800-wahab-PC
1561466920:   calcul/temp (QoS 0)
1561466920: mosqsub!3800-wahab-PC 0 calcul/temp
1561466920: Sending SUBACK to mosqsub!3800-wahab-PC
1561466980: Received PINGREQ from mosqsub!3800-wahab-PC
1561466980: Sending PINGRESP to mosqsub!3800-wahab-PC
1561467023: New connection from ::1 on port 1883.
1561467023: New client connected from ::1 as mosqpub!6896-wahab-PC (c1, k60).
1561467023: No will message specified.
1561467023: Sending CONNACK to mosqpub!6896-wahab-PC (0, 0)
1561467023: Received PUBLISH from mosqpub!6896-wahab-PC (d0, q0, r0, m0, 'calcul
e/temp', ... (2 bytes))
1561467023: Sending PUBLISH to mosqsub!3800-wahab-PC (d0, q0, r0, m0, 'calcul
e/temp', ... (2 bytes))
1561467023: Received DISCONNECT from mosqpub!6896-wahab-PC
1561467023: Client mosqpub!6896-wahab-PC disconnected.
1561467041: Received PINGREQ from mosqsub!3800-wahab-PC
1561467041: Sending PINGRESP to mosqsub!3800-wahab-PC
```

Figure 4.25 : l'inscription et l'envoi des données par le broker

✓ Consultation des données

- Connexion au broker

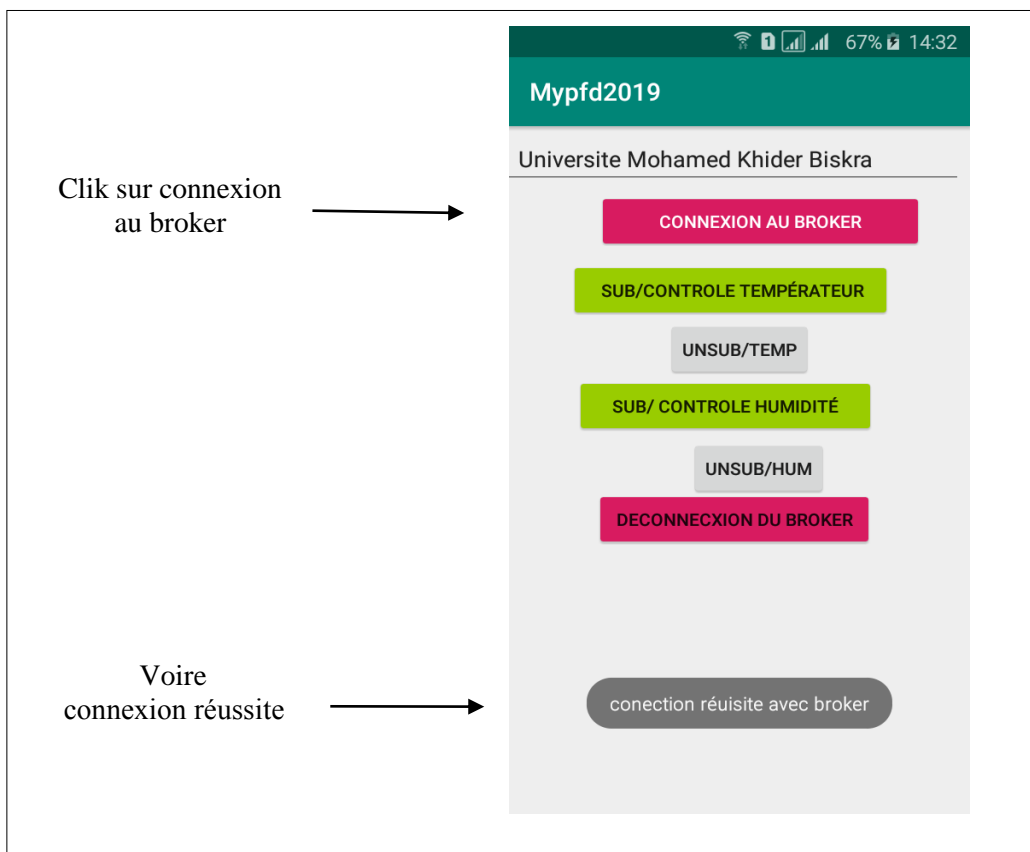


Figure 4.26 : connexion au broker par smart phone

- Inscription au topic

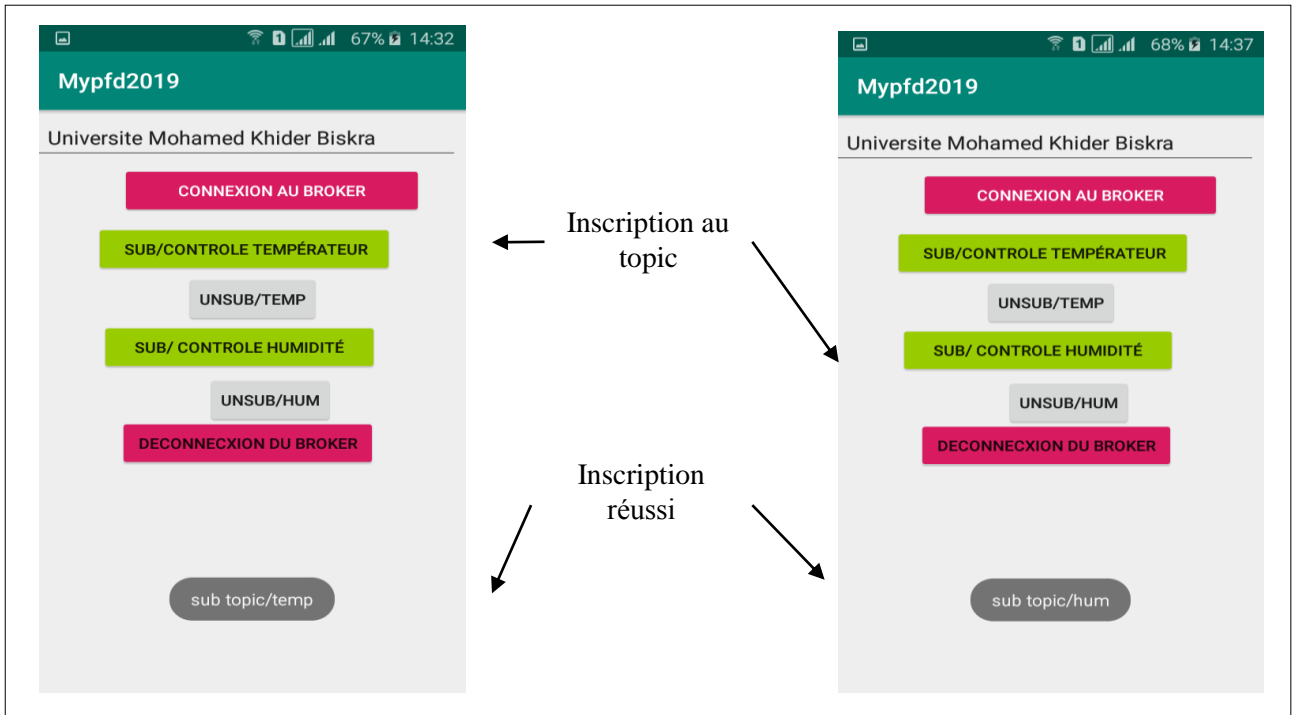


Figure 4.27 : inscription au topic par smart phone

- Affichage du résultat

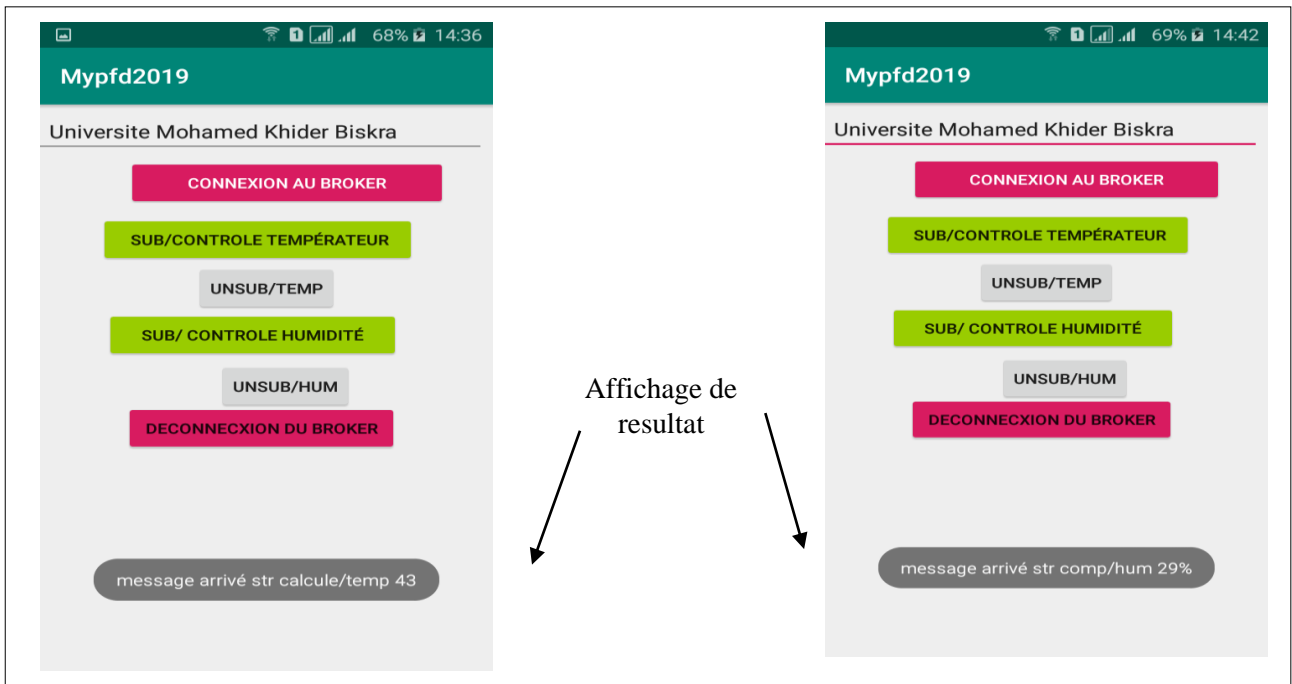


Figure 4.28 : affichage du résultat avec smart phone

8 Conclusion

Dans ce dernier chapitre, nous avons exposé une étude de cas, en essayant de présenter les notions essentielles relatives à l'implémentation de notre Projet. Nous avons présenté le maximum des concepts relatifs à l'environnement de développement ainsi que la présentation de notre application.

Chapitre 5 : Comparison avec RESTful

Chapitre 5

Comparison avec RESTful

1. Introduction

Comme illustre dans l'introduction générale les dispositifs IoT peuvent être intégrés dans le Web à l'aide des ressources RESTful pour exposer la fonctionnalité des dispositifs IoT [2]. REST est un style architectural pour la construction du Web et d'autres articles Soft- dans réseau [83]. La particularité principale de cette architecture est que la partie serveur (l'API) et la partie client communiquent sans que le client ne connaisse la structure et le contenu des informations stockées sur le serveur. La seule chose que les deux parties de l'application connaissent est le média qu'elles utilisent pour communiquer.

Les applications REST s'appuient sur les verbes fournis par le protocole HTTP. Ce sont des mots-clés qui définissent l'action que l'on souhaite effectuer sur une ressource. Tout ce que REST rend disponibles doivent être disponibles vu en termes de ressources. Les ressources peuvent être chose dans REST Partout qui peut être nommé et identifié. Par exemple, le capteur de température au humidité est une ressource qui peut fournir des informations sur la température au la humidité de l'arrondi sur- où il est installé.

2. les éléments essentiels de l'architecture RESTful

2.1 identifiant de la ressource

Les ressources sont identifiées à l'aide identifiant de ressource ou URI. Une ressource peut donner de l'information unique ou la collecte d'informations. Par exemple, « dispositifs » est une ressource qui donne les informations sur la liste des périphériques. Considérant que, « dispositif » est une source d'information unique.

Les ressources « périphériques » identifiés par URI, `http://192.168.0.1/devices`

où `http://` est le protocole utilisé, `192.168.0.1` est l'adresse IP de l'hôte et `/` périphériques est l'URL de la ressource.

De même, les applications installées dans l'appareil est également une ressource et il est identifié par URI, `http://192.168.0.1/device/{device_id}/applications`.

2.2 Interface uniforme

Ressources dans le reste devrait avoir l'interface uniforme avec l'ensemble fixe de

Comparison avec RESTful

méthodes HTTP et de représentation qu'elle retourne. REST utilise des méthodes HTTP GET, POST, PUT, SUPPR pour effectuer des opérations sur les ressources.

2.3 la représentation

Il est défini par un type de support utilisé par le destinataire pour traiter la représentation des ressources [83] exp HTML,XML,JSON.

2.4 Sans état (Statelessness)

Ce qui signifie que le serveur ne conserve pas les enregistrements des informations sur la demande faite par les clients. Chaque demande est une demande unique pour le serveur et le client doit contenir toutes les informations nécessaires

3. Comparaison entre le REST et MQTT

REST est un style architectural, alors que MQTT est protocole de messagerie basé sur publish/subscribe simple et léger, il semble qu'il n'y a pas de similitude entre eux. Cependant, l'objectif du REST et à MQTT est le même, qui est l'échange d'informations entre les points d'extrémité du réseau. selon[83] REST est constitué d'éléments qui sont décrits dans la section 2. Tout en architecture REST fonctionne autour des ressources. cela vous dit que la ressource est l'entité principale REST par contre dans le MQTT le client joue l'entité principale (subscriber/publisher).

Les sous-sections suivantes traitent de la comparaison entre les éléments de REST et éléments MQTT.

3.1. Client-Serveur / Publisher-Subscriber

Dans REST le client est référencé en tant que demandeur de service et le serveur est référencé en tant que fournisseur de service correspondant à la demande. Par contre, MQTT le subscriber et le publisher sont des clients. Dans MQTT le publisher et le subscriber sont séparés, ils ne ont pas besoin de connaître l'existence de l'autre. En revanche, le client REST doit connaître l'adresse du serveur pour envoyer la demande et obtenir la réponse.

Rest	MQTT
Ressources	client MQTT
URI	Les sujets
Serveur client	Editeur-abonné
Demande de réponse	Publish-subscribe

Table 5.1 : Comparaison de REST et MQTT

3.2 URI / MQTT Sujet (topic)

Dans REST, les ressources sont identifiées à l'aide d'URI, qui est la localisation du serveur et en plus la ressource disponible sur le serveur. Dans MQTT, le sujet est joint au message publié et l'abonnement ce fait sur le même sujet pour obtenir ce message.

3.3 Requête -réponse / publish-subscribe

REST fonctionne sur les principes de HTTP, et il utilise modèle requête-réponse. HTTP requête-réponse synchroniser, à la différence de MQTT L'inscription et la publication qui est asynchrone.

4. Conclusion

Requête -réponse de HTTP ne convient pas pour la communication avec le dispositif dont l'adresse est inconnue. Pour client-serveur requête -réponse ils doivent connaître l'existence de l'autre pour la communication. Par conséquent, dans le réseau où NAT est utilisé et les dispositifs IoT derrière NAT, les ressources REST exposées en tant que dispositifs IoT et alors ne seront pas accessibles.

La mise en œuvre REST fonctionne bien sur le réseau local. Mais les dispositifs IoT sont distribués à travers le réseau étendu. réseau étendu relie plusieurs réseaux qui utilisent NAT (Network Address Translation). En MQTT il n'y a pas de problème avec les clients derrière un NAT, car le client MQTT communique avec le Broker qui peut être consulté sur Internet. En outre, le client MQTT envoie le premier message au Broker comme un message CONNECT, connexion ouvert du Broker au client et vice-versa. En outre, dans MQTT, le subscriber et le publisher sont séparés dans le temps. Cela signifie subscriber et publisher n'ont pas besoin de l'adresse de subscriber, ils n'ont pas besoin de Broker dans le même temps et l'exécution de l'opération dans l'éditeur ne perturberont pas l'opération dans publisher. Par conséquent, ils sont totalement indépendants les uns des autres.

C'est la raison pour laquelle nous avons choisi ce protocole pour l'implantation et la réalisation d'une application de contrôle de température et humidité dans IoT.

Conclusion générale

Conclusion général

Ce travail nous a permis de faire un état de l'art sur différents domaines de recherches, à savoir l'IoT, l'implémentation du protocole MQTT. En outre, la mise en œuvre d'un système d'IoT pour le contrôle à distance de la température et l'humidité nous a permis d'apprendre et d'exploiter plusieurs plateformes, outils et langages de développement, entre autres les plateformes Arduino et Android.

La particularité de ce projet est la réalisation physique d'un système d'IoT qui présente la première expérience pour nous. Plusieurs concepts sont acquis, qui concernent non seulement la partie software mais aussi la partie hardware dont on a réalisé un système d'IoT pour gérer les dispositifs.

La solution proposée dans ce manuscrit comporte trois parties : matérielle, software, protocoles de communication.

- Ce qui concerne la partie matérielle, on a réalisé un objet connecté placé dans l'environnement ciblé pour le captage.
- Ce qui concerne la partie software, nous avons créé deux applications dans différentes plateformes
 - ✓ Une application mobile sous Android pour la visualisation des données captées à distance (Internet)..
- Ce qui concerne la partie des protocoles de communication, on a utilisés les protocoles suivants :
 - ✓ Wifi : pour la communication entre l'objet connecté et le Broker.
 - ✓ MQTT : pour la communication entre le serveur et les objets connectés.

BIBLIOGRAPHIE

Bibliographie

- [1] DAVE, Evans. *L'Internet des objets Comment l'évolution actuelle d'Internet transforme-t-elle le monde ?*. Avril 2011, 12 p. (Cisco Internet Business Solutions Group (IBSG)).
- [2] D. Guinard, V. Trifa, and E. Wilde, "A resource oriented architecture for the web of things," in *Internet of Things (IOT)*, 2010. IEEE, 2010, pp. 1–8.
- [3] B. Emmerson, « M2m: Internet de 50 milliards d'appareils, » *WinWin Magazine*, vol. 1, pp. 19-22, 2010.
- [4] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," *Computer networks*, vol. 54, no. 15, pp. 2787–2805, 2010.
- [5] « Version 3.1.1 mqtt, plus errata 01. Edité par Andrew Banks et Rahul Gupta 10 Décembre 2015, » OASIS standard Incorporant approuvé Errata 01. <http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/errata01/os/mqtt-v3.1.1-errata01-os-complete.html>, déc 2015 Dernière version: <http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/mqtt-v3.1.1.html>.
- [6] DAVE, Evans. *L'Internet des objets Comment l'évolution actuelle d'Internet transforme-t-elle le monde ?*. Avril 2011, 12 p. (Cisco Internet Business Solutions Group (IBSG)).
- [7] E. Dobin et al. *Analyse et perspectives de l'Internet des Objets*. Juillet 2011, 132 p. (Les Cahiers de la RFID - Collection CITC)
- [8] SYNOX Group. *4 choses à savoir sur l'Internet des objets*. [en ligne]. Disponible sur : < <https://www.synox.io/4-choses-a-savoir-sur-linternet-des-objets/> > (Consulté le 02/02/2019)
- [9] MELITI Nedjema. *Architecture Basée Agents pour le diagnostic d'un système d'IoT (Internet of Things)*. Mémoire de Master académique, Architectures Distribuées. Oum-El-Bouaghi : Université Larbi Ben M'hidi Oum-El-Bouaghi, 2017, 111 p.
- [10] SUNDMAEKER, Halard, GUILLEMIN, Patrick, FRIESS, Peter, WOELFFLÉ, Sylvie. *Vision and Challenges for Realising the Internet of Things*. European. Luxembourg : Publications Office of the European Union, Mars 2010, 229 p. (ISBN 978-92-79-15088-3)
- [11] Livre Blanc . PRÉPARER LA RÉVOLUTION DE L'INTERNET DES OBJETS Document N° 1 – Une cartographie des enjeux. 7 novembre 2016, (PDF, 46 p)
- [12] ITU. *Internet of Things Global Standards Initiative*. [en ligne]. Disponible sur : <<https://www.itu.int/en/ITU-T/gsi/iot/Pages/default.aspx>> (Consulté le 03/02/2019)
- [13] IEEE. *Internet of Things*, [en ligne]. Disponible sur : <<https://iot.ieee.org/definition.html> > (Consulter le 03/02/2019)
- [14] CHALLAL, Yacine. *Sécurité de l'Internet des Objets : vers une approche cognitive et*

Bibliographie

systemique. Thèse de doctorat : Technologies de l'Information et des Systèmes. France : Juin 2012, 78 p.

[15] BABAKHOUYA, Abdelaziz, CHALLAL, Yacine, BOUABDALLAH, Abdelmadjid, GHAROUT, Said. *Securing Distance Vector Routing Protocols for Hybrid Wireless Mesh Networks*. SAR- SSI 2010.

[16] COULON Alain. *L'Internet des Objets Un gisement à exploiter*. La Lettre d'ADELI n°78 – Hiver 2010. (PDF, 5 p [page 26 à 30])

[17] Intel. *Applications pour l'internet des objets dans l'industrie*. [en ligne]. Disponible sur : <https://www.intel.fr/content/www/fr/fr/internet-of-things/industry-solutions.html> (Consulté le 02/02/2019)

[18] AUTOMATION&PLC KNOWLEDGE CENTER. *Les domaines d'application de l'IoT et du machine to machine*. [en ligne]. Disponible sur : <https://www.automation-sense.com/blog/blog-objets-connectes/les-domaines-d-application-de-l-iot-et-du-machine-to-machine.html>

[19] Objeko.com. Disponible sur : < <https://www.objeko.com/selection-meilleur-objetconnecte-2016-1777/> > (consulté le 02/02/2019)

[20] Objeko.com. Disponible sur : <https://www.objeko.com/selection-meilleur-objet-connecte-2016-1777/> (Consulté le 09/02/2019)

[21] amazon. Disponible sur <<https://www.amazon.fr/gp/product/B01KV0JCOS?ie=UTF8&linkCode=as2&camp=1634&creative=6738&tag=objeko21&creativeASIN=B01KV0JCOS>> (Consulté le 09/02/2019)

[22] Objeko.com. [en ligne]. Disponible sur : <<https://www.amazon.fr/gp/product/B01LR8PG3O?ie=UTF8&linkCode=as2&camp=1634&creative=6738&tag=objeko-21&creativeASIN=B01LR8PG3O>>. (Consulté le 09/02/2019)

[23] DARTY. Disponible sur : <<https://www.darty.com/darty-et-vous/high-tech/objets-connectes/les-objets-connectes-au-jardin-vous-la-main-verte>> (Consulté le 09/02/2019)

[24] CNET. Disponible sur : <<http://www.cnetfrance.fr/special/objets-connectes/4000237247.htm>> (Consulté le 09/02/2019)

[25] LeClubGQ. *Les 10 objets High-Tech les plus attendus en 2018*. [en ligne]. Disponible sur : <<http://www.gqmagazine.fr/lifestyle/high-tech/diaporama/les-10-objets-high-tech->

Bibliographie

[les-plus- attendus-en-2018/48067#un-deux-ou-trois-iphone-x-1](#) >. (Consulté le 20/03/2019)

[26] TechMoka. [en ligne]. Disponible sur : < <http://techmoka.com/lentilles-connectees-samsung/> >. (Consulté le 20/03/2019)

[27] IoT Agenda. [en ligne]. Disponible sur : < <http://internetofthingsagenda.techtarget.com/definition/microcontroller> > (Consulté le 10/02/2019)

[28] ADJIBA, Brahim, CHALGHOUM, Abdelmonaim. *Commande des équipements électriques par microcontrôleurs "Simulations et Réalisations". Mémoire de Master académique, Commande Électrique. d'El-Oued: Université Echahid Hamma Lakhdar d'El-Oued, 2015, 51p.*

[29] Jlassi, Khaled. *Microprocesseurs et Microcontrôleurs Représentation de l'information en numérique.*

Tunis : Université virtuelle de Tunis , 2008, 18p.

[30] Techno-Science. *Microcontrôleur.* [en ligne]. Disponible sur : < <http://www.techno-science.net/?onglet=glossaire&definition=6737> > (Consulté le 11/02/2019)

[31] SIMON, Monk. *Mouvement lumière et son avec ARDUINO et RASPBERRY PI.* Paris : Groupe Eyrolles, 2016, 325 p. (ISBN : 978-2-212-11807-0)

[32] ARDUINO. *Le site officiel d'arduino.* Disponible sur : < <https://www.arduino.cc/> >. (Consulté le 12/02/2019)

[33] CHRISTIAN, Tavernier. *Raspberry Pi A+ B+ et 2 Prise en main et premières réalisations.* 2^{ème} édition . Paris : Dunod, 2015, 333 p. (ISBN 9782100742639)

[34] Raspberry Pi France. [en ligne]. Disponible sur : < <http://www.raspberrypi-france.fr/> >. (Consulté le 13/02/2019)

[35] Supercondensateur. [en ligne]. Disponible sur : < <http://www.supercondensateur.com/breadboard-et-simulateurs-de-circuits-electroniques> >. (Consulté le 14/02/2019)

[36] Velleman. *PLAQUES DE MONTAGE RAPIDE - HAUTE QUALITE.* [en ligne]. Disponible sur : < <https://www.velleman.eu/products/view/?id=40574> >. (Consulté le 14/02/2019)

[37] ARDUINO. *PROTO SHIELD REV3 (UNO SIZE).* [en ligne]. Disponible sur : < <https://store.arduino.cc/proto-shield-rev3-uno-size> >. (Consulté le 14/02/2019)

[38] La Boutique. [en ligne]. Disponible sur : < <https://boutique.semageek.com/fr/> >. (Consulté le 14/02/2019)

[39] TipTopBoards. *SHIELD DE PROTOTYPAGE ARDUINO UNO AVEC PLAQUETTE D'ESSAIS.* [en ligne]. Disponible sur : < <http://tiptopboards.com/72-shield-de-prototypage-arduino-avec-plaquette-d-essais.html> >. (Consulté le 14/02/2019)

Bibliographie

- [40] GO TRONIC. Robotique et composants électroniques. [en ligne]. Disponible sur : < <https://www.gotronic.fr/cat-capteurs-1114.htm> >. (Consulté le 17/02/2019)
- [41] SANOGO, Yacouba.. [15] MELITI Nedjema. Conception et fabrication de capteurs et de leur techniques d'interrogation por des applications dans Les domaines de la santé et de l'environnement. Thèse de doctorat. Physique. France : laboratoire de photonique quantique et moléculaire, 2013, 180 p.
- [42] ASCH, Georges, POUSSERY Bernard. LES CAPTEURS EN INSTRUMENTATION INDUSTRIELLE. 8^e éd. Dunod : 1983, 2006, 2010, 2017, 919 p. (ISBN : 978-2-10-076020-6). Disponible sur : <https://www.dunod.com/sites/default/files/atoms/files/Feuilletage_57.pdf>. [PDF].(Consulté le 20/03/2019)
- [43] Letmeknow. [en ligne]. Disponible sur : < <https://letmeknow.fr/shop/6-arduino> >. (Consulté le 01/03/2019)
- [44] Wikipédia. [en ligne]. Disponible sur : < <https://fr.wikipedia.org/wiki/Actionneur> >. (Consulté le 21/02/2019)
- [45] MESTERS, Jean-Paul, COLLIGNON, Patrick. Monter son réseau Wi-Fi ou Ethernet en un jour Spécial particuliers, indépendants et TPE. Groupe Eyrolles, 2005, 58 p. (ISBN : 2-7464-0493-1)
- [46] New Technologie. Les types de réseau. [en ligne]. Disponible sur : < <https://sites.google.com/site/wwwnewtechnologiecom/reseau/reseau-informatique/terminologie-des-rseaux> >. (Consulté le 19/02/2019)
- [47] BASTIEN, Jorge, BASTIEN, Lacroix, PROUX, Alexander. Les protocoles réseau de l'Internet des Objets -. Disponible sur : < <https://www.overleaf.com/articles/protocole-iot-vulnerabilites-connues/sgbsqjhmtx.pdf> >. [PDF]. (Consulté le 26/02/2019)
- [48] PEYRARD, Fabrice, CONCHON, Emmanuel. Technologie sans contact NFC embarquée – Aspect protocoloraires. Toulouse : Institut de Recherche en Informatique de Toulouse, 2014, 43p. [en ligne]. Disponible sur : < http://www-public.tem-tsp.eu/~lauren_m/WEB-ActionSSO/2014-SSO-School-NFC-IRIT-ENSEEIH-T-FabricePeyrard.pdf >. (Consulté le 26/02/2019)
- [49] BONNET, Sébastien. MQTT : Un protocole dédié pour l'IoT · [en ligne]. digital dimension, Publié le :27/02/2015. Disponible sur: <http://www.digitaldimension.solutions/blog/avis-d-experts/2015/02/mqtt-un-protocole-dedie-pour-liot/> > (Consulté le 26/02/2019)
- [50] PIPER, Andy. Choosing Your Messaging Protocol: AMQP, MQTT, or STOMP.

Bibliographie

[en ligne]. Disponible sur : < <https://blogs.vmware.com/vfabric/2013/02/choosing-your-messaging-protocol-amqp-mqtt-or-stomp.html> > (Consulté le 26/02/2019)

[51] Guides essentiels . Internet des Objets : bien comprendre MQTT. [en ligne]. Disponible sur : < <http://www.lemagit.fr/conseil/Internet-des-Objets-bien-comprendre-MQTT> >. (Consulté le 26/02/2019)

[52] FUTURA MAISON. [en ligne]. Disponible sur : < <https://www.futura-sciences.com/maison/definitions/maison-led-6968/> >. (Consulté le 19/02/2019)

[53] techopedia. Light-Emitting Diode (LED). [en ligne]. Disponible sur : < <https://www.techopedia.com/definition/2231/light-emitting-diode-led> >. (Consulté le 19/02/2019)

[54] 123RF. Stock Photo - Set of color 3 mm LED diodes isolated on white background. [en ligne]. Disponible sur : < https://www.123rf.com/photo_19706682_set-of-color-3-mm-led-diodes-isolated-on-white-background.html?undefined >. (Consulté le 19/02/2019)

[55] astuces pratiques. La résistance : Description et définition. [en ligne]. Disponible sur : <https://www.astuces-pratiques.fr/electronique/la-resistance-description-et-definition> (Consulté le 21/02/2019)

[56] Thomas et Mathieu DUBAËLE. Disponible sur : < <https://www.apprendre-en-ligne.net/crypto/passecret/resistances.pdf> >. [PDF]. (Consulté le 21/02/2019)

[57] NDIAYE , Yacine. IoT Cloud Platforms . [en ligne]. Disponible sur : < https://air.imag.fr/index.php/VT2016_IoT_cloud_platforms#Comparaison_de_quelques_plateformes_IoT >. (Consulté le 23/02/2019)

[58] BEN FREDJ , Sameh. BIEN CHOISIR SA PLATE-FORME POUR FAIRE COMMUNIQUER SES OBJETS CONNECTÉS. [en ligne]. Disponible sur : < <https://blog.xebia.fr/2015/10/21/bien-choisir-sa-plate-forme-pour-faire-communiquer-ses-objets-connectes/> >. (Consulté le 23/02/2019)

[59] White paper. IOT PLATFORMS The central backbone for the Internet of Things. IOT ANALITICS , 2015. Disponible sur : < <https://air.imag.fr/images/7/79/White-paper-IoT-platforms-The-central-backbone-for-the-Internet-of-Things-Nov-2015-vfi5.pdf> > [PDF]. (Consulté le 23/02/2019)

[60] CAREY, Scott . COMPUTERWORLDUK. Best cloud IoT platforms. [en ligne]. Disponible sur : < <https://www.computerworlduk.com/galleries/data/best-internet-of-things-platforms-3635185/> >. (Consulté le 23/02/2019)

Bibliographie

- [61] *Top 15 Most Popular IoT Platforms 2016- A Complete LIST*. [en ligne]. Disponible sur : < <http://iotfunda.com/top-15-popular-iot-platforms-2016-complete-list/> >. (Consulté le 23/02/2019)
- [62] David Adams, *The 4D Web Companion*, Chapter 47
- [63] *SPDY: An experimental protocol for a faster web*
- [64] Michael Jak, *REST, Representational State Transfer*
- [65] Source:<http://the-internet6.blogspot.sg/2013/04/01/archive.html>
- [66] source: <http://tools.ietf.org/html/draft-ietf-core-observe-13>
- [67] B. Berard M. Bidoit A. Finkel F. Laroussinie, A. Petit L. Petrucci Ph. Schnoebelen, with P. McKenzie, *Systems and Software Verification- Model-Checking Techniques and Tools With 67 Figures*, Springer 2001
- [68] Carles Anton-Haro and Mischa Dohler, *Machine-to-machine (M2M) Communications Architecture, Performance and Applications*, Elsevier, 2015, p 91.
- [69] <http://www.hivemq.com/blog/mqtt-essentials-part-1-introducing-mqtt>
- [70] *MQTT Version 3.1.1*. Edited by Andrew Banks and Rahul Gupta. 29 October 2014. OASIS
- [71] <https://www.oasis-open.org/news/pr/oasis-mqtt-internet-of-things-standard-now-approved-by-iso-iec-jtc1>.
- [72] Protocol Analysis - oneM2M, page 21, URL:
http://www.onem2m.org/images/files/deliverables/TR-0009-Protocol_Analysis-V0_7_0_1.pdf
- [73] Lucy Zhang, *Building Facebook Messenger*, août 2011,
URL:<https://www.facebook.com/notes/facebook-engineering/building-facebook-messenger/10150259350998920/>
- [74] <https://aws.amazon.com/fr/iot-platform/how-it-works/>
- [75] *mqtt-topics-best-practices*, URL: <http://www.hivemq.com/blog/mqtt-essentials-part-5-mqtt-topics-best-practices>.
- [76] *Building Smarter Planet Solutions with MQTT and IBM WebSphere MQ Telemetry*, IBM Redbooks, septembre 2012, p 26.
- [77] *mqtt-essentials-part-9-last-will-and-testament*,
URL:<http://www.hivemq.com/blog/mqtt-essentials-part-9-last-will-and-testament>.
- [78] Eric J. Bruno, *MQTT Programming In Depth*, March 2016,
URL: <http://programmingwithreason.com/article-mqtt-in-depth.html>.
- [79] *Internet of Things MQTT Quality of Service Levels*
URL:<http://www.ossmentor.com/2015/04/internet-of-things-mqtt-quality-of.html>
- [80] Jan-Piet Mens, *Using-the-MQTT-IoT-protocol-for-unusual-but-useful-purposes*
URL : <http://www.admin-magazine.com/Archive/2015/30/Using-the-MQTT-IoT-protocol-for-unusual-but-useful-purposes>
- [81] *Travaux de la 4e promotion (2013-2014) du Cycle « Sécurité des usages numériques »*, p 28

Bibliographie

[82] Peter R. Egli, *MQTT - MQ Telemetry Transport for Message Queueing*, 2016, p25,

URL: <https://www.slideshare.net/PeterREgli/mq-telemetry-transport>

[83] RT Fielding et RN Taylor, « la conception du Principled architecture web moderne, » *ACM Transactions sur les technologies Internet () TOIT*, vol. 2, non. 2, pp. 115-150, 2002.