الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne démocratique et populaire

وزارة التعليم العالي و البحث العلمي
Ministère de l'enseignement supérieur et de la recherche scientifique

جامعة سعد دحلب البليدة
Université SAAD DAHLAB de BLIDA

كلية التكنولوجيا
Faculté de Technologie

قسم الإلكترونيك
Département d'Électronique

Mémoire de Master

Filière télécommunication
Spécialité Réseaux et Télécommunication

Présenté par

MAIGA ABDOUL AZIZ

&

LAGHOUATI BOUCHRA

Conception d'un système de simulation des objets connectés

Proposé par : Mr Kabir Yacine

Année Universitaire 2019-2020

Remerciements

La réalisation de ce mémoire a été possible grâce au concours de plusieurs personnes à qui nous voudrons témoigner toute notre gratitude.

Nous tenons tout d'abord à exprimer toutes nos reconnaissances à notre directeur de mémoire, Monsieur **Yacine KABIR**. Nous le remercions de nous avoir encadré, orienté, aidé et conseillé.

Nous tenons également à remercier sincèrement Monsieur **Mehdi**, Monsieur **Bensebti** et Monsieur **Djendi** de nous avoir apporté tant de connaissances le long de notre parcours scolaire.

Nous ne manquerons pas de remercier l'ensembles des professeurs du département d'électronique de nous avoir accompagné durant notre cursus scolaire.

Nous ne manquerons pas de remercier aussi tous nos camarades étudiants avec qui on a su se soutenir mutuellement durant toutes ces années scolaires.

En fin nous réitérons nos remerciements à nos parents, amis et connaissances qui nous ont apporté leur soutient durant nos moments difficiles.

Merci à tous

Dédicace

Je dédicace ce modeste travail accompagné d'un profond amour ;

A ma mère

Celle qui a tout donné, tout mis en œuvre afin de me voir un jour heureux. Aujourd'hui je vous rends hommage maman, grâce à vos prières je suis à bon port.

A mon père

Celui qui à toujours su donner les meilleurs conseils de la vie sociale à ces enfants. Grâce à vous j'ai gardé la tête haute dans mes études. Merci papa

A mes frères et sœurs

Vous qui m'avez toujours soutenue tant matériel que morale, ma dédicace à vous d'avoir toujours été là pour moi. En plus de vous être reconnaissant, ma fierté est immense. Merci à vous

A tous les grand parents, oncles, tantes, cousins, cousines, neveux, nièces, parent proches ou lointain, je vous dédicace ce travail de fin d'étude de Master.

MAIGA Abdoul-Aziz

Dédicace :

Je dédie ce modeste travail en premier lieu à ma maman qui s'est sacrifier et qui a travailler sans relache pour assurer mon éducation et mon bonheur, que dieux me la garde.

A mes préscieux frères et sœurs qui m'ont aidé et soutenu tout au long de mon cursus Khalila Amina et Hichem.

A mes très chers amis qui m'ont toujours encouragé en particulier : Camélia, Célia, Anya, Nabila.

A mes amis de promotion Master 2 Réseaux et Télécommunications.

LAGHOUATI Bouchra

ملخص:

في هذا المشروع، نقترح نظامًا أساسيًا لمحاكاة الكائنات المتصلة. بهدف تقديم حل محاكاة لمشغلي إنترنت الأشياء، تتيح المنصة إنشاء مشاريع محاكاة لكائنات إنترنت الأشياء في بيئة محددة جيدًا. يمكن أن تكون البيئة منزلًا، أو، مصنعًا، أو مزرعة، أو ورشة عمل، إلخ. عند إنشاء المشروع، يمكن للمستخدم إضافة غرف إلى بيئته من أجل تثبيت أجهزة استشعار ومحركات في كل غرفة والتي ستكون موضوع محاكاة. تتمتع أجهزة الاستشعار بالقدرة على إنشاء البيانات تلقائيًا ويمكن للمشغلين استخدام هذه البيانات لتغيير حالاتهم. يوفر النظام ككل أمان وصول الوسيط لتوفير تجربة أمان أفضل للمستخدمين.

الكلمات الرئيسية: الكائنات المتصلة؛ مجسات المحركات. سماسرة إنترنت الأشياء

Résumé : Dans ce projet, nous proposons une plateforme de simulation d'objets connectés. Dans le but de proposer une solution de simulation aux acteurs de l'internet des objets, la plateforme permet de créer des projets de simulation d'objets IOT dans un environnement bien défini. Un environnement pourrait être une maison, une usine, une ferme, un atelier, etc. Dans la création d'un projet, l'utilisateur peut ajouter des locaux à son environnement afin d'installer dans chaque local des capteurs et des actionneurs qui vont faire l'objet d'une simulation. Les capteurs ont la capacité de générer des données de manière automatiques selon des profiles définies et les actionneurs peuvent exploiter ces données pour changer leurs états. Le système dans son ensemble offre une sécurité d'accès au broker afin d'offrir une meilleure expérience de sécurité aux utilisateurs.

Mots clés : Object connectés ; Capteurs ; Actionneurs ; Brokers ; IOT

Abstract: in this project, we are offering a platform for simulating connected objects. With the aim of providing a simulation solution to Internet of Things players, the platform makes it possible to create IOT object simulation projects in a well-defined environment. An environment could be a house, factory, farm, workshop, etc. in the creation of a project, the user can add rooms to his environment in order to install in each room sensors and actuators, which will be the subject of a simulation. Sensors have the ability to generate data automatically according to predefined profiles and actuators can use this data to change their states. The system as a whole provides broker access security to provide a better security experience for users.

Keywords: Connected objects; Sensors; Actuators; Brokers; IOT.

Liste des acronymes et des abréviations

IOT:	Internet Of Things
IA:	Intelligence Artificielle
IBN:	Intent-Based Network
LPWAN:	Low Power Wide Area Network
B2B:	Buisness to Buisness
M2M:	Machine to Machine
GSM:	Global System Mobile
LoRaWAN:	Long Rang Wide Area Network
LoRa:	Long Rang
MH:	Mega-Hertz
KbPS:	Kilobites Par Seconde
2G:	2eme Génération
3G:	3eme Génération
4G:	4eme Génération
Mb/S:	Megabit/Secondes
5G:	5eme Génération
CoVid-19:	Corona Virus Disease-2019
mMTC:	massive Machine Type Communications
eMBB:	enhanced Mobile Broadband
uRLLC:	ultra-reliable low latency
SIDA:	Syndrome d'Immunodéficience Acquise
ONU:	Organisation des Nations Unies
FAO:	Food and Agriculture Organization
GPS:	Global Positioning System
V2V:	Vehicle to Vehicle
V2I:	Vehicle to Infrastructure
OpenSHS:	Open Smart-Home Simulator

MQTT:	Message Queuing Telemetry Transport
HTTP:	Hyper Text Transfer Protocol
TinyOS:	Tiny Operating System
WSN:	Wireless Sensor Network
Matlab:	Matrix Laboratory
SQL:	Structured Query Language
JSON:	JavaScript Object Notation
CSS:	Cascade Style Sheet

Table des matières

Introduction générale 1

Chapitre 1 : Les fondamentaux de l'internet des objets 2

1.1. Introduction 2

1.2. Les objets 3

1.2.1. Les objets traditionnels et nouveaux objets 4

1.2.2. Les capteurs 5

1.2.3. Les actionneurs 7

1.3. Les réseaux de connexion 8

1.3.1. Les LPWAN : Sigfox et Lorawan 8

1.3.2. Les réseaux mobiles : 2G, 3G, 4G et 5G 10

1.3.3. La 5G 12

1.4. Domaines d'applications 13

1.4.1. La santé 13

1.4.2. L'agriculture et l'élevage 14

1.4.3. Territoires intelligents et transport 17

1.5. Conclusion 19

Chapitre 2 : Etude de faisabilité et architectures de conception 20

2.1. Introduction 20

2.2. Étude de faisabilité 20

2.2.1 Etat d'art 21

2.2.2. Description et but du projet 23

2.2.3 Analyses techniques 24

2.2.4. Validation 27

2.3. Architectures de conception 28

2.3.1. Architecture de l'interface principale 28

2.3.2. Architecture de l'interface de visualisation 32

2.3.3. L'architecture d'ensemble 33

2.4. Programmation des interfaces 34

2.4.1. Python et MySQL pour l'interface principale 34

2.4.2. Javascript pour le backend de l'interface de visualisation 35

2.4.3. HTML et CSS pour le frontend 35

2.5. Conclusion 37

Chapitre 3 : implémentation et résultats de simulation 38

3.1. Introduction 38

3.2. Gestion des communications entre les interfaces du système 38

3.2.1. Le protocole MQTT 39

3.2.2. Le protocole websocket 40

3.2.3. Serveur Mosquitto 41

3.3. Structure du programme et présentation des interfaces 42

3.3.1. Structure du programme 42

3.3.2. Présentation des interfaces 44

3.4. Phase de conception d'un projet 46

3.4.1. Choix d'environnement 47

3.4.2. Choix des locaux 50

3.4.3. Choix de capteurs et actionneurs 52

3.5. Phase de configuration et simulation 54

3.5.1. Configuration des capteurs 55

3.5.2. Configuration des actionneurs 60

3.5.3. Lancer la simulation 62

3.5.4. Sécurité 65

3.6. Conclusion 68

Conclusion générale 69

Liste des figures

Figure 1.1 : Prévision de nombre d'objets connectés de 2015 à 2025[2]. 3
Figure 1.2 : objets traditionnels 4
Figure 1.3 : quelques types de capteurs.....6
Figure 1.4 : quelques types d'actionneurs.....8
Figure 1.5 : l'IOT appliqué à l'agriculture [19]. 16
Figure 1.6 : capteur de récolte de donnée sur la santé animale 17

Figure 2.1 : résumé des grands points de l'étude de faisabilité 28
Figure 2.2 : architecture de l'interface principale. 29
Figure 2.3 : Architecture de l'interface de visualisation 32
Figure 2.4 : Architecture résumée du système de simulation.....33

Figure 3.1 : configurer websocket avec Mosquitto 41
Figure 3.2 : fonctionnement d'un serveur MQTT 42
Figure 3.3 : déclaration de la classe application 43
Figure 3.4 : déclaration de la classe thread_class.....43
Figure 3.5 : photo illustrative du code source de l'interface de visualisation.....44
Figure 3.6 : interface d'accueil du système de simulation. 45
Figure 3.7 : interface de visualisation du système avant simulation 46
Figure 3.8 : architecture de conception d'un projet.....47
Figure 3.9 : ouverture d'une session pour un nouveau projet.....48
Figure 3.10 : l'interface après ouverture d'un nouveau projet.....48
Figure 3.11 : listes des environnements de choix par défaut.....49
Figure 3.12 : modifier le nom d'un environnement.....50
Figure 3.13 : liste des locaux.....51
Figure 3.14 : ajout d'un local a un environnement.....52
Figure 3.15 : liste des capteurs par défaut 53
Figure 3.16 : listes des actionneurs par défaut.....53
Figure 3.17 : le comportement des objets en simulation sans une configuration préalable ..55
Figure 3.18 : interface de configuration d'un capteur 55
Figure 3.19 : affichage des données entrées pour un capteur.....57
Figure 3.20 : exemple de graphe d'évolution d'un capteur 59
Figure 3.21 : interface de configuration d'un actionneur 60
Figure 3.22 : étape de lancement de la simulation.....62
Figure 3.23 : données d'un capteur sous un format json 63
Figure 3.24 : résultat de la simulation.....64
Figure 3.25 : commande de création d'un mot de passe sur Mosquitto.....66
Figure 3.26 : mot de passe crypté sous Mosquitto.....66

Figure 3.27 : bloquer les accès anonymes sur Mosquitto.....66
Figure 3.28 : champ de saisie nom d'utilisateur, identification Mosquitto 67
Figure 3.29 : champ de saisie mot de passe, identification Mosquitto 67
Figure 3.30 : message d'erreur en cas d'authentification incorrecte.....67

Liste des tableaux

Tableau 2.1: tableau résumant les éléments de l'architecture de l'interface principale 31
Tableau 2.2 : résumé des langages, leurs atouts, leurs utilités pour le projet et l'interface d'utilisation. 36
Tableau 3.1 : description des champs de configuration d'un capteur 56
Tableau 3.2 : description des champs de configuration d'un actionneur 61

Introduction générale

L'Internet des objets (ou IOT) se traduit à l'heure actuelle par l'accroissement du nombre d'objets connectés, c'est-à-dire d'appareils possédant une identité propre et des capacités de communication de plus en plus sophistiquées : téléphones, montres, appareils ménagers, etc. Ces objets embarquent un nombre grandissant de capteurs et d'actionneurs leur permettant de récolter des donner l'environnement et d'agir sur celui-ci.

L'internet des objets fait actuellement la une de la transformation numérique vers une société plus connectée intégrant cette fois ci un écosystème de communication entre les humains et les objets. Ainsi beaucoup d'entreprises, de particuliers ou encore des gouvernements se lancent dans la mise en place de solutions IOT mobilisant quelques capteurs à des milliers voir des millions de capteurs et actionneurs. Cependant, il serait très coûteux voir difficile de faire un essai pratique de la mise en place d'une solution IOT avant sa conception. Le monde de l'IoT a donc besoin des technologies permettant une simulation virtuelle de ces objets avant une mise en œuvre réelle.

Notre projet a pour but de mettre en place un environnement de simulation d'objets connectés afin de permettre aux acteurs de l'IoT de mieux analyser leur solution IOT avant une mise en place réelle. Ce qui pourrait réduire les couts de conceptions ainsi que le temps de réalisation d'un projet IOT.

Dans la mise en œuvre de ce projet, nous allons aborder premièrement, une étude bibliographique des fondamentaux de l'IOT, ensuite réaliser une analyse conceptuelle du projet en vue d'une mise en œuvre plus efficace. Et nous terminerons par une analyse des performances du projet final par rapport aux attentes théoriques des parties précédentes.

Chapitre 1 : Les fondamentaux de l'internet des objets

1.1. Introduction

L'Internet des objets (IoT) désigne l'interconnexion d'un grand nombre d'appareils et de capteurs intelligents connectés à Internet. Ces capteurs et ces appareils connectés collectent et partagent des données qui seront utilisées et analysées par plusieurs organismes, dont des entreprises, des villes, des gouvernements, des hôpitaux et des particuliers. L'avènement de l'IoT a possible, en partie, grâce à l'apparition des processeurs bon marché et des réseaux sans fil. Des objets jusqu'à présent inanimés (comme des poignées de porte ou des ampoules électriques) peuvent désormais être équipés d'un capteur intelligent qui collecte des données et les envois vers un serveur distant via un réseau.

Les chercheurs estiment que 3 millions de nouveaux terminaux se connectent à Internet chaque mois. Dans les quatre prochaines années, ce chiffre devrait atteindre les 30 milliards d'appareils connectés dans le monde entier **[1]**.

Environ un tiers des appareils connectés seront des télévisions intelligentes, des smartphones, des tablettes et des ordinateurs. Les deux tiers restants correspondront à des « objets » : des capteurs, des actionneurs et des appareils intelligents innovants qui surveillent, gèrent, analysent et optimisent notre monde.

Le monde a rapidement été couvert par des réseaux qui permettent aux appareils numériques de s'interconnecter et de transmettre des données. Nous vivons actuellement une transformation numérique, à mesure que les réseaux numériques continuent de gagner du terrain dans le monde entier et que les bénéfices économiques de la numérisation se multiplient. La transformation numérique consiste à appliquer la technologie numérique pour créer les bases de l'innovation dans les entreprises et le secteur de l'industrie. On assiste de plus en plus à des applications liées à l'internet des objets. Le but de ce chapitre est de faire une brève description sur les différents composants qui entrent dans la constitution d'un système de réseau d'objets connectés.

1.2. Les objets

Les objets sont tous types d'équipements actifs ou passifs pouvant soit générer une donnée exploitable et créatrice de valeur pour l'utilisateur, soit capable de réaliser une action mécanique en vue d'assurer une tâche donnée. Ainsi dans l'internet des objets on peut classer objets en trois catégories différentes : les objets traditionnels, les nouveaux objets connectés, les capteurs et actionneurs. Les objets sont en quelque sorte, la matière première dans l'IOT, tout commence par ces objets.

Dans cette partie nous allons illustrer les objets suivant les différentes catégories que nous venons de citer.

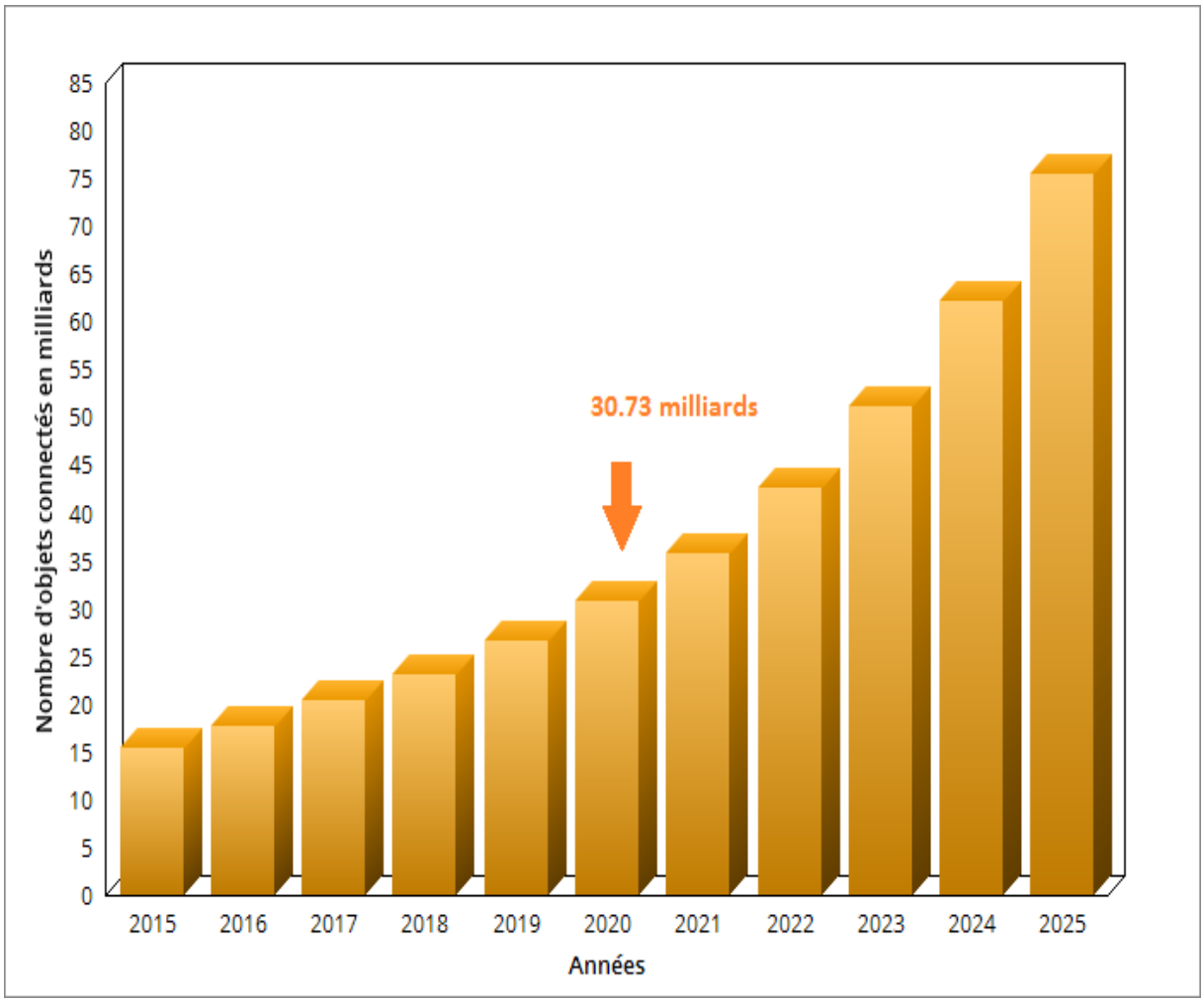

***Figure 1*.1 :** Prévision de nombre d'objets connectées de 2015 à 2025 [2].

1.2.1. Les objets traditionnels et nouveaux objets

On s'est tous déjà connecté à internet via un objet sans forcément se rendre compte ou prendre conscience que l'ordinateur, le smartphone ou la tablette que vous utilisez est un objet. Cependant l'évolution de l'internet va faire surgir d'autres types d'objets connectés que nous appellerons nouveaux objets.

- **Les objets traditionnels**

Lorsqu'on parle d'objets traditionnels, on parle simplement des premiers objets à avoir la capacité de se connecter à internet et utilisé par l'homme depuis bien longtemps. Ce sont notamment les ordinateurs, les smartphones ou encore les tablettes. Ces premiers objets ont changé notre vie quotidienne avant même l'arrivé de l'internet.

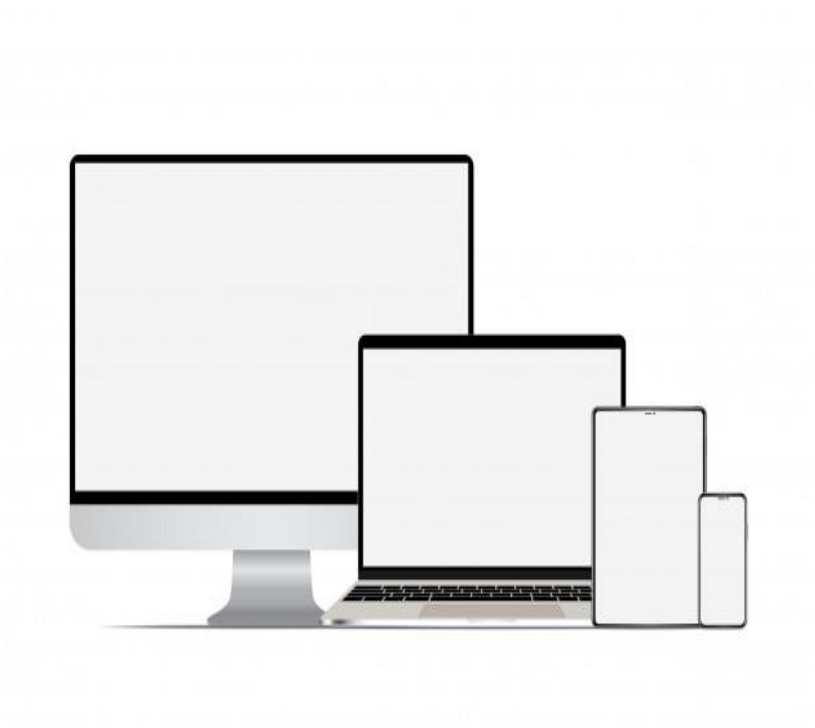

Figure 1.2 : objets traditionnels [1].

- **Les nouveaux objets connectés :**

Ce sont des objets qui, autrefois l'on n'aurait pas crue qu'ils pouvaient être connectés à internet pour rendre service à l'homme. Ce sont notamment les appareils électroménagers, les instruments de mesure, les robots, serrures

de portes, machines-outils, bennes à ordures, drones, jouets, montres, véhicules, etc.

1.2.2. Les capteurs

Les capteurs sont des objets permettant la collection et la transmission d'énormes quantités de données issues du monde physique. Ils permettent en effet de traduire une grandeur physique en un signal électrique pour être transmis au système informatique. Ces données peuvent être stockées et analysées à une date ultérieure, ou être analysées et exploitées immédiatement **[2].** Les capteurs peuvent être installés dans les maisons, dans les espaces publiques, dans les champs agricoles, sur nos corps, etc. Les données des capteurs sont analysées et utilisées par les gouvernements, les villes, les entreprises et les particuliers pour mettre en place des changements, comme surveiller l'environnement, prévoir les tendances démographiques, contrôler la gestion des déchets, sécuriser une maison, contrôler l'humidité des champs agricoles, détecter des fuites de gaz ou d'eau, etc. On distingue une multitude de capteurs selon le besoin d'utilisation :

- **Capteur de température**

  Un capteur de température permet de traduire L'amplitude de la température en une tension électrique. Cette Dernière est numérisée puis transmise sur le réseau, où des appareils comme le climatiseur ou le chauffage peuvent s'en servir pour réguler la température d'un environnement donné.

- **Capteur de détection de mouvement**

  Les capteurs de détection de mouvement font partie intégrante du monde qui nous entoure. Un capteur de détection de mouvement permet de détecter les mouvements dans une zone d'opération précise afin de permettre le déclenchement d'un dispositif comme le système d'alarme, l'allumage des ampoules, etc.

❖ **Capteur d'humidité**

Un capteur d'humidité permet, comme son nom l'indique, de mesurer l'humidité ambiante dans un environnement donné. Ce genre de capteur est surtout utilisé dans le domaine de l'agriculture ou dans les fermes d'élevage.

❖ **Capteur de luminosité :**

C'est un capteur qui permet de détecter la luminosité de l'environnement qui lui est dédié. Ce genre de capteur est déjà intégré dans nos smartphones et permet de réguler la luminosité de l'écran selon la luminosité extérieur. Dans l'IOT on peut installer ce genre de capteur dans les maisons intelligentes afin de réguler la luminosité des ampoules selon celle d'extérieure et dans les villes intelligentes pour la régulation des éclairages publics. Il existe d'autres types de capteurs tel que les capteurs de détection de fuite d'eau, de fuite de gaz, de détection de feu, etc. Des milliers de capteurs sont déjà déployés dans des réseaux IOT à travers le monde et font déjà l'objet du big data.

Figure 1.3 : quelques types de capteurs [19].

1.2.3. Les actionneurs

Les actionneurs connectent désormais des milliards de capteurs et peuvent modifier leurs environnements physiques sans intervention humaine. Les réseaux du futur tourneront autour de l'IA (intelligence artificielle) et de l'IBN (réseaux basés sur l'intention). S'ils sont correctement programmés, les appareils intelligents sont capables d'évaluer les données qui leur sont fournies, et de modifier des processus ou des paramètres. Si on leur fournit suffisamment de données, ils peuvent « apprendre » et modifier leur propre code sur la base de ces nouveaux paramètres.

Les actionneurs sont des dispositifs qui transforment une donnée digitale en un phénomène physique pour créer une action. La plupart du temps ces données viennent des capteurs. En effet lorsque les capteurs recueillent des données et les transmettent à un serveur, celui si les redirige vers les actionneurs qui vont exécuter une action physique selon le type et la valeur de la donnée reçue. Ainsi un système d'alarme va se déclencher une alerte lorsqu'il reçoit un signal d'un capteur de mouvement lui indiquant la présence d'un intrus dans une maison par exemple, ou un climatiseur qui se déclenche en recevant une information du capteur de température lui indiquant qu'il fait chaud dans le salon. Les actions couramment utilisées sont :

- Allumage d'un éclairage
- Déclenchement d'un avertisseur sonore
- Allumage d'une machine
- Génération de mouvements (ex. servomoteur)
- Commande de robots
- Commande de moteurs (à courant continu, pas-à-pas, etc.)
- Contrôle de débits (air, pression, liquides, etc.)

Figure 1.4 : quelques types d'actionneurs [20].

1.3. Les réseaux de connexion

Qui parle d'objets connectés parle de réseaux internet puisque sans ces réseaux, nos objets ne pourront ni accéder à internet ni échanger des données entre eux.

1.3.1. Les LPWAN : Sigfox et Lorawan

Actuellement en plein essor mondial, le LPWAN pour « Low Power Wide Area Network » ou réseaux basse consommation et longue distance, constitut une véritable révolution IOT en ce qui concerne le volet B2B des objets connectés.

A cet effet, les industrielles travaillent à intensifier le déploiement des réseaux LPWAN ainsi qu'à l'interopérabilité des solutions IOT qui les utiliseront. Par ailleurs, les réseaux LPWAN contribuent grandement à l'optimisation des batteries et des émetteurs des objets dites connectés par leur faible consommation de ressources énergétiques. Il existe cependant deux principaux réseaux LPWAN dédiés à l'internet des objets qui représentent tout de même deux univers et deux promesses différentes ce sont Sigfox et Lorawan.

a) Le réseau Sigfox :

Créé en 2009, Sigfox fut le premier réseau dédié uniquement à l'IOT. C'est un système de connexion utilisant des signaux de fréquence radio ultra-rapide et de longue portée. On parle de bande ultra étroite pour le système M2M (communication machine to

machine) qu'il révolutionne d'ailleurs. Le principe de Sigfox est de réinventer la transmission d'informations en réduisant de manière significative les consommations d'énergie des périphériques connectés et leur prix. On parle surtout de révolution car il créé un réseau longue portée et à bas débit qui permet la communication de données de tailles réduites entre les objets connectés sans passer par un téléphone mobile. Les objets communiquent entre eux et partages des commandes, des données entre eux et communiquent.

Par ailleurs, Sigfox a pour but de connecter le monde réel avec le monde virtuel. De ce fait les objets deviennent intelligents et pourrons exécuter leurs tâches respectives sans l'intervention de l'homme. On parle ici de connecter des objets simples comme les congélateurs, les machines à café, le chauffage, etc...

Sigfox reste un réseau dédié qui ne concerne pas les appareils mobiles comme les smartphones, les ordinateurs portables et les tablettes disposant déjà de leur propre moyen de connexion comme le réseau GSM. En somme Sigfox n'a peu de limites. Il œuvre dans tous les domaines d'activités possibles et tous les secteurs imaginables de l'internet des objets.

b) Le réseau Lorawan :

Le réseau Lorawan de son vrai nom LoRa signifie Long Range ou « longue portée », a vu le jour en 2012. Tout comme son concurrent Sigfox, LoRa est une technologie qui permet aux objets connectés d'échanger des données de faible taille en bas débit. Cela permet de réduire considérablement la consommation énergétique des appareils, leur conférant jusqu'à 10 ans d'autonomie.

LoRa utilise à la fois des fréquences radio libres 868MH et internet. Etant peu gourmande en débit et en énergie, elle a l'avantage d'être très économique pour l'utilisateur final et présente de plus, une excellente capacité de pénétration des bâtiments, caves et sous-sols : compteurs d'eau intelligent, géolocalisation des biens industriels, ainsi que pour l'agriculture connectée. Elle intègre également le domaine de l'internet des objets par le champ de communication de machine to machine M2M.

Sa capacité de pénétration est une caractéristique intéressante permettant de localiser des objets de valeurs gardés dans des zones difficiles d'accès.

Contrairement à son concurrent Sigfox, le réseau LoRa est plus facile d'accès. Toute entreprise peut donc créer son propre réseau pour l'exploiter. Il faut pour cela se munir d'une antenne reliée à internet. La bande de fréquence utilisée change par pays et selon les normes de ce pays. La bande de fréquence utilisée par le réseau LoRa est par exemple de 868MHz en France et de 915MHz aux Etats unis.

En outre les objets connectés doivent être équipés d'une puce LoRa qui leur permet de recevoir le signal de l'antenne. Cependant la puissance du débit du réseau est bien moins élevée que celui de la 3G. Il est question d'un débit compris entre 0.3 à 50kbps selon les besoins. Ce débit s'adapte automatiquement aux besoins des objets afin de limiter l'emploie de la bande passante et donc la consommation.

En somme le concept de LoRa joue principalement sur l'économie d'énergie pour les objets en eux-mêmes. Elle entend répondre aux problématiques de l'IOT en apportant des liaisons bidirectionnelles entre l'objet et le réseau de collecte sur plusieurs kilomètres de distance avec une faible consommation.

Sigfox promet une meilleure pénétration et une meilleure portée et, LoRa annonce une meilleure communication bidirectionnelle et une localisation par triangulation plus fine.

1.3.2. Les réseaux mobiles : 2G, 3G, 4G

Nous avons vu que LoRa et Sigfox étaient des réseaux bas débit et longue portée, ce qui ne leurs permet pas d'être efficace pour certains types de solution IOT sollicitant un haut débit comme l'IOT dans le domaine de l'automobile autonome ou dans la réalité virtuelle. C'est là que vient l'importance des réseaux de télécommunications mobiles qui d'ores et déjà prennent en charge des transmissions à haut débit.

a) La 2G et la 3G :

A la base les technologies réseaux de deuxième et troisième génération (2G et 3G) n'étaient pas destinées à la connexion des objets proprement dite mais juste pour la

téléphonie mobile et l'internet. Mais la transformation numérique oblige les opérateurs télécoms à intégrer de nouveaux services par rapport aux nouveaux besoins sur le marché. Ainsi sans toucher à la technologie de base des réseaux GSM existants, on peut utiliser la 2G ou la 3G et même la 4G (que nous verrons dans le prochain point) pour mettre en place une solution IOT. Ils permettent surtout une couverture réseau plus étendue pour les objets connectés. A la limite la 2G et la 3G n'assure que la transmission des données plus ou moins volumiques, de la voix et des sms mais pas adaptés à des vidéos en direct à cause d'un problème de débit. C'est là que la 4G intervient pour résoudre ce problème.

b) La 4G :

La 4G a révolutionné l'univers des objets connectés. En effet avant l'arrivée de celle-ci, les objets utilisant de la vidéo ou requérant un haut débit avaient du mal à fonctionner efficacement. Mais avec le déploiement de ce réseau haut débit, les objets utilisant de la vidéo tel que les caméras de surveillance, ou requérant un haut débit comme les terminaux de paiement, peuvent tout à fait fonctionner de manière efficace en temps réel.

La 4G a donné un espoir à l'internet des objets et de plus en plus apparait sur le marché des petites et moyennes entreprises ainsi que des start-ups qui se spécialisent dans la conception d'objets IOT ainsi qu'au déploiement de ses derniers.

L'un des points fort de la 4G est la possibilité d'offrir des débits pratiques plus élevés allant jusqu'à 40 Mbit/s. Ce qui fait qu'elle vise surtout le secteur industriel qui est un peu gourmand en débit internet. Ainsi elle permet aux objets d'être plus réactif et de transmettre un grand nombre de données en temps réel. On a également la naissance des plateformes en ligne spécialisées dans l'IOT et qui permettent de contrôler à distance un ensemble d'objets connectés tel que la climatisation, l'éclairage, le système d'alarme, le thermostat, etc.

Malgré les possibilités grandioses que nous offre la 4G sur l'internet des objets, elle reste tout de même limitée à satisfaire à 100% le besoin du monde de l'IOT.

1.3.3. La 5G

L'internet des objets est un réseau des réseaux aux possibilités infini qui va révolutionner notre manière de vivre. Cependant, l'accroissement du nombre d'objets connectés impose la nécessité d'une mise en place d'un réseau plus puissant que la 4G. c'est ainsi que voit le jour la 5G.

En regardant les réseaux de la 2G jusqu'à la 4G, nous constatons simplement une amélioration continue du problème de débit. Quant à la 5G c'est tout une révolution qui va au-delà de la résolution d'un problème de débit mais promet de nouvelles possibilités et une capacité d'adaptation sans précédente. La 5G sera 100 fois plus performant que la 4G et s'adapte à tout type d'objets connectés. Le réseau 5G n'aura donc rien avoir avec les réseaux mobiles traditionnels que nous connaissons aujourd'hui. En effet on parle ici de tout un arsenal technologique révolutionnaire qui est en quête d'être mis en place.

On distingue de manière générale trois grandes catégories d'usage de la 5G.

- mMTC (Massive Machine Type Communication) :

  Cette catégorie illustre la communication entre une grande quantité d'objets avec des besoins de qualité de service variés. Son objectif est de répondre à l'augmentation exponentielle de la densité d'objets connectés. En effet la 5G offres jusqu'à 100 fois plus d'appareils connectés par unité de surface par rapport à la 4G ; une couverture réseau à 100% et une disponibilité de 99,99%.

- eMBB (Enhanced Mobile Broadband) :

  Une connexion en ultra haut débit en outdoor et en indoor avec uniformité de la qualité de service. La 5G offre un débit allant jusqu'à 10 Gbit/s et 1000 fois plus de bande passante par unité de surface que la 4G. Ce qui est vraiment énorme pour assurer une connexion haut débit à tout type d'appareil comme les applications de la réalité virtuelle ou augmentée.

- uRLLC (Ultra-Raliable Low Latency Communication) :

L'uRLLC a pour but une communication ultra-fiable pour les besoins critiques avec une très faible latence, pour une réactivité accrue. Ce besoin est nécessaire pour les applications aux véhicules connectés ou encore à la télémédecine. Pour ce fait la 5G nous offre une latence de 1ms (10ms pour la 4G) et une disponibilité de 99,99%.

En somme la 5G ne signifie pas seulement un internet plus rapide, mais surtout encore plus d'objets connectés. L'avènement de la 5G va changer l'image du monde. On aura en effet plus de voitures autonomes, des maisons intelligentes, des villes intelligentes partout dans le monde ainsi que le développement sans précèdent de la réalité virtuelle.

1.4. Domaines d'applications

Plusieurs domaines d'applications sont touchés par l'internet des objets, parmi ces domaines : la santé, le transport, l'agriculture, l'industrie, l'automobile, l'habitat, etc.

1.4.1. La santé

L'application de l'internet des objets dans le domaine de la santé connait un essor fulgurant, ce qui permet à un patient et à son médecin d'établir un contact à distance et recevoir des informations en temps réel. Des objets connectés spécialisés dans la médecine voient progressivement le jour.

Avec la situation pandémique du covid-19, nous pouvons remarquer à quel point l'utilisation de l'IOT dans la médecine pourrait aider les populations dans des situations de pandémies pareilles. Plusieurs plateformes ont d'ailleurs vu le jour permettant des consultations en ligne et ainsi éviter le contact direct avec un maximum de la population dans le but de limiter la propagation du virus. Par exemple une startup française, **PKVitality** a pu développer une montre connectée, **K'watch** qui permet la surveillance de la glycémie dans le sang sans prélèvement sanguine.

Les objets connectés peuvent faciliter la prévention ou l'accessibilité aux soins des populations dans un contexte où le nombre de médecins par habitant n'est que

de 15 pour 100 000 **[1].** L'IoT permet de dépasser les situations de déserts médicaux notamment grâce aux outils de télédiagnostic :

Les bracelets connectés enregistrent les informations de santé (vaccins, pathologies, allergies...) et permettent un suivi régulier.

Les applications mobiles de télémédecine permettent de réaliser des diagnostics à distance via un simple téléphone connecté à l'internet mobile.

Au Kenya, les actions de santé mobile menées par Orange Healthcare ont permis d'améliorer de 11 % le taux d'adhérence aux traitements de lutte contre le SIDA **[4].**

1.4.2. L'agriculture et l'élevage

L'internet des objets appliqué à l'agriculture et à l'élevage va révolutionner ces deux domaines tant dans la rentabilité que dans la gestion des ressources animales et agricoles. Dans cette partie, nous abordons la nécessité de l'IOT dans ces deux domaines.

a) L'agriculture

Les statistiques rapportés par l'ONU ont fait ressortir que la terre héberge 7.7 milliards d'habitants à la date du premier Janvier 2020 avec une croissance de +1.1%, ce qui conduit à une augmentation de 80 millions de personnes chaque année (soit près de 220,000 personnes chaque jour) [18]. Ainsi en 2100 le monde va héberger 10.9 milliards d'habitants.

Pour nourrir autant de personnes sachant que la famine fait de nombreux dégâts dans le monde, la solution est tout simplement de marier l'agriculture à la technologie afin de booster les rendements des espaces agricoles. L'IOT intégré à l'agriculture permet non seulement d'augmenter la qualité de production, mais aussi de diminuer les coûts de production.

L'agriculture intelligente utilise des capteurs IOT qui permettent de récolter les données des champs tel que la température, l'humidité des sols, le PH des sols,

mesurer de quantité des nutriments contenus dans le sol, etc. Ainsi, ces données vont être envoyé dans le cloud pour être traité et prendre des décisions le plus rapidement possible, parfois de manière automatique tel que l'irrigation automatique, l'apport de solutions nutritives automatisé, etc. Il y'a aussi l'utilisation des drones pour le contrôle aérien tel que la détection des irrégularités d'évolution des plantes dans les champs, ou encore la vérification des drainages de sol pour les campagnes de semences. Cela constitue un gain de temps énorme aux agriculteurs.

En outre, l'IOT appliqué dans l'agriculture pourrait aider l'Afrique non seulement à atteindre l'autosuffisance, mais aussi à booster l'économie du continent. En effet, l'Afrique à elle seule regorge 60% des terres arables non exploités dans le monde mais malheureusement dépenses 30 milliards de dollars dans les importations (selon un rapport de la FAO) de produits alimentaires [**18**]. Et si rien n'est fait ce seront 120 milliards en 2050, sachant que plus de 60% de la population active en Afrique exerce dans l'agriculture.

Le problème n'est pas le manque d'agriculteurs, mais plutôt le manque de rentabilité car les terres sont mal exploitées et mal gérés. L'IOT peut régler ce problème et même réduire les coûts de productions.

Ça serait dommage de ne pas profiter des nouvelles technologies pour bouleverser la tendance.

Figure 1.5 : l'IOT appliqué à l'agriculture [19].

b) L'élevage :

Une autre application de l'IOT est d'équiper les fermes et les animaux de capteurs intelligents permettant d'offrir un environnement idéal (une meilleure vie) aux animaux.

L'élevage connecté représente une véritable opportunité pour les acteurs du domaine. Il permet de contrôler les individus dans les fermes et la ferme elle-même de manière automatisée. Les capteurs peuvent servir à détecter les maladies au niveau des animaux et alerter immédiatement le vétérinaire en vue d'un soin immédiat. Les capteurs GPS peuvent servir par exemple à suivre les déplacements des animaux à distance.

Dans la filière volaille, les capteurs peuvent servir dans la recherche en détectant les types poulets qui avalent le plus de nourriture. Ces types de poulets vont être ensuite sélectionnés en vue d'obtenir des poulets de race pour une production massive de viande.

Figure 1.6 : capteur de récolte de donnée sur la santé animale [11]

1.4.3. Territoires intelligents et transport

D'autres applications de l'internet des objets sont liées aux territoires dit intelligents (l'habitat, l'industrie, les villes intelligentes, etc.) ainsi qu'à l'automobile.

a) Territoires intelligents :

Grace à l'internet des objets, les logements et lieux de travail deviennent plus confortables, plus facile à gérer et moins couteux à l'usage. Le bâtiment connecté, incluant la maison connectée, offre notamment des possibilités de contrôle des consommations énergétiques.

Dans le domaine de l'**habitat**, l'IOT va permettre de prendre le contrôle total d'une habitation à distance (éteindre les ampoules, mettre la climatisation en marche, etc.). Les maisons peuvent intégrer des capteurs et actionneurs intelligents contrôlé par des algorithmes basés sur l'intelligence artificielle. Ces algorithmes sont capables d'étudier

dans un premier temps les habitudes de leurs utilisateurs (locataires d'une maison par exemple) pour ensuite adapter la prise des décisions de façon autonome, offrant ainsi un environnement plus sécurisé et confortable.

En ce qui concerne les **villes intelligentes**, voici une définition selon Joan Enric Ricard (professeur à l'Institut d'Etudes Supérieurs de Barcelone I.E.S.E) rapporté par le magazine Forbes **[17]** « une ville vraiment intelligente, est une ville qui a pour objectif d'améliorer la qualité de vie de ses résidents, ce qui suppose d'assurer la sécurité économique et sociale ainsi que la durabilité environnementale ». Cela dit, les villes intelligentes est un projet futur qui prend de l'ampleur. Ainsi nous assisterons à la mise en place de parkings intelligents, de feus de routes intelligents, des éclairages publics intelligents qui vont réduire considérablement les dépenses publiques en énergies. Un autre domaine qui sera révolutionné par l'IOT, c'est **l'industrie**. L'industrie du futur va majoritairement reposer sur l'IOT. Il va apporter d'importantes modifications dans la manière de produire, de gérer et de communiquer dans les entreprises. En effet les machines industrielles peuvent être organisées en réseaux avec des technologies de détection (capteurs) leurs permettant d'avoir une grande autonomie et des précisions de conception accrues. Des capteurs peuvent être installés tout le long de la chaine de production afin de suivre la qualité de production, détecter les erreurs de fabrications mais aussi détecter les pannes avec précision permettant des interventions rapides et efficaces. Ce qui est profitable économiquement en termes de temps et de coût aux industriels.

b) L'automobile :

Avec l'avènement de la 5G, l'IOT n'a plus de limite. Presque tous les types d'objets pourront se connecter. Le secteur de l'automobile n'en fera pas d'exception. En effet l'IOT peut apporter une aide à la conduite afin d'assister les conducteurs pour des voyages de longues distances engendrant la fatigue du conducteur. C'est le cas par exemple de **Volvo** qui équipe certaines de ses voitures (les modèles s90 et xc90) de technologies de conduite semi-autonome. Ses voitures adaptent leur vitesse sur autoroute en respectant la distance de sécurité et intervient sur la direction lorsque

c'est nécessaire. En plus de l'aide à la conduite, l'IOT compte modifier un autre aspect de l'automobile et de manière efficace grâce à la 5G : c'est la voiture quasi autonome avec l'avènement des technologies V2V (pour vehicle to vehicle) et V2I (pour vehicle to infrastructure). Celles-ci permettront à terme aux voitures de communiquer entre elles mais aussi avec le réseau afin d'avoir une meilleure connaissance de leurs environnements et pouvoir agir d'elles même. Les voitures pourront prévenir leur homologue en cas d'accident ou en cas de route glissante. Elles pourront également adapter leur vitesse en conséquence, éviter les bouchons, éviter les accidents, etc. La marque Allemande Audit propose déjà, pour ses modèles Q7 et A4 des technologies pareilles pouvant indiquer à leur conducteur le temps d'attente d'un feu rouge ou même adapter leur vitesse afin d'éviter les feux rouges.

En plus de la conduite, l'IOT va aussi révolutionner les chaines de productions. En effet les constructeurs font déjà cohabiter les robots intelligents et autonomes avec les humains sur les chaines de production. C'est le cas par exemple de la marque Ford.

1.5. Conclusion

Ce premier chapitre a fait l'objet d'une étude théorique permettant de comprendre les concepts clés de l'internet des objets à savoir : les objets, les réseaux pour la connexion de ces objets et les domaines dans lesquels ces objets peuvent apporter de la valeur.

Les objets concernent essentiellement les téléphones, les ordinateurs, les équipements domestiques et industriels, les capteurs et les actionneurs. Ils constituent la base même de l'IOT.

Les réseaux servent à mettre en réseaux les objets en vue de communiquer, et ainsi on parle d'objets connectés. Les objets connectés sont alors installés dans des environnements dédiés pour apporter de la valeur. Ainsi les objets connectés appliqués à des domaines forment l'écosystème de l'IOT.

Maintenant qu'on connait les fondamentaux de l'IOT, nous passerons à la deuxième partie de notre travail qui concerne l'étude des différentes étapes de conception du système de simulation réalisé.

Chapitre 2 : Etude de faisabilité et architectures de conception

2.1. Introduction

Si l'internet des objets est sur la voie de révolutionner notre mode de vie et notre environnement, n'est-il pas nécessaire de mettre en place des outils pour faciliter d'avantage sa mise en œuvre dans un environnement sûre et durable ?

L'idée de conception d'un système de simulation est ainsi survenue dans le but de mettre au service de l'IOT un outil de simulation analytique de projets IOT en amont.

La simulation est déjà largement utilisée dans beaucoup de domaines comme l'aéronautique, les réseaux ou le traitement de signal. Et elle fait bien ses preuves d'utilité, alors pourquoi pas en faire profiter à l'internet des objets ?

Cependant, la mise en œuvre de tout projet nécessite au préalable une réflexion, en vue de mettre en place, un meilleur plan de travail, avoir une bonne organisation, et un gain de temps considérable.

C'est dans cette optique nous allons exposer différentes étapes permettant la mise en œuvre du système de simulation.

D'abord nous commençons par un point essentiel relative à l'étude de faisabilité. Ensuite il sera question de la conception des architectures des interfaces du système. Et pour finir nous abordons la partie relative à la programmation, et plus précisément les outils softwares ayant servis à la mise en œuvre du noyau du système et ses interfaces.

2.2. Étude de faisabilité

Avant la mise en œuvre de ce projet, il était nécessaire de se poser une série de questions sur sa faisabilité :

Quel état d'art lié à la simulation dans l'IOT ?

Le projet est-il techniquement possible ?

Y'a-t-il un besoin de notre système dans le monde de l'IOT ?

Avons-nous les compétences nécessaires pour sa réalisation ?

A-t-on besoins de moyens financiers pour sa mise en œuvre ?

Le système de simulation va-t-il apporter de la valeur au monde de l'IOT comme nous le souhaitions ?

La réponse à toutes ces questions va déterminer la faisabilité et la viabilité de notre projet. La viabilité de notre projet fait partie de nos objectifs car le but n'est pas de ranger notre système de simulation dans les archives de l'université Saad Dahleb après la soutenance, mais d'apporter une solution réelle et viable dans le monde des objets connectés. C'est dans ce contexte que nous allons illustrer dans ce qui suit, une description détaillée du projet suivi de l'ensemble de différents domaines de faisabilité.

2.2.1 Etat d'art

Avec l'essor que connait l'internet des objets, les réalisations et innovations faites dans le domaine de la simulation ne cessent d'augmenter. L'état d'art, revient à faire un inventaire des technologies de simulations IOT existantes déjà sur le marché. Ce qui permet de mieux comprendre la problématique posée par le thème en étudiant les travaux antérieurs, les solutions apportées à ce jour et ainsi pouvoir apporter une plus-value à travers notre solution.

Concernant les systèmes de simulation IOT existants, nous pouvons citer quelques-uns :

- **OpenSHS :**

Open Smart Home Simulator (OpenSHS) est un simulateur de maisons intelligentes 3D hybride. C'est un logiciel open source et multiplateforme.

OpenSHS offre une opportunité aux chercheurs dans le domaine de l'Internet des objets (IoT) et de l'apprentissage automatique de tester et

d'évaluer leurs modèles. Suivant une approche hybride, OpenSHS combine les avantages des approches interactives et basées sur des modèles. Cette approche réduit le temps et les efforts nécessaires pour générer des ensembles de données simulés d'une maison intelligente.

Cet outil divise le processus de génération de jeux de données en trois phases distinctes : première conception : l'utilisateur conçoit l'environnement virtuel initial en construisant la maison, en important des appareils intelligents et en créant des contextes ; deuxièmement, la simulation : le participant simule ses événements contextuels ; et troisièmement, l'agrégation : l'utilisateur applique l'algorithme de réplication pour générer l'ensemble de données final.**[7]**

- **Simulateur IoT Bevywise**

Créé en 2016 par Bevywise Networks, Bevywise est un simulateur qui permet de simuler des dizaines de milliers d'appareils IOT.

Axé sur le protocole MQTT, le simulateur Bevywise permet de développer, tester et faire une démonstration des serveurs et gestionnaires IoT, clients MQTT, capteurs MQTT et appareils MQTT. **[8]**

- **SimpleIoTSimulator**

SimpleIoTSimulator est un simulateur de capteurs/appareils IOT qui permet de créer des environnements de test composés de milliers de capteurs et de passerelles, le tout sur un seul ordinateur. SimpleIoTSimulator prend en charge de nombreux protocoles IOT courant. Ils comprennent : MQTT, MQTT-SN qui est une variante de MQTT, le client http, le serveur http, le serveur BACnet, Loragatway pour recevoir des trames Lora. Il permet en effet de simuler les réseaux Lora.**[9]**

- **PowerTOSSIM et PowerTOSSIM-z**

PowerTOSSIM a été mis en place en 2003. C'est un environnement de simulation évolutif pour réseaux de capteurs sans fil qui consiste à estimer

l'énergie consommé par nœuds. PowerTOSSIM est une extension à TOSSIM, une simulation événementielle environnement pour les applications TinyOS **[8]**, plus précisément, PowerTOSSIM détermine la consommation d'énergie de chaque périphérique dans tous les nœuds d'un WSN.**[10]**

- **MATSNL**

En 2007 un package des fichiers m-file de Matlab a été créé pour calculer la durée de vie, le budget et la puissance des nœuds de capteurs sans fils ainsi que pour désigner l'architecture optimale des nœuds. Désormais, ses fonctionnalités s'étendent à l'architecture de plate-forme plus complexe **[11]**. MATSNL compare la consommation de deux plateformes pour la même application.**[12]**

2.2.2. Description et but du projet

La partie précédente a fait l'objet d'une récolte d'informations sur les technologies déjà réalisées ayant un rapport avec la simulation dans l'IOT. En vue d'appréhender le vif du projet.

Nous allons décrire dans cette partie notre système de simulation ainsi que le but recherché :

a) Description du projet :

Ce projet consiste à créer un système de simulation des objets connectés de façon virtuelle. Dans ce système, il est possible de créer un environnement virtuel, tel qu'une maison, une ferme, un atelier, une rue, ou tout type d'environnement que nous souhaitons simuler. Pour chaque environnement, il est possible d'intégrer des locaux comme des chambres, un couloir, une cuisine, un magasin, etc...

Dans chaque local d'un environnement, nous pouvons ajouter des capteurs permettant de simuler l'état de l'environnement. Les capteurs devront être capables de fonctionner de manière continue et autonome sur tout le temps définie par l'utilisateur.

Nous avons ajouté également des actionneurs tel qu'une ampoule, un climatiseur, une chaudière, un thermostat, une alarme etc. chaque actionneur pourrait être

ensuite lié à un capteur afin de recevoir les données et effectuer une action selon l'état de ces données.

Cette solution constitue, une plateforme de test pour les utilisateurs et concepteurs des systèmes intelligents basés sur les IOT.

b) Le but :

Le but de notre projet est de faciliter la tâche à tous ceux et celles qui souhaitent mettre en place une solution IOT, en leur permettant de simuler leur projet en amont afin d'analyser la faisabilité ainsi que la fiabilité de leur installation. Ainsi notre plateforme leur donnera la possibilité d'avoir une vision réelle de leur future installation tout en leurs permettant de prendre les meilleures décisions pour optimiser le rendement et réduire les coûts lors de la mise en œuvre de leur système IOT.

2.2.3 Analyses techniques

Cette section est un point essentiel qui permettra d'exposer une analyse sur les points techniques avant l'implémentation et la conception du système.

a) L'étude sur la nécessité d'un système de simulation

La réponse à cette question est d'une grande importance car il en va de la pertinence de notre thème de mémoire. Pour se faire, le moyen le plus facile pour nous de trouver la réponse à notre question est d'aller à la recherche des analyses des experts dans le domaine de l'iot et de la simulation. C'est ainsi que nous avons découvert une analyse très riche en informations de **MICHEL GENARD,** rapporté par le site « Silicon.fr » [16]. Michel Genard est le directeur général du département **simulation et déploiement IOT** de l'américain « Wind River » qui est une section du géant Intel. Il raconte que vu le nombre d'objets connectés prévus dans les prochaines années (plus de 50 milliards), ce sont des millions de millions de systèmes connectés qu'il va falloir gérer, et donc la simulation va devenir un outil d'infrastructure en amont et en aval prévient-il. Il prend l'exemple de deux de ses clients dont il n'a pas voulu citer les noms. Le premier a déployé tout un réseau de capteurs et Gateway pour

remonter les informations propres aux maintenances d'infrastructures routières et ferroviaires. Le client rencontrait un phénomène de panne bizarre qui revenait tous les 25 à 40 jours de manière inexplicable. Pour pallier au problème, il trouva qu'aller sur le terrain d'un système déployé sur des centaines de kilomètres aurait pris trop de temps et couteux. Un recourt à la simulation virtuelle a permis de détecter et palier au problème en un temps record. Il s'agit d'une simulation en aval. Le deuxième client lui recouru à une simulation en amont. En effet il voulait optimiser la configuration d'un bâtiment connecté en simulant des milliers de capteurs et les passerelles de communication qu'il n'aurait pas pu déployer à l'échelle d'une maquette matérielle. La simulation est un moyen de tester les nouveaux services avec un minimum de risque et donc réduire les coûts lors de la phase de développement, poursuit l'expert dans le domaine [16]. En prenant en compte l'analyse et l'expérience de Michel Genard, on peut se rendre compte de la nécessité de la simulation dans le monde de l'IOT.

Cela permet en plus de réduire les coûts lors de la phase de développement, d'être prévisible vis à vis du comportement de l'environnement IOT dont on souhaite mettre en place.

Nous savons à présent que notre projet sur la conception d'un système de simulation à bien et belle sa place dans le monde de l'internet des objets. Cependant quelles compétences techniques aurons-nous besoin pour sa mise en œuvre ?

b) Quelles compétences pour sa mise en œuvre ? :

Connaitre les prérequis de notre projet est une étape importante pour sa mise en œuvre et aussi un gain de temps éviter les échecs.

Ainsi pour aller au bout de nos réflexions, un tas de questions méritaient d'être posées : quelles sont les compétences nécessaires pour sa mise en œuvre **?** Avons-nous déjà ses compétences **?** Serons-nous capables d'appréhender ses compétences rapidement en vue d'offrir une application de qualité vu sa complexité **?**

Cette partie est une étape essentielle de la mise en œuvre de notre projet car il fallait maitriser les outils et technologies nécessaires à la conception de notre système.

Tout juste, après le choix du thème, avec notre encadreur, nous n'avons pas manqué à solliciter son avis quant aux besoins techniques pour la conception de notre système. Ainsi une liste de compétences dont nous aurions besoin pour la réussite de notre mission a été établie. Nous pouvons citer entre autres :

- Se familiariser avec la notion d'IOT dans sa globalité
- Maitriser un langage de programmation orienté objet : python
- Maitriser particulièrement la programmation orientée objet en python
- Maitriser les langages web basiques (Html, Css)
- Maitriser une base de données SQL
- Savoir comment fonctionne le protocole MQTT
- Savoir comment fonctionne le protocole websocket
- Maitriser le langage de programmation javascript
- Se familiariser avec un serveur MQTT dont Mosquitto

Le gros défi pour nous était de maitriser toutes ces technologies dans un bref délai afin de passer à l'étape suivante. Mais maintenant que nous avons une idée sur les prérequis de notre projet, nous pouvons passer à l'étape de la recherche bibliographique afin de maitriser les technologies indispensables à la mise en œuvre de notre système.

c) Coûts et délais de réalisation :

Tout projet de conception, même dans le cadre d'un mémoire de fin d'étude universitaire, à un coût et un délai de réalisation. La prise en compte de ses deux points était importante dans notre étude de faisabilité car vu les défis à réaliser, il était judicieux de savoir si on aurait besoin d'être financièrement capable pour remplir certaines exigences, et si on serait capable de finir le travail avant les dates prévu.

Notre projet étant basé particulièrement sur la programmation, en dépits du coût des forfaits internet, nous avons décidé de tout apprendre sur internet plutôt que de prendre des cours de programmation en présentiel payant. Ceci a été nécessaire car on devrait apprendre un tout nouveau langage de programmation dont nous n'avons pas eu l'opportunité d'apprendre au cours de notre cursus universitaire. Cependant,

une course contre la montre s'engage parallèlement car on avait le calendrier universitaire qui prévoyait les soutenances en juin. On a dû alors se mettre au travail très vite début février, tout juste après la fin du premier semestre.

2.2.4. Validation

Cette section illustre surtout les raisons pour lesquelles notre projet de conception d'un système de simulation mérite d'être retenu. L'état d'art nous a permis d'analyser et faire sortir l'innovation que vise notre système de simulation par rapport aux technologies de simulation IOT déjà existantes.

Ainsi nous pouvons constater que :

Tous ces systèmes énumérés au niveau de l'état d'art génèrent les données de capteurs de manière aléatoire (juste pour la simulation) sans forcément se rapprocher de la réalité ni se soucier sur des valeurs fausses que peut émettre un capteur.

Alors que notre système offre des simulations de capteurs avec des valeurs plus proches de la réalité avec un taux d'erreur très faible (grâce à une fonctionnalité de régression polynomiale configuré dans les capteurs) qui pouvant même être visualisé graphiquement.

Contrairement à ces systèmes qui simulent les actionneurs indépendamment des capteurs, notre système offre une possibilité de connecter les actionneurs avec les capteurs afin d'effectuer dynamiquement des mouvements selon les données envoyées par les capteurs.

Côté sécurité, notre système, contrairement à ceux existants, intègre une authentification avant le lancement d'une simulation. En effet lors de la simulation un intrus connaissant vos topics de publication peut envoyer des données aléatoires a votre système, ce qui peut compromettre les résultats de la simulation.

Coté flexibilité, notre système offre un choix libre aux utilisateurs de définir les noms de capteurs et actionneur ainsi que les types de données à envoyer. Ainsi l'utilisateur peut même créer un capteur virtuel qui n'existe pas en réalité.

Après avoir pu se démarquer des systèmes de simulation existants sur le marché, notre projet de système de simulation peut enfin être validé pour passer à l'étape de la planification de mise en œuvre.

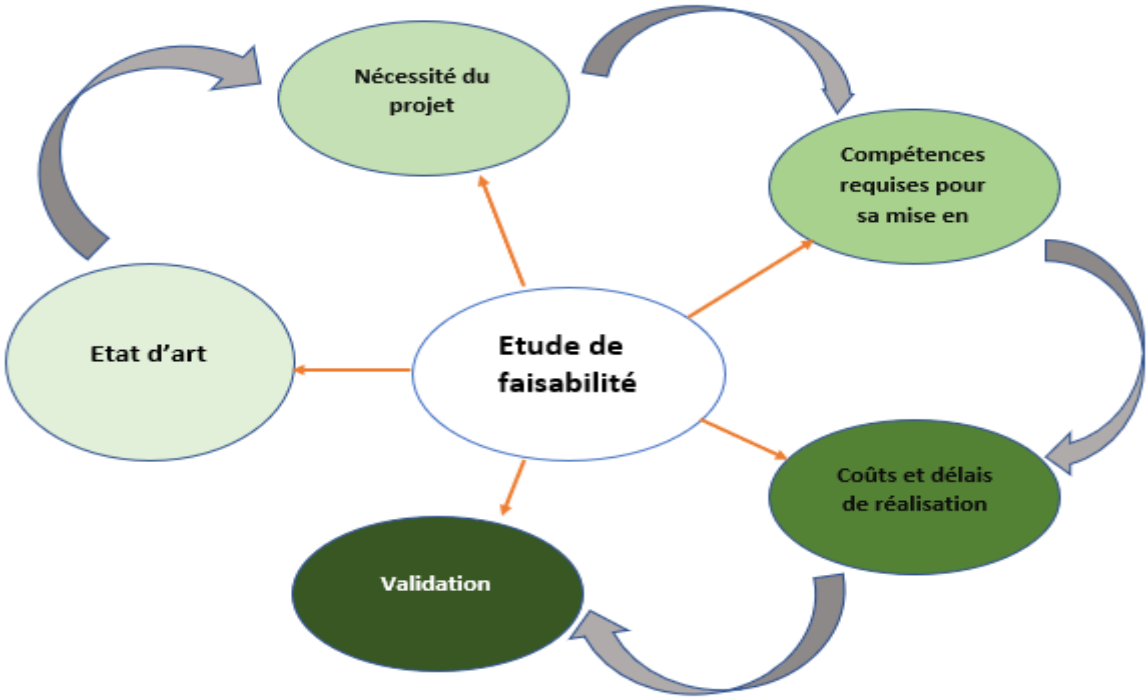

Figure 2.1 : résumé des grands points de l'étude de faisabilité

2.3. Architectures de conception

Avant de se lancer dans la programmation du système de simulation, il est primordial de concevoir l'architecture de ce dernier afin de faciliter sa mise en œuvre. Le système étant composé de de deux interfaces : l'interface principale et l'interface de visualisation.

2.3.1. Architecture de l'interface principale

L'interface principale est l'interface d'accueil servant de base pour la création de nouveaux projets IOT. Son architecture se présente comme suit :

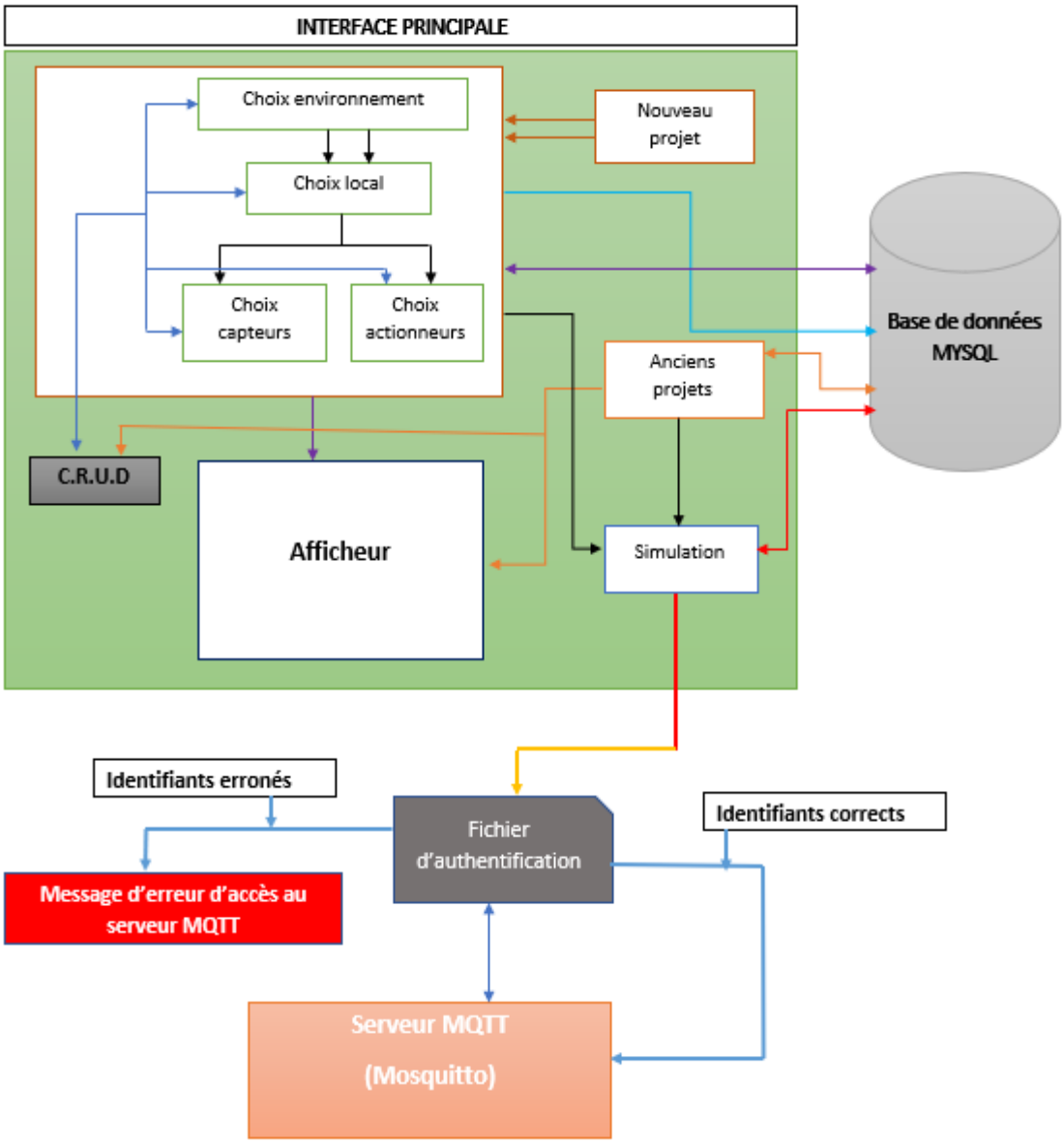

Figure 2.2 : architecture de l'interface principale.

- **Option « nouveau projet » :**

L'option nouveau projet permet à un utilisateur de créer un nouveau projet IOT. Il est relié à la base de données MySQL afin de stocker les éléments du projet. La première étape donne la possibilité à l'utilisateur de choisir ou de créer le type d'environnement dans lequel il souhaite réaliser sa simulation (voir schéma d'architecture).

Après le choix de l'environnement, vient le choix des locaux. Il permet à l'utilisateur de définir les locaux que peut contenir l'environnement choisi dans

l'étape précédente. L'utilisateur à la possibilité d'ajouter autant de locaux qu'il souhaite.

Une fois les locaux définis, l'utilisateur peut maintenant ajouter des capteurs et des actionneurs dans chaque local défini dans l'étape précédente. Il a la possibilité d'ajouter également autant de capteurs et d'actionneurs qu'il le souhaite.

L'option nouveau projet est relié à un afficheur (voir architecture) qui permet à l'utilisateur d'observer l'évolutions de son projet lors de la création.

- **Option « anciens projets » :**

L'option anciens projets permet à l'utilisateur de revenir sur ses anciens projets, de manipuler, effectuer des changements des mises à jour, etc. En effet, relié à la BDD, l'utilisateur peut retrouver ses anciens projets en un clic. Elle est reliée à l'afficheur pour observer les changements lors de la manipulation.

- **Le module C.R.U.D :**

D'appellation anglaise, C.R.U.D signifie : Create Read Update Delete (Créer, Lire, Mettre à jour, Supprimer). Il est relié aux options nouveau projet et anciens projets. En effet c'est ce module qui permet de lire, modifier ou supprimer les éléments d'un projet lors de la création ou après la création.

- **Option « simulation » :**

L'option permet à l'utilisateur, après avoir créé un projet de lancer la simulation et observer les résultats. Relié à la base de données, elle récupère toutes les informations concernant le projet lancé en simulation pour les envoyer vers le serveur MQTT Mosquitto.

ELEMENT	FONCTIONNEMENT
Nouveau projet	Permet de créer un nouveau projet : -choix d'environnement - choix des locaux - choix de capteurs -choix d'actionneurs
Anciens projet	Permet à l'utilisateur de revenir sur ses anciens projets avec possibilité d'effectuer des changements
C.R.U.D	Permet de réaliser les tâches suivantes dans un projet : -Créer des données -lire des données -mettre à jour des données - supprimer des données
MySQL	Représente la base de données dans la quelles est enregistrés tous les éléments des projets
Afficheur	Permet d'afficher les projets dans l'interface graphique
Simulation	Permet de lancer la simulation une fois qu'un projet a été créé.

Tableau 2.1: tableau résumant les éléments de l'architecture de l'interface principale

Quand la simulation est lancée, avant de recevoir les données, le serveur Mosquitto vérifie dans son fichier d'authentification si les identifiants de l'utilisateurs y figurent. Si oui, les données sont acceptées par le serveur, sinon elles sont rejetées et un message d'erreur sera affiché pour montrer à l'utilisateur que ses identifiants sont incorrects.

2.3.2. Architecture de l'interface de visualisation

Nous avons vu précédemment l'architecture de l'interface principale de notre système de simulation qui permet de créer un projet IOT virtuel. Dans cette section, nous allons décrire l'architecture de la seconde interface qui constitue l'interface de visualisation de notre système (Voir figure suivante), cette interface permet de suivre et afficher en temps réel l'évolution du système en cours de simulation.

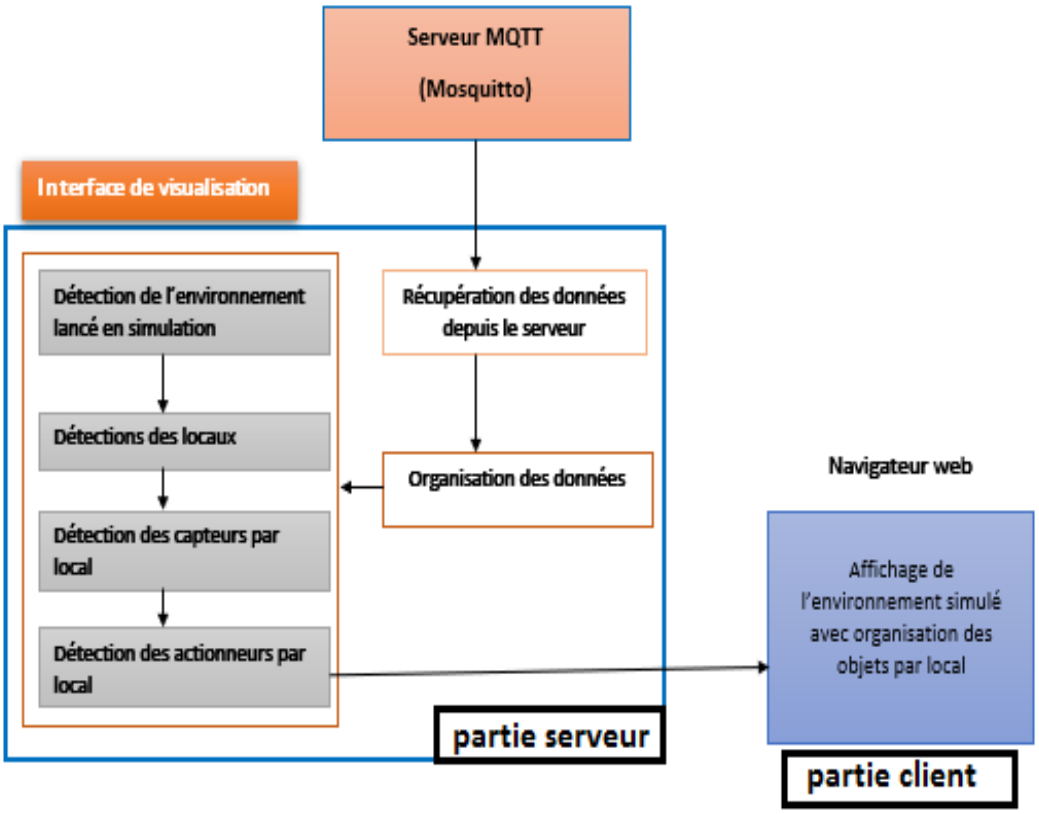

Figure 2.3 : Architecture de l'interface de visualisation

L'interface de visualisation est reliée au serveur (Broker MQTT) en backend. Il récupère les données envoyées par l'interface principale sur le broker. Ses données sont ensuite réorganisées pour enfin être visualisées dans le navigateur web. Ainsi l'utilisateur peut observer les résultats de la simulation.

2.3.3. L'architecture d'ensemble

L'architecture d'ensemble du système se résume comme suit :

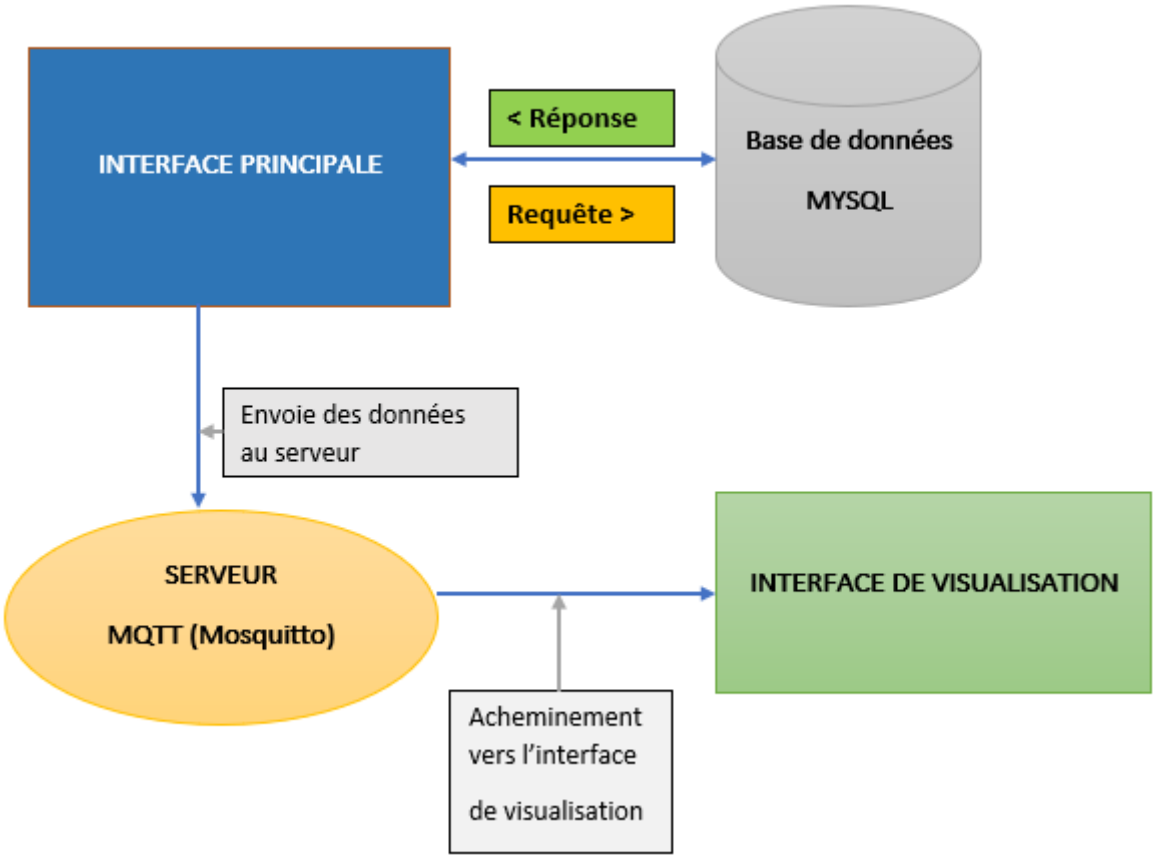

Figure 2.4 : Architecture résumée du système de simulation

En résumé, le système de simulation est constitué d'une interface principale relié à une base de données MySQL permettant de créer et enregistrer des projets. Lorsqu'une simulation est lancée à partir de l'interface principale, les données sont récupérées de la BDD pour être transférés au broker MQTT Mosquitto. Le broker MQTT va à son tour envoyer ses données à l'interface de visualisations pour observer les résultats.

Le serveur (Mosquitto) du système étant le même type de serveur utilisé dans la mise en œuvre d'un projet réel d'IOT, le système pourra mieux simuler le comportement réel des objets.

Dans la prochaine partie, nous aborderons la programmation finale des interfaces du système.

2.4. Programmation des interfaces

L'étude sur les architectures des différentes interfaces nous a permis de voir en amont la composition du système et faciliter sa mise en œuvre. Dans cette partie, nous aborderons la programmation des différentes interfaces.

2.4.1. Python et MySQL pour l'interface principale

Python est un langage de programmation orienté objet dont la première version a vu le jour le 20 février 1991. Il est placé sous une licence open source c'est-à-dire libre de circulation. C'est un langage puissant et dont un code unique peur fonctionner sur tous les systèmes d'exploitation d'où son appellation de langage multiplateforme.

Python est un langage très puissant, mais aussi plus facile à maitriser que les autres langages POO tel que java ou C++. En outre python, parce qu'il regorge de nombreuses bibliothèques graphiques dont Tkinter qui permettent de programmer directement des interfaces graphiques.

Tkinter, de l'anglais « Tool Kit Interface », est une bibliothèque graphique qui permet de réaliser des applications graphiques en python. La force de tkinter est qu'une application réalisée sur Windows va fonctionner sur tous les autres systèmes d'exploitation comme linux, mac et vice versa sans devoir à changer une seule ligne de code. Ainsi Tkinter nous a permis de coder toute la partie graphique de la première interface.

L'interface principale est liée à une base de données. C'est là qu'intervient un nouveau langage : SQL ?

SQL est un langage de requête structuré permettant de gérer les bases de données relationnelles.

Alors qu'est-ce que MySQL ?

MySQL n'est qu'un système de gestion de base de données relationnelles (SGBDR) permettant de récupérer, modifier, et administrer une base de donner en utilisant SQL. Dans notre application, MySQL nous a permis de gérer les données manipulées dans l'interface principale (les environnements, les locaux, les capteurs, les actionneurs et les liens entre eux).

En résumé, la bibliothèque graphique Tkinter de python a permis de développer l'interface principale graphiquement et MySQL pour connecter cette interface à une base de donner via le langage SQL.

2.4.2. Javascript pour le backend de l'interface de visualisation

Javascript est un langage de programmation de scripte employé surtout dans la programmation de pages web interactives. Il est parfois considéré comme l'une des technologie cœur du world wide web (www).

Javascript est intéressant pour sa légèreté et est directement intégré aux navigateurs web. Ainsi, on n'a pas besoin d'installer une quelconque dépendance pour faire fonctionner JavaScript. Il suffit juste de disposer d'un navigateur comme Firefox ou google chrome par exemple.

Grace à une extension du serveur MQTT, javascript permet de coder le backend de l'interface de simulation (récupérer des données depuis le broker MQTT).

2.4.3. HTML et CSS pour le frontend

De façon générale, le HTML permet l'écriture du texte brut sans mise en page avancée. C'est le CSS qui assure la partie mise en page. **Cascading Style Sheets** (**CSS**) est un langage de feuille de style utilisé pour décrire la présentation d'un document écrit en HTML. CSS décrit la façon dont les éléments doivent être affichés à l'écran. C'est un des langages principaux du web et a été standardisé par le W3c.

Le HTML et le CSS ont servi à la programmation de l'interface du navigateur permettant de décorer les données brutes reçues depuis le serveur grâce à javascript. Cela pour mieux visualiser la simulation.

Langage	Atouts	Utilité pour le projet	Utilisé pour l'interface :
PYTHON	- open source - langage POO - très puissant - facile à apprendre - contient des bibliothèques graphiques - stabilité	- développer l'interface principale de manière graphique -créer de nouveaux projets - communiquer avec la base de données MySQL et le serveur mqtt	Principale
JAVASCRIPT	- Une rapidité accrue - directement supporté par les navigateurs web. - très puissant en frontend -stable	Communiquer avec le server MQTT, récupérer les données dans le but de les afficher	Secondaire
HTML ET CSS	- facile à apprendre - langage web - très populaires	Afficher les résultats de la simulation sur une page web.	Secondaire

Tableau 2.2 : résumé des langages, leurs atouts, leurs utilités pour le projet et l'interface d'utilisation.

2.5. Conclusion

Avant tout début de mise en œuvre d'un projet, prendre le temps de planifier les étapes de conception permet non seulement de mieux s'organiser mais aussi de gagner du temps.

Dans ce second chapitre, nous avons présenté l'étude de faisabilité dans un premier temps. Ce qui nous a permis de faire non seulement un inventaire des projets de système de simulation IOT existants, mais aussi trouver des réponses à des questions techniques liés à notre projet.

Nous avons élaboré les architectures de conception des différentes interfaces de notre système de simulation. Cela a permis de visualiser en amont les fonctionnalités de notre système de simulation avant même sa mise en œuvre.

Après l'élaboration de l'architecture du système, nous somme passer finalement à la programmation du système de simulation grâce aux outils de programmation python, MySQL, HTML, CSS et Javascript.

Maintenant que le système de simulation est prêt, nous pouvons passer à la prochaine étape relative à l'utilisation pratique du système. Ceci nous permet de valider le système conçu en rapport avec l'étude conceptuelle.

Chapitre 3 : implémentation et résultats de simulation

3.1. Introduction

La simulation d'objets connectés revient à reproduire l'architecture de ces objets IOT sans utilisation d'équipements physiques. Elle prédit les évènements qui seraient amenés à se produire en prenant en compte leurs caractéristiques.

Le but de ce chapitre est de présenter le fonctionnement du système de simulation final mis en œuvre. Cela va nous permettre d'avoir un comparatif entre les études théoriques effectuées précédemment et la version finale obtenue de notre système de simulation.

Dans ce contexte, nous allons présenter d'abord la gestion des communications dans l'environnement global du système de simulation en vue de comprendre comment sont acheminées les données.

Ensuite nous allons présenter les différentes phases de création d'un projet IOT allant du choix d'environnement, de locaux, de capteurs à actionneurs comme nous l'avons bien abordé dans le chapitre précèdent concernant la partie architecture.

Enfin, présenterons les résultats de simulation suivant les variations de données ainsi que l'aspect sécurité pour assurer authenticité des données.

3.2. Gestion des communications entre les interfaces du système

La gestion des communications entre les interfaces du système se fait grâce à des protocoles de communication gérés par un serveur centralisé. Nous verrons dans cette partie les protocoles qui serviront à transporter les données ainsi que le serveur qui sert à gérer ses données.

3.2.1. Le protocole MQTT

MQTT est un **protocole de transfert de données Machine-to-Machine (M2M)** basé sur la technologie TCP/IP. Il fournit une méthode légère d'exécution de la messagerie à l'aide d'un modèle de publication / abonnement. Cela le rend adapté à la messagerie Internet des objets, par exemple avec des capteurs de faible puissance ou des appareils mobiles tels que des téléphones, des ordinateurs intégrés ou des microcontrôleurs.

MQTT a été développé à la base pour avoir un protocole de messagerie léger pour faire communiquer des machines dans un environnement où les déconnexions sont fréquentes. Ce qui lui donne une grande utilité dans l'IOT, car les déconnexions des capteurs ne sont pas rares et il faut quand même pouvoir recevoir les informations lors de la reconnexion.

De nombreux objets connectés destinés aux particuliers, ainsi que des applications comme Facebook Messenger, reposent sur le MQTT. De même **Amazon IoT se base sur ce protocole.** En général, ce protocole est le plus adapté pour les systèmes de contrôle utilisés par les entreprises industrielles. Son taux d'adoption devrait continuer à augmenter dans les années à venir.

La légèreté et l'efficience du MQTT permettent d'augmenter la quantité de données surveillées ou contrôlées. Auparavant, **près de 80% des données restaient à des emplacements distants**. Il n'était donc pas possible de les exploiter. Grâce au MQTT, il est désormais possible de collecter, de transmettre et d'analyser bien plus de données. En outre, MQTT est un protocole bidirectionnel publish/subscribe permettant aux appareils de publier vers un broker. Les clients se connectent ensuite au broker qui joue le rôle de mandateur entre les deux appareils. Avec MQTT, vous avez la possibilité de régler le QOS (Quality Of Service – Qualité de Service), c'est à dire que pour chaque message envoyé vous pouvez choisir comment le broker doit le gérer :

QoS0 : Le message envoyé n'est pas stocké par le Broker. Il n'y a pas d'accusé de réception. Le message sera perdu en cas d'arrêt du serveur ou du client. C'est le mode par défaut

QoS1 : Le message sera livré au moins une fois. Le client renvoie le message jusqu'à ce que le broker envoi en retour un accusé de réception.

QoS2 : Le broker sauvegarde le message et le transmettra jusqu'à ce qu'il soit reçu par tous les souscripteurs connectés.

MQTT consomme 11 fois moins d'énergie pour envoyer des messages et 170 fois moins pour en recevoir que le protocole HTTP. Il est également 93 fois plus rapide que le protocole HTTP.

Le système de simulation a été développé afin de gérer les données de la même manière que les données sont gérées dans un système réel. Ainsi dans un premier temps, MQTT est utilisé dans l'implémentation de l'interface principale comme protocole de transport des données vers le serveur Mosquitto. Il sera ensuite complété par le protocole websocket pour acheminer les données vers l'interface de visualisation.

3.2.2. Le protocole websocket

Le protocole websocket, est un protocole réseau qui vise à développer un canal de communication full-duplex entre le navigateur et le serveur web. Il a été standardisé en 2011 et est intégré dans tous les navigateurs web modernes. Similairement au protocole MQTT, le protocole websocket est basé sur le TCP/IP ce qui le rend largement portable.

Nous avons vu précédemment que MQTT est le protocole idéal pour l'internet des objets. Alors pourquoi ne pas utiliser directement MQTT que d'utiliser websocket ?

La raison est que, à l'heure où nous somme, il est impossible de communiquer avec un navigateur web via le protocole MQTT. Et nous avons vu que l'interface de visualisation du système de simulation va fonctionner via un navigateur web. Nous aurons donc besoin du protocole websocket pour communiquer avec le serveur Mosquitto que nous verrons dans le point suivant.

En outre, ce que l'on doit retenir est que ce n'est pas le protocole websocket qui va transporter les messages mais toujours le protocole MQTT. Le protocole websocket va servir juste à encapsuler les messages (frames) MQTT pour permettre aux navigateurs web de les interpréter.

3.2.3. Serveur Mosquitto

Mosquitto est un serveur MQTT open source que l'on peut installer sur tous les systèmes d'exploitation (Mac, Windows, Linux, etc.). Contrairement au principe du client/serveur utilisé sur le web, MQTT utilise celui de la publication/souscription. En effet, Mosquitto sert de broker unique permettant à plusieurs clients connectés de soit publier des informations, soit souscrire pour recevoir des informations. On parle de ce fait de topic.

Les topics sont des canaux (chemins) d'accès à une ressource. Les clients de souscription MQTT s'enregistrent alors auprès du broker via ces topics. Un client inscrit à un topic demande alors au broker à être notifié lorsque quelqu'un publie sur ce topic. Ainsi, un client peut s'inscrire à plusieurs topics.

Cependant, en ce qui concerne le système de simulation on devrait en plus configurer le broker Mosquitto à supporter le protocole websocket car celui-ci n'est pas supporté par défaut. Rien de plus facile, il suffit d'ajouter dans le fichier « Mosquitto.conf » qui se trouve dans le dossier d'installation du serveur les deux dernières lignes :

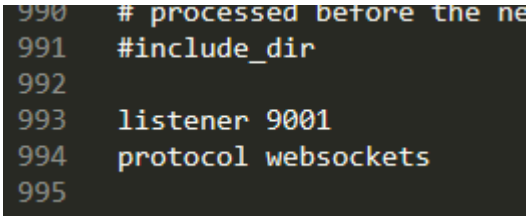

Figure 3.1 : configurer websocket avec Mosquitto

On a : « listener » suivit du port d'écoute du protocole websocket et le nom du protocole au pluriel. Dans notre cas on a choisi 9001 mais rien n'empêche de choisir un autre numéro de port pourvu qu'il soit en dessus de 1024.

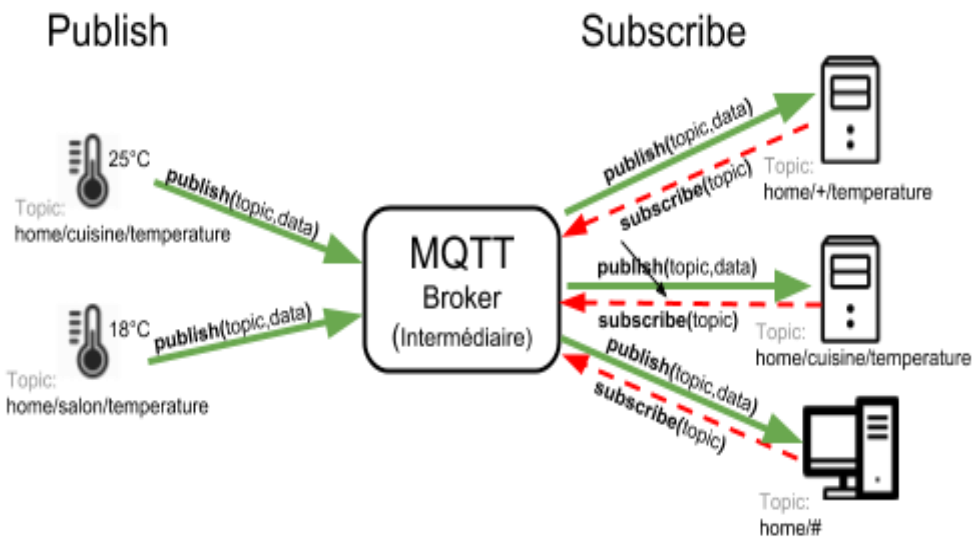

Figure 3.2 : fonctionnement d'un serveur MQTT [17]

3.3. Structure du programme et présentation des interfaces

Une note explicative sur la structure du programme va permettre de comprendre comment est structuré le programme source du système. Ceci permettra également de comprendre comment le programme est exécuté dans son ensemble.

3.3.1. Structure du programme

Dans cette partie, nous allons faire une description technique du code source du système de simulation.

Le programme source de l'interface principale est composée de deux classes, c'est-à-dire une programmation orientée objet. Une classe est un ensemble incluant des variables (attributs) et des fonctions (méthodes).

On distingue :

- **La classe « application » :**

```
38
39 ▼ class Application(tk.Tk):
40 ▼     def __init__(self):
41           tk.Tk.__init__(self)
42           self.frams()
43           self.Menue()
44           self.etat1()
45           self.zone_text()
46           self.suprime()
47
```

Figure 3.3 : déclaration de la classe application

Elle constitue le noyau du programme englobant toute la structure graphique (les boutons, les fenêtres toplevel ainsi que les boites à liste, etc.) de l'application. Cette classe gère tout ce qui est éléments graphiques du programme de base.

- **La classe « thread_class »**

```
1151
1152 class Thread_class(threading.Thread):
1153     """docstring for Thread_class"""
1154     def __init__(self):
1155         global i_thread
1156         global client1
1157         global secu
1158         secu = False
1159         # super(Application, self).__init__()
1160         threading.Thread.__init__(self)
1161         times = True
1162         self.user_input = simpledialog.askstring(title="Test", prompt="What's your Name?:")
1163         self.user_passwd = simpledialog.askstring(title="Test", prompt="What's your password?:")
1164         self.client1= paho.Client("control2")
1165         self.client1.username_pw_set(username=self.user_input,password=self.user_passwd)
1166
1167
1168     def on_connect(self, client, userdata, flags, rc):
```

Figure 3.4 : déclaration de la classe thread_class

Le threading est une technique de programmation qui permet d'exécuter indépendamment les parties d'un programme. Vu que notre programme devrait fonctionner en boucle infini lors de la simulation, on devrait trouver une alternative de faire exécuter le programme sans le planter car il est bien connu

de la programmation qu'une boucle infinie fait planter un programme. C'est ainsi qu'on a créé la classe thread_class qui a pour objectif de gérer uniquement la partie simulation qui doit fonctionner en boucle infini.

En somme, toute l'interface principale est codée sur deux classes : la classe « application » permettant de gérer les widgets graphiques et la classe « thread_class » pour gérer et assurer la simulation en boucle infinie dans le temps.

Quant à l'interface de visualisation, le code source n'est que du langage web basique et des scripts javascript :

Figure 3.5 : photo illustrative du code source de l'interface de visualisation

3.3.2. Présentation des interfaces

Dans cette partie nous présentons les interfaces du système.

❖ **L'interface principale**

L'interface principale est l'interface d'accueil du système pour la création de projets.

Elle est composée de 4 sections (voir figure ci-dessous) : une section supérieure (**cadre1**) en haut contenant l'ensemble des boutons permettant de démarrer un nouveau projet, reprendre un projet existant, un bouton « Run » pour lancer la simulation et un bouton « stop » pour arrêter la simulation. Par

défaut, la majorité des boutons sont désactivés. Il faut ouvrir un nouveau projet pour accéder à ces boutons.

Nous avons en outre la section de gauche (**cadre2**) en bleue ciel qui va constituer la partie où seront affichées les listes des environnements, des capteurs et des actionneurs. Ensuite une troisième section en blanc (**cadre3**) là où sera affichée la liste des locaux lors de la création d'un projet. Enfin, la section au centre (**cadre4**) va permettre d'observer l'évolution de son projet lors de sa création et aussi effectuer des manipulations sur les éléments d'un projet. (**Voir figure**) :

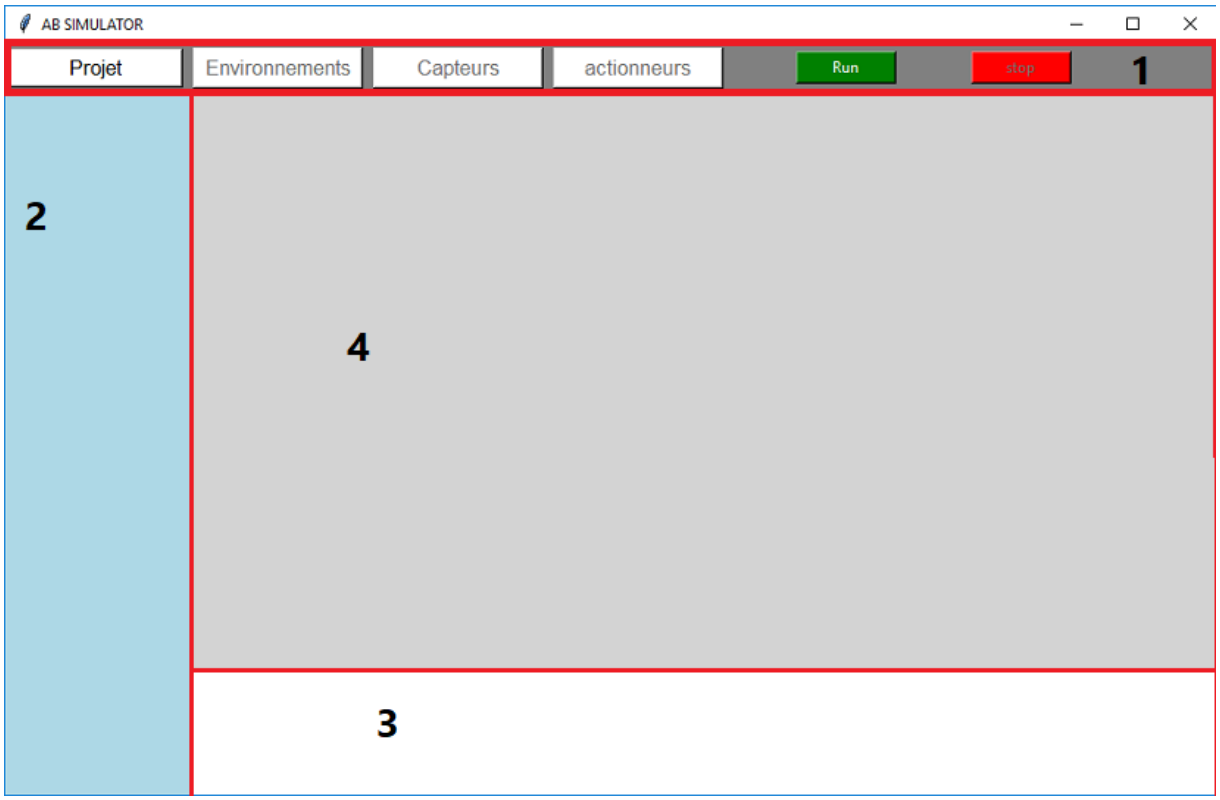

Figure 3.6 : interface d'accueil du système de simulation.

❖ **L'interface de visualisation**

L'interface de visualisation représente l'interface d'observation des résultats de simulation.

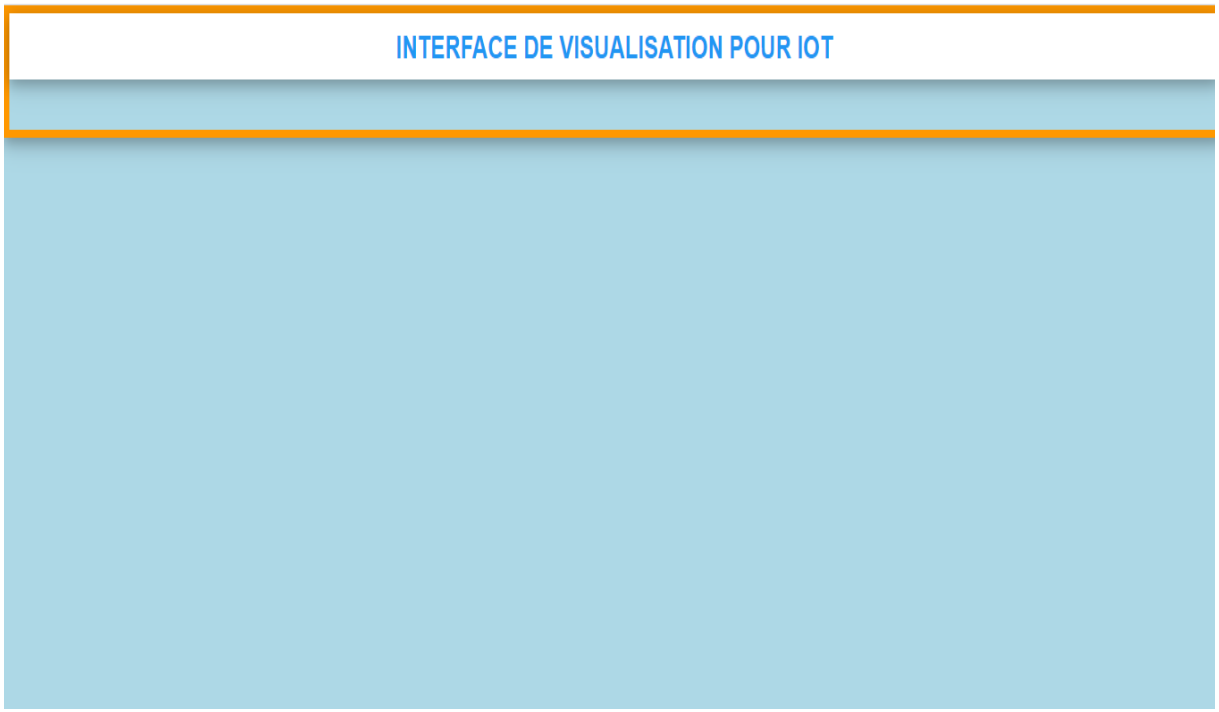

Figure 3.7 : interface de visualisation du système avant simulation

L'interface de visualisation a pour rôle d'afficher uniquement les objets IoT lancés en simulation, donc aucune manipulation ne se fait sur cette interface.

3.4. Phase de conception d'un projet

Cette phase montre les différentes étapes de création d'un projet IOT. C'est-à-dire comment choisir un environnement, comment ajouter des locaux à un environnement et comment ajouter des capteurs et des actionneurs à un local.

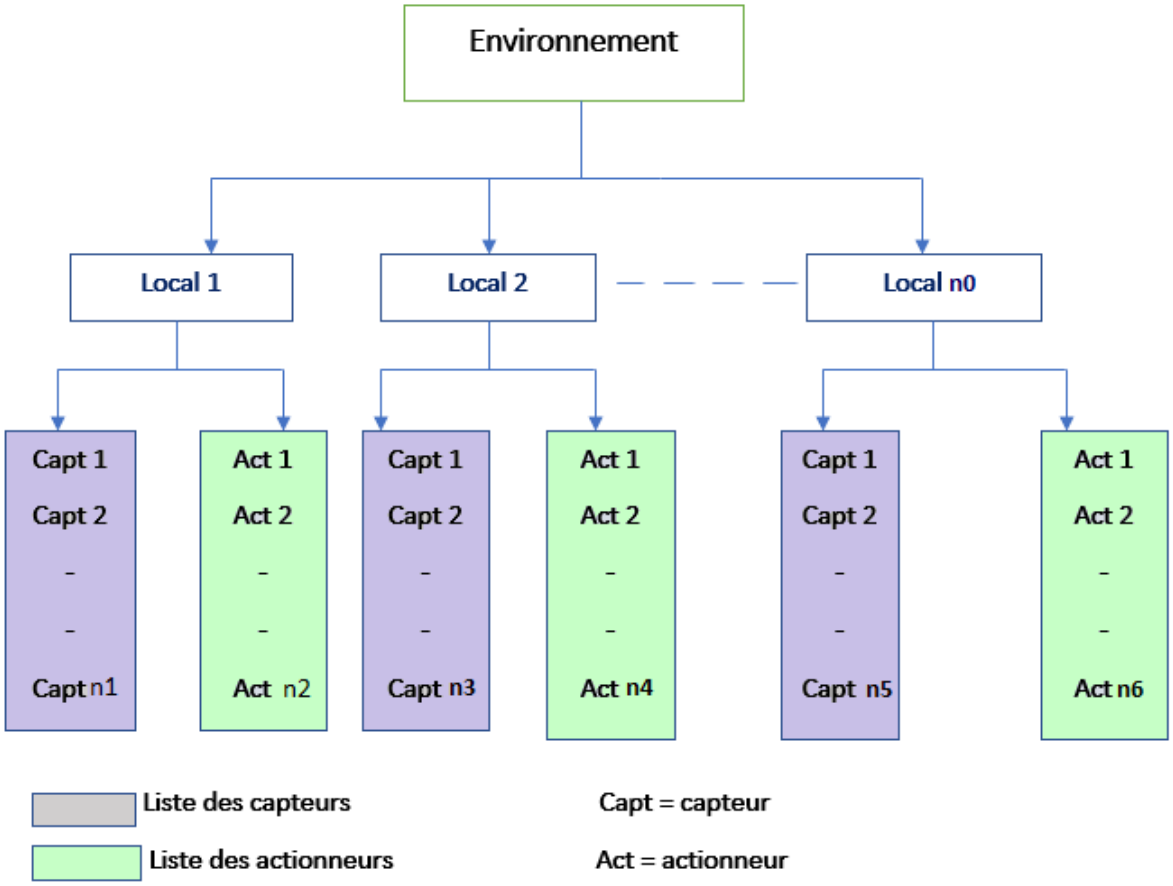

Figure 3.8 : architecture de conception d'un projet

3.4.1. Choix d'environnement

L'environnement désigne l'espace ou le type de lieu où on souhaite installer les objets IOT. Cela peut être un habitat, une ferme, un champ, une usine, etc. il faut donc choisir son environnement de simulation avant toute chose. Cependant pour débuter vous devez d'abord démarrer un nouveau projet depuis l'interface d'accueil.

Il faut commencer par ouvrir une session de projet en faisant un clic sur « projet » puis « nouveau projet » (voir image ci-dessous) :

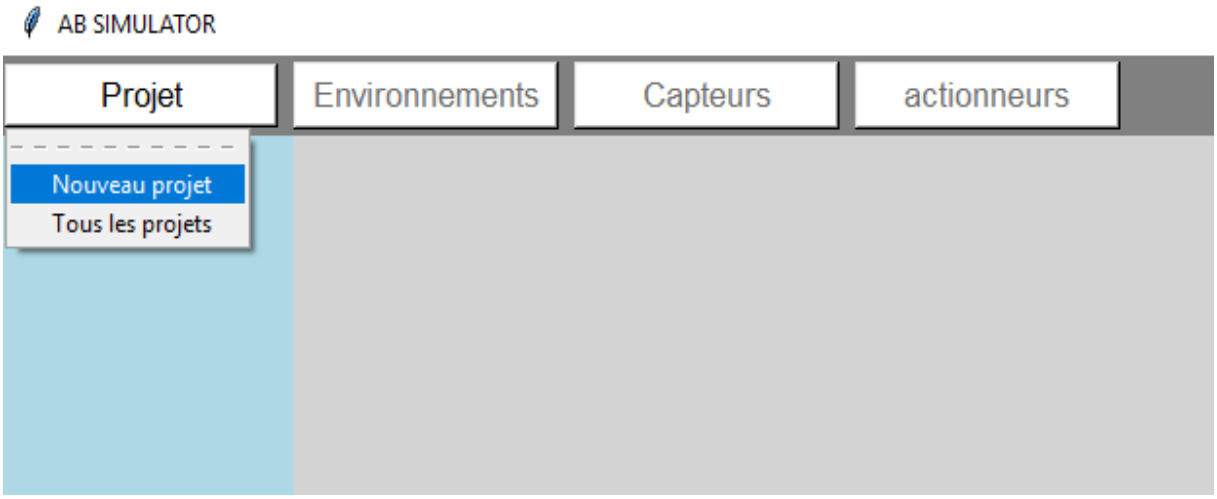

Figure 3.9 : ouverture d'une session pour un nouveau projet

Lorsqu'on clique sur nouveau bouton, on observe une activation des boutons qui au début étaient désactivés. Ce qui indique qu'on vient d'activer une session de création de projet. En plus, il y'a apparition de sections supplémentaires au centre :

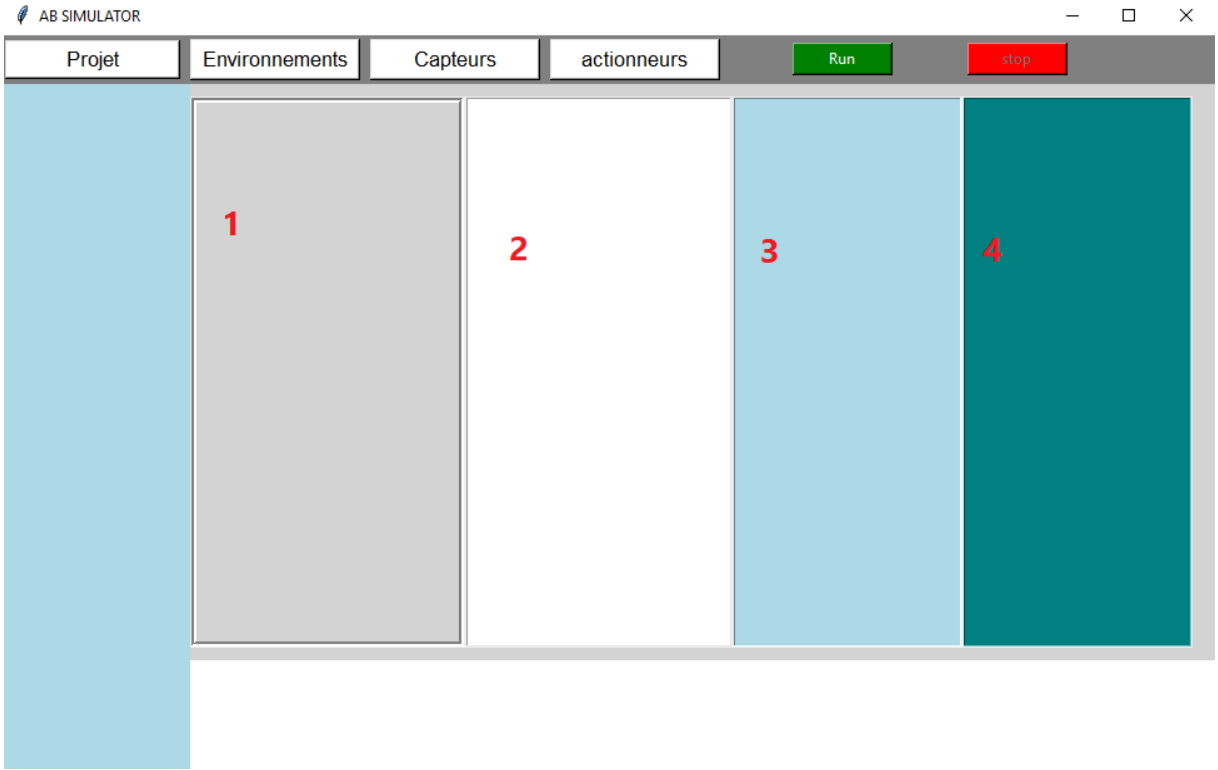

Figure 3.10 : l'interface après ouverture d'un nouveau projet

Comme indiqué sur l'image ci-dessous, 4 nouvelles boites de listes qui apparaissent au centre. Chaque boite de liste à un rôle particulier. La boite de liste

une (indiqué 1 sur l'image) en grise permet d'afficher l'environnement sur lequel vous êtes en train de travailler. La boite de liste deux en blanc permet d'afficher tous les locaux que vous allez attribuer à l'environnement choisi. La boite de liste trois en bleue ciel permet d'afficher tous les capteurs que vous ajoutez à un local. En fin la dernière boite de liste permet d'afficher tous les capteurs appartenant à un local donné.

Pour ajouter un environnement, cliquez sur le bouton « environnements » et vous avez la liste des environnements par défaut affichée à gauche.

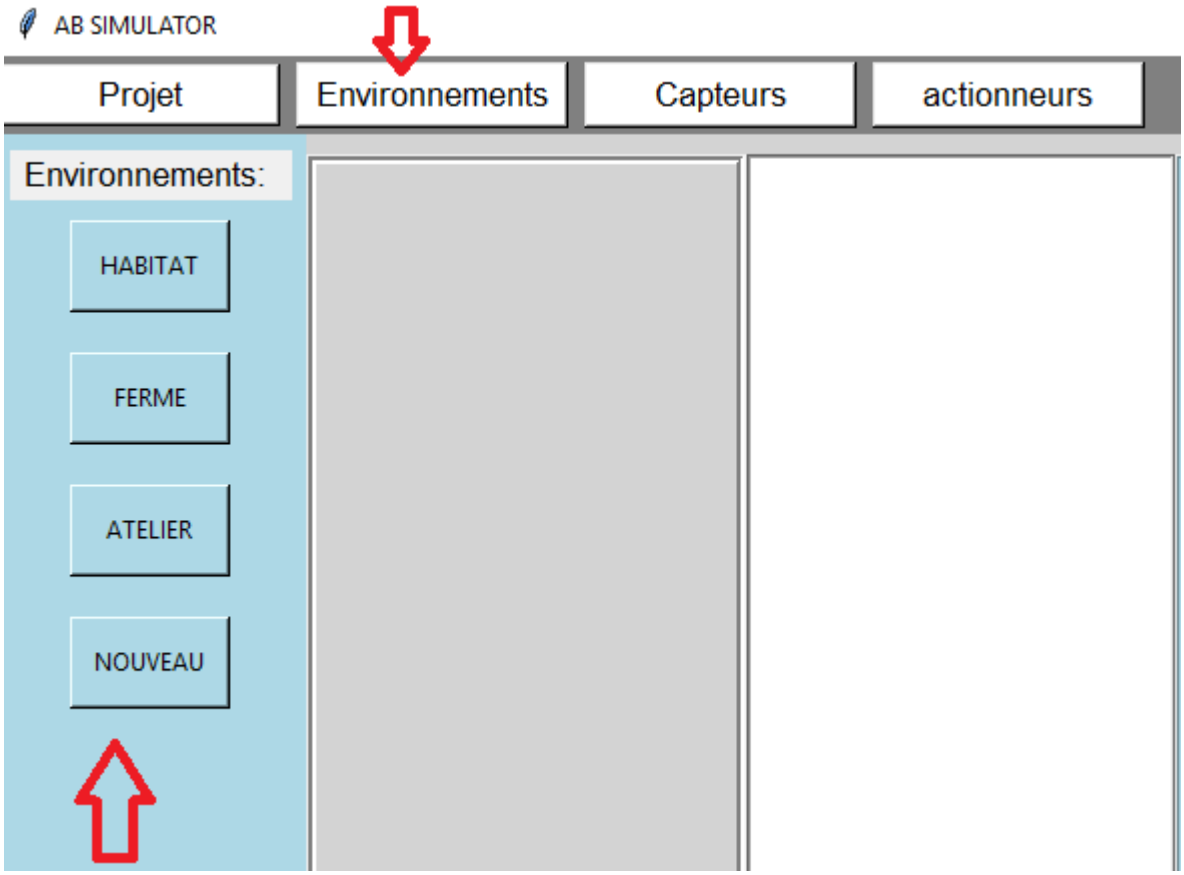

Figure 3.11 : listes des environnements de choix par défaut

Il suffit ensuite de cliquer sur un environnement de votre choix. Le dernier environnement « NOUVEAU » vous permet de définir un nouvel environnement n'existant pas dans la liste. Lorsque vous cliquez par exemple sur « NOUVEAU », vous venez d'ajouter un environnement à votre projet et vous verrez le nom apparaitre

dans la première boite de liste. Vous pouvez modifier le nom en double cliquant sur le nom dans la boite de liste afin de définir un nouveau nom à votre environnement.

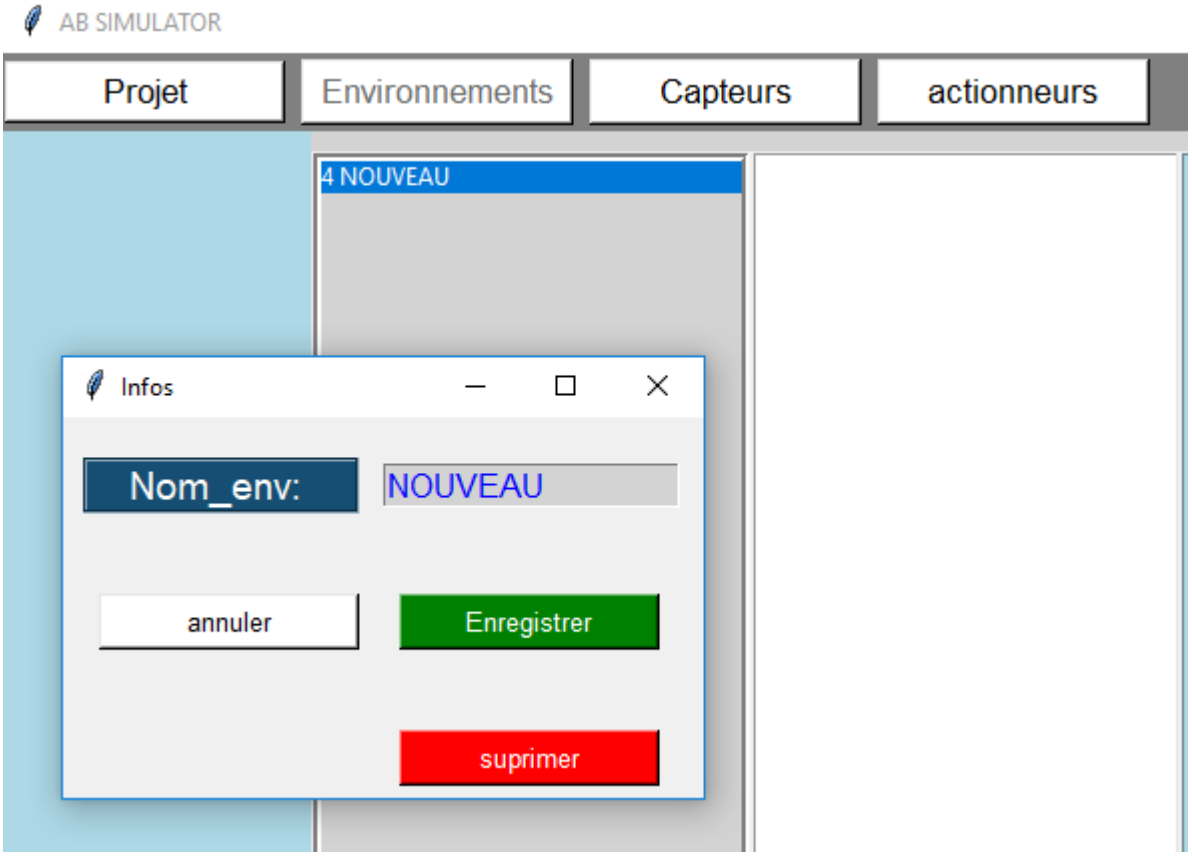

Figure 3.12 : modifier le nom d'un environnement

Une fois l'environnement défini, vous pouvez passer à l'étape de définition des locaux.

3.4.2. Choix des locaux

Le local représente une partie intégrante de l'environnement. Par exemple une habitation est composée de chambres, de cuisine, de salle à manger, etc. une ferme peut comporter un distributeur de vivre, un système à eau, etc. donc un local est toute structure physique mobile ou immobile entrant dans la constitution d'un environnement.

Une fois le choix de l'environnement fait, vous pouvez ajouter un ou plusieurs locaux à votre environnement. Lorsque vous sélectionnez un environnement, vous obtenez par défaut, une liste de locaux qui apparaissent tout juste en bas de l'interface.

Pour ajouter un local, il suffit de cliquer sur un des locaux et vous aurez une petite fenêtre toplevel où vous pouvez ajouter le local ou annuler l'action

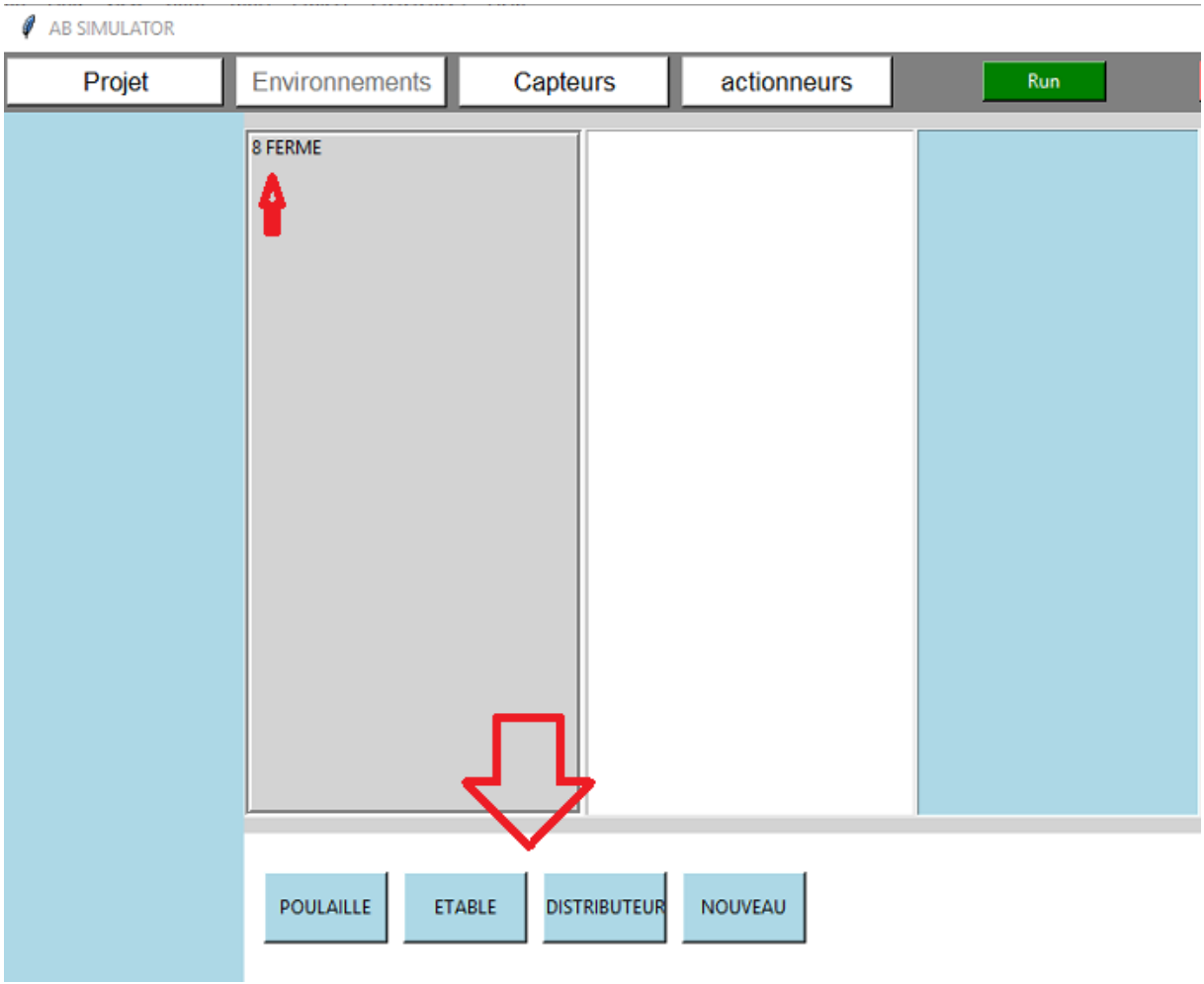

Figure 3.13 : liste des locaux

Ici nous avons une ferme comme environnement donc on obtient une liste par défaut de locaux liés à une ferme. Le numéro **« 8 »** devant le nom de l'environnement indique son identifiant dans la base de donnée. Le local « NOUVEAU » permet de définir un nouveau local qui n'existe pas dans la liste. Lorsque vous ajoutez un local, celui-ci apparait dans la seconde boite à liste comme le montre la figure suivante :

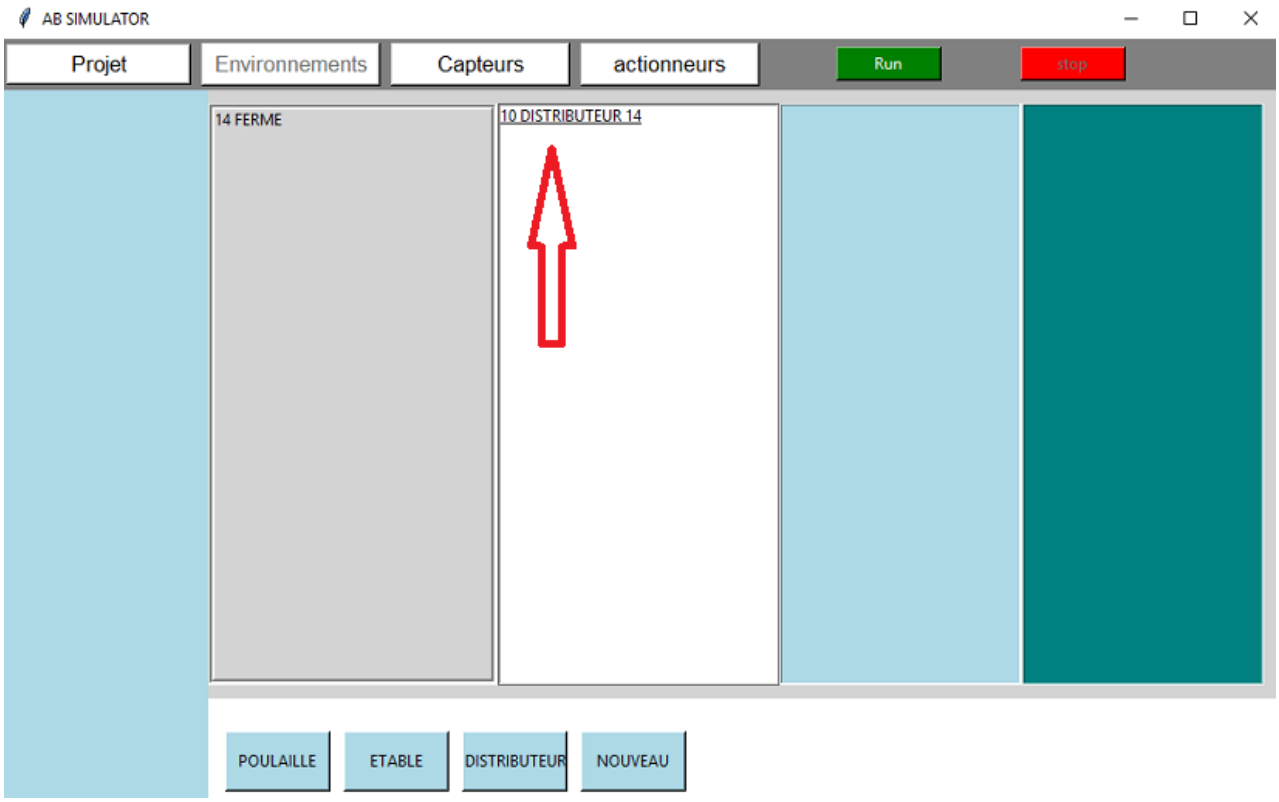

Figure 3.14 : ajout d'un local a un environnement

A gauche on à l'environnement et ensuite, la liste des locaux, ce qui vous donne une bonne visibilité de l'évolution dans la création de votre projet. Vous pouvez également changer le nom d'un local en double cliquant dessus, vous obtenez une boite de dialogue pour changer le nom à votre guise. Vous pouvez ajouter autant de locaux que vous souhaitez à votre environnement.

Nous verrons dans la prochaine étape comment ajouter des capteurs et des actionneurs à un local. Mais vous pouvez par ailleurs ajouter les capteurs et actionneurs au fur et à mesure que vous ajoutiez des locaux.

3.4.3. Choix de capteurs et actionneurs

Maintenant que vous avez ajouté un ou plusieurs locaux, vient l'étape d'ajouter des objets (capteurs et actionneurs) à vos locaux. Dans l'interface principale, vous avez des boutons « capteurs » et « actionneurs » qui vous permettent après un clic de lister les capteurs ou actionneurs par défaut.

Vous pouvez ensuite sélectionner dans la liste de capteurs ou actionneurs les objets que vous aimeriez intégrer dans vos locaux.

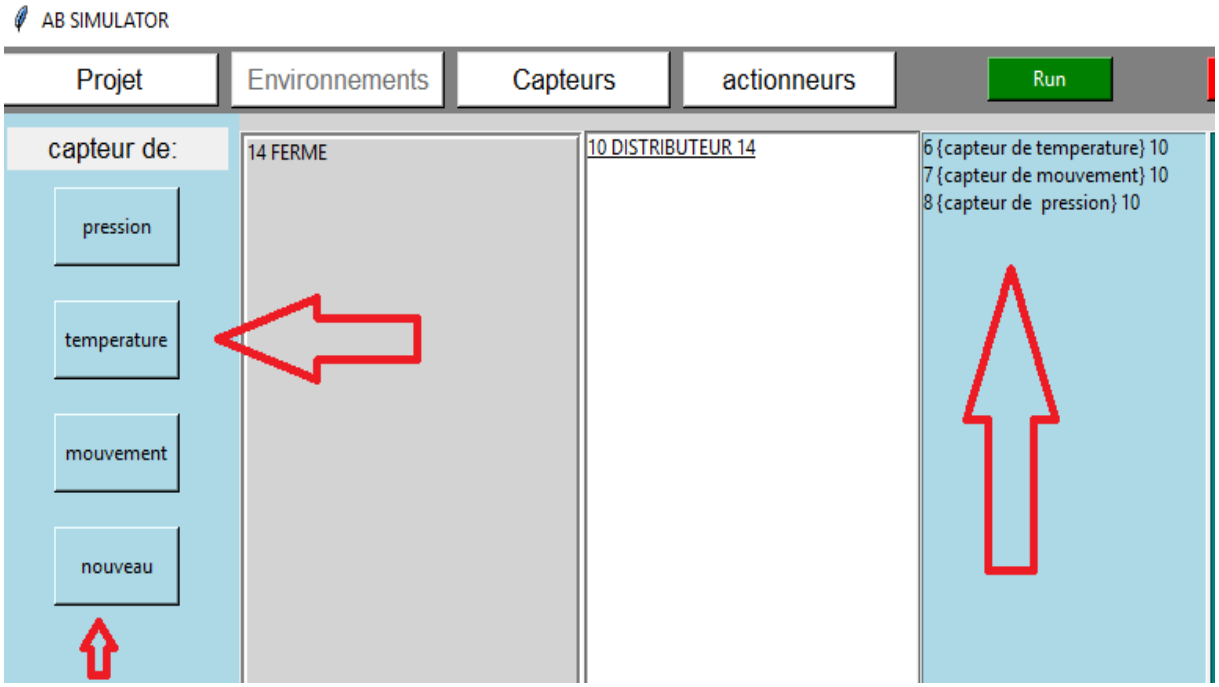

Figure 3.15 : liste des capteurs par défaut

L'image ci-dessus montre la liste des capteurs par défaut à gauche indiqué par la flèche et une liste de capteurs dans la boite à liste trois. On observe bien un nombre 10 devant chaque nom de capteur. Ce numéro représente en fait l'identifiant du local au quel ces capteurs sont liés. Lorsqu'on regarde le nom du local (distributeur) on voit bien que son identifiant est 10.

Figure 3.16 : listes des actionneurs par défaut

Lorsque vous cliquez sur actionneurs, vous avez également une liste par défaut d'actionneurs qui apparait. Pour ajouter un actionneur, ça se passe de la même façon que les capteurs. Vous faite un clic sur le nom de l'actionneur que vous voulez ajouter puis ajouter. Vous verrez le nom du capteur apparaitre dans la dernière boite à liste (figure 3.13).

On remarque alors que la création d'un projet est vraiment simple avec une interface très facile à comprendre. Les boites à listes permettent de voir l'évolution de votre projet mais aussi vous permet de modifier supprimer ou mettre à jours les éléments de l'environnement

Dans la partie qui suit, on va voir comment paramétrer les capteurs et actionneurs avant de lancer la simulation.

3.5. Phase de configuration et simulation

Avant de lancer la simulation, il faut d'abord configurer les capteurs selon les types de données qu'ils doivent émettre et aussi connecter les actionneurs aux capteurs afin d'effectuer des actions selon les données reçues de ces capteurs.

Dans cette partie nous allons montrer comment configurer les capteurs et les actionneurs en vue de lancer une simulation. Par défaut si vous lancer la simulation sans configuration voilà ce que vous obtenez par exemple :

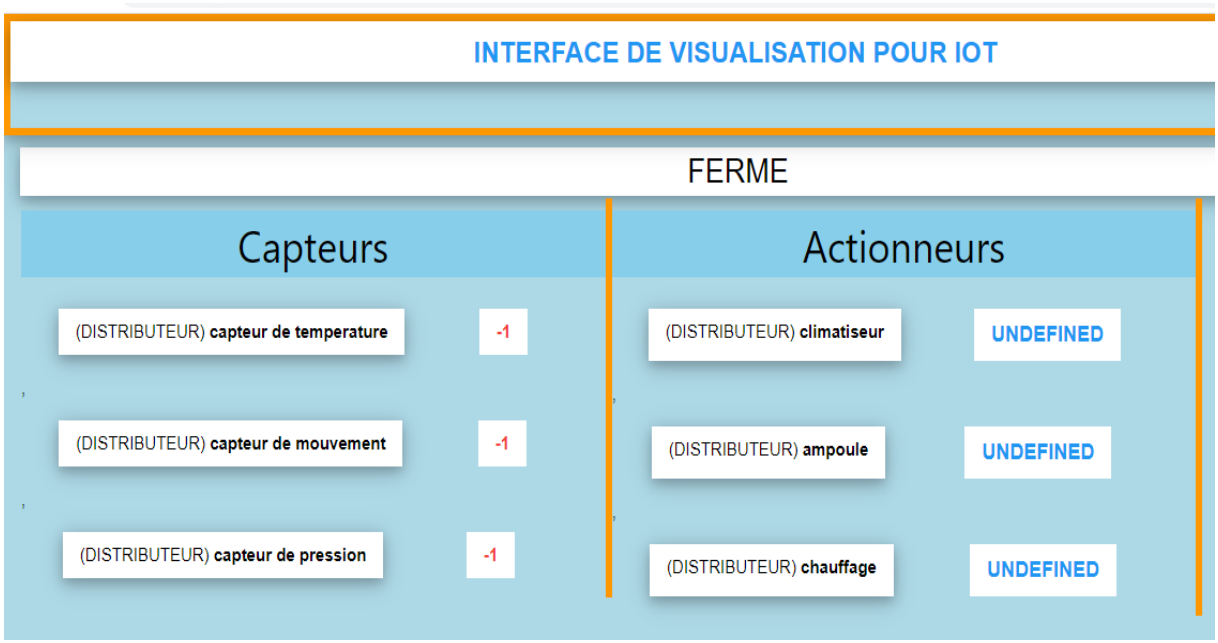

Figure 3.17 : le comportement des objets en simulation sans une configuration préalable

On peut remarquer que tous les capteurs par défaut on la valeur « -1 » et tous les actionneurs la valeur « undefine » qui signifie indéfinie, ce qui est tout à fait normal car on n'a pas défini les types de données que ses objets doivent émettre.

3.5.1. Configuration des capteurs

La configuration des capteurs consiste à définir les types de données que les capteurs doivent émettre. Un capteur de température par exemple n'a pas les mêmes types de données qu'un capteur d'humidité. Notre système offre la possibilité à l'utilisateur de définir les données des capteurs le plus proche de la réalité afin d'avoir une meilleure expérience de la simulation. Pour configurer un capteur, il faut faire un clic sur le nom du capteur (dans la boite à liste) que vous souhaitez configurer, une fenêtre s'ouvre :

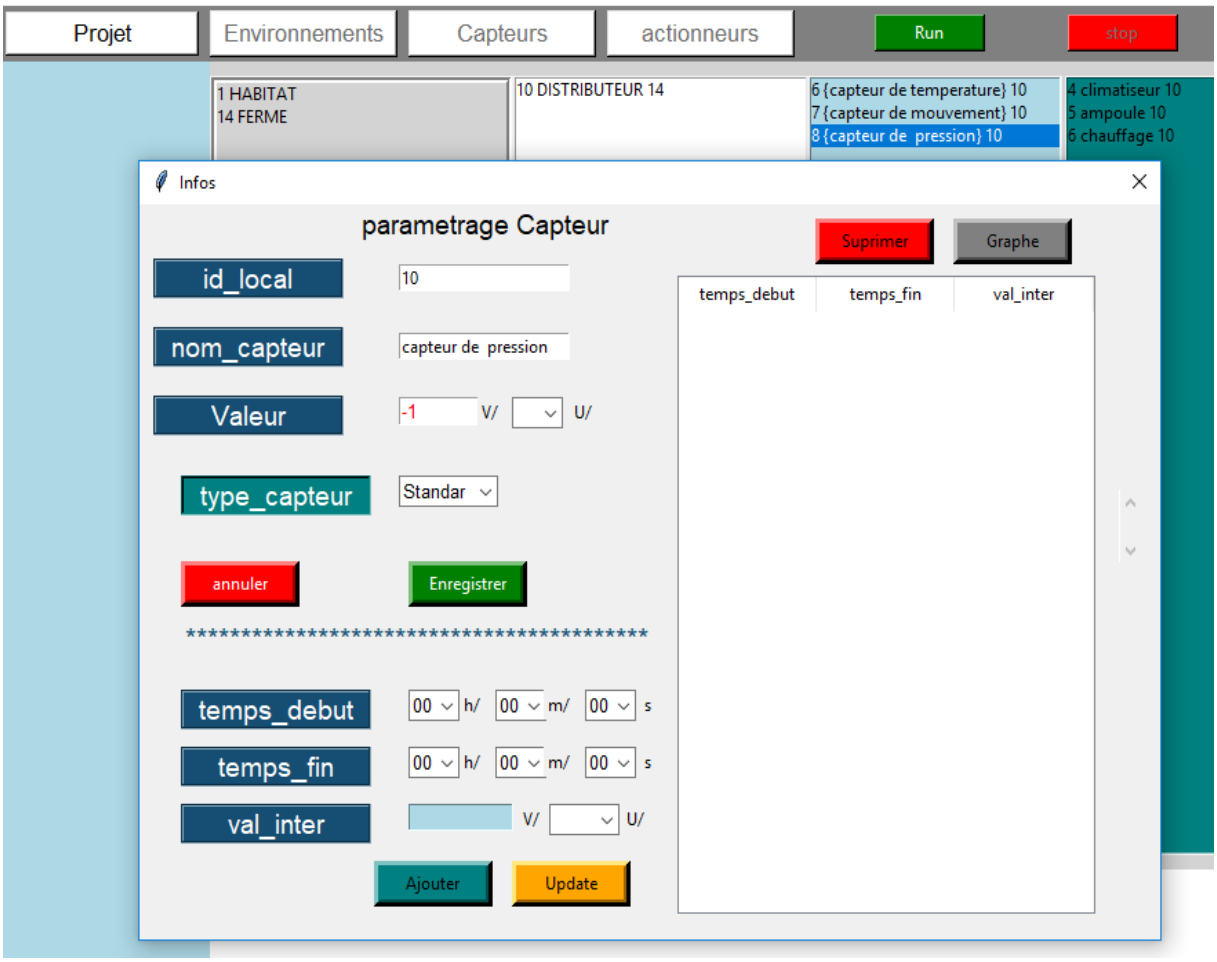

Figure 3.18 : interface de configuration d'un capteur

On a de nombreux champs de saisie dans l'interface, et chaque champ à un rôle particulier. Dans le tableau ci-dessous, on va décrire le rôle de chaque champ :

CHAMP	DESCRIPTION
Id_local	Contient l'identifiant du local auquel appartient le capteur. Il suffit de saisir l'identifiant du nouveau local auquel vous voulez attribuer le capteur avant d'enregistrer.
Nom_capteur	Contient le nom du capteur lui-même et donne la possibilité à l'utilisateur de changer le nom du capteur
Valeur	Permet de saisir la valeur par défaut du capteur. Il comporte deux champs : le premier pour la valeur numérique du capteur et le second pour l'unité.
Type_capteur	Permet de définir le type de données que va emmètre le capteur. Il comporte deux valeurs de choix : - « Standard » pour les types de données non booléen (1 ou 0) - « booléen » pour les données de type booléen
Temps_debut	Permet de définir le début d'un intervalle de temps. Par exemple dans la réaction du capteur entre 8h et 9h ; le temps début serra 8h.
Temps_fin	Permet de définir la fin d'un intervalle de temps. Par exemple dans la réaction du capteur entre 8h et 9h ; le temps fin serra 9h.
Val_inter	Permet de définir la valeur intermédiaire entre deux intervalles de temps défini dans les champs Temps_debut et Temps_fin.

Tableau 3.1 : description des champs de configuration d'un capteur

Ce sont les champs (temps_debut, temps_fin, val_inter) qui permettent de configurer le capteur afin de fonctionner dans le temps. L'objectif est que chaque capteur puisse fonctionner sous 24h.

Le champ d'affichage à droite dans l'interface de configuration du capteur permet d'afficher les données entrées pour le capteur. Il est possible de supprimer ou mettre à jour ses données.

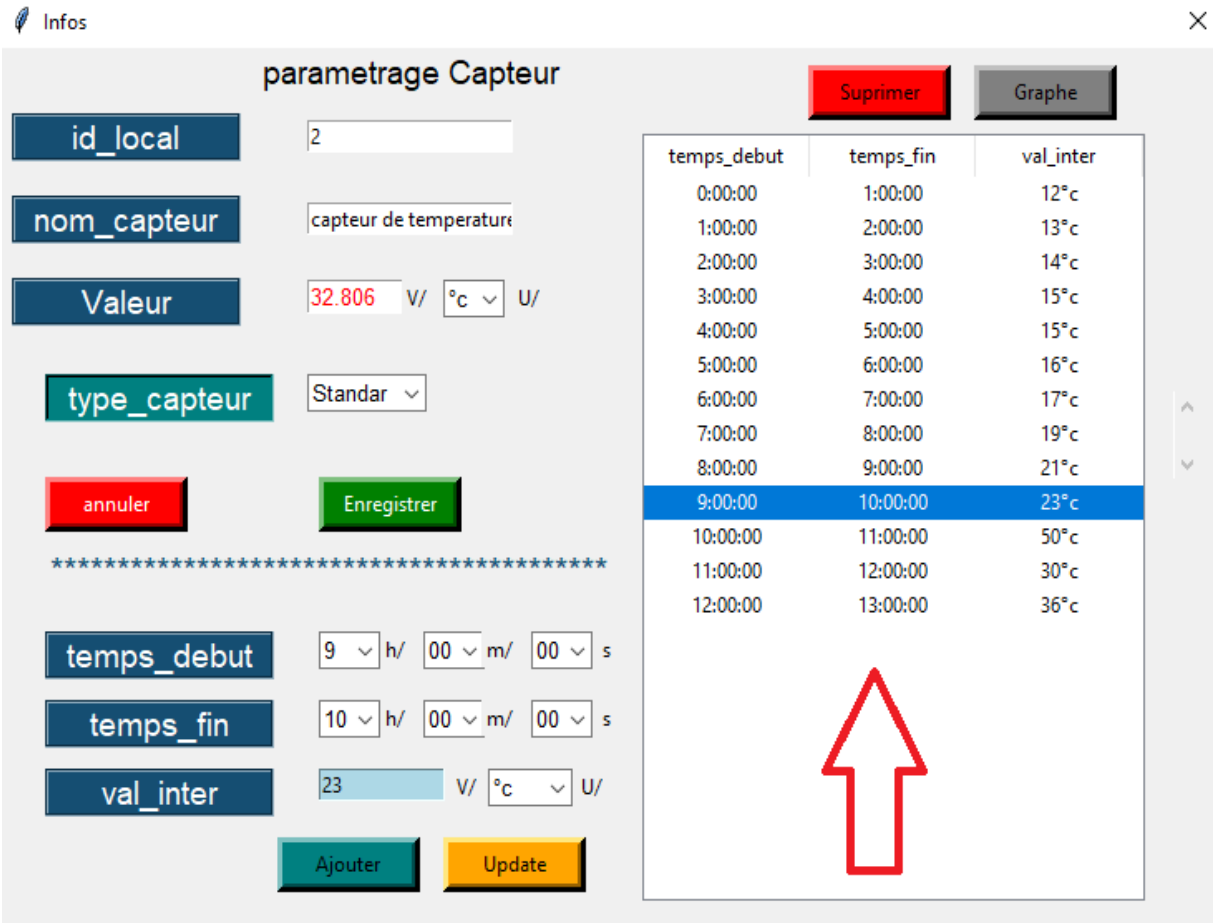

Figure 3.19 : affichage des données entrées pour un capteur

Chaque capteur a la capacité d'envoyer les données par rapport à sa situation.

Imaginons que la période de vos données est d'une heure de décalage entre deux intervalles de temps. Imaginons maintenant que votre capteur doit fonctionner sous 24h ; on aura donc 24 valeurs définis au total pour ce capteur. Mais ce qu'on doit comprendre est que le capteur n'envoie pas que 24 valeurs dans une journée ça serait insensé. Entre deux intervalles de temps le capteur peut envoyer dix, 20 voir cents valeurs. Ce qui se passe est que le capteur va générer une fonction d'interpolation

polynomial en fonction des valeurs entrées par l'utilisateur afin de prédire de manière automatique la valeur qu'elle doit envoyer et cela de manière régulière avec une probabilité d'erreur très faible. Mathématiquement, on parle de la régression polynomiale.

En outre, la fréquence d'envoi de données diffère d'un capteur à un autre. Cela dit les capteurs n'envoie pas tous leurs données en même temps ; chaque capteur à sa fréquence d'envoi de données propres à sa fonction de régression. Ce qui rend chaque capteur unique.

Pour mieux comprendre, on a intégré dans notre système de simulation, une possibilité de voir le graphe d'évolution des données que doit émettre le capteur en question. Cela permet même de voir les probabilités d'erreur que peut avoir les données envoyées par le capteur.

Pour observer le graphe du capteur, après avoir entré les données du capteur dans l'interface de configuration du capteur, il suffit de cliquer sur le bouton « graphe » dans la partie droite en haut de l'interface. Vous aurez un graphe comme ceci :

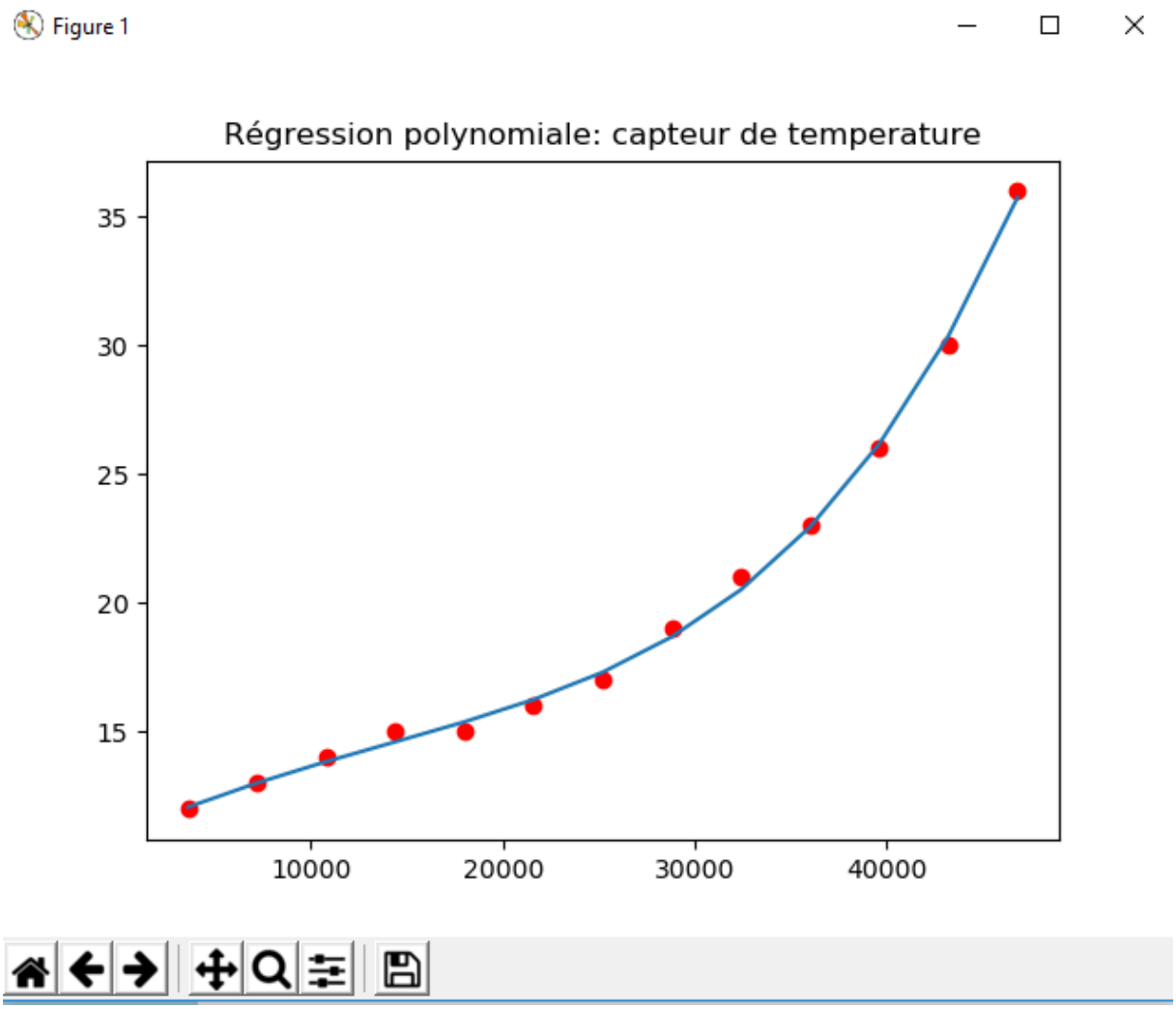

Figure 3.20 : exemple de graphe d'évolution d'un capteur

Dans ce graphe (figure 3.17), les points rouges représentent les données de référence introduit dans le capteur. Le graphe en bleu construit à partir de ces points permet au capteur de fonctionner de manière continue.

Nous pouvons observer que presque tous les points sont sur la courbe en bleue ou très proche. Ce qui implique que la probabilité que le capteur envoie une valeur erronée est très faible. Par contre si on avait des points rouges très éloignés de la courbe, on aurait conclu que le capteur risquerait d'envoyer des valeurs erronées lors de la simulation. Il est possible également d'enregistrer le graphe sur le disque local.

Dans la prochaine partie, nous allons voir comment configurer les actionneurs à réagir face aux données des capteurs auxquels ils sont liés.

3.5.2. Configuration des actionneurs

La configuration des actionneurs consiste à connecter ces actionneurs avec les capteurs afin de fonctionner au dépend de ses capteurs. Pour configurer un capteur, il suffit de cliquer sur le nom du capteur dans la boite à liste, vous aurez une interface en toplevel comme le cas vu avec les capteurs. L'interface de configuration comporte également des champs de saisi ayant tous un rôle particulier.

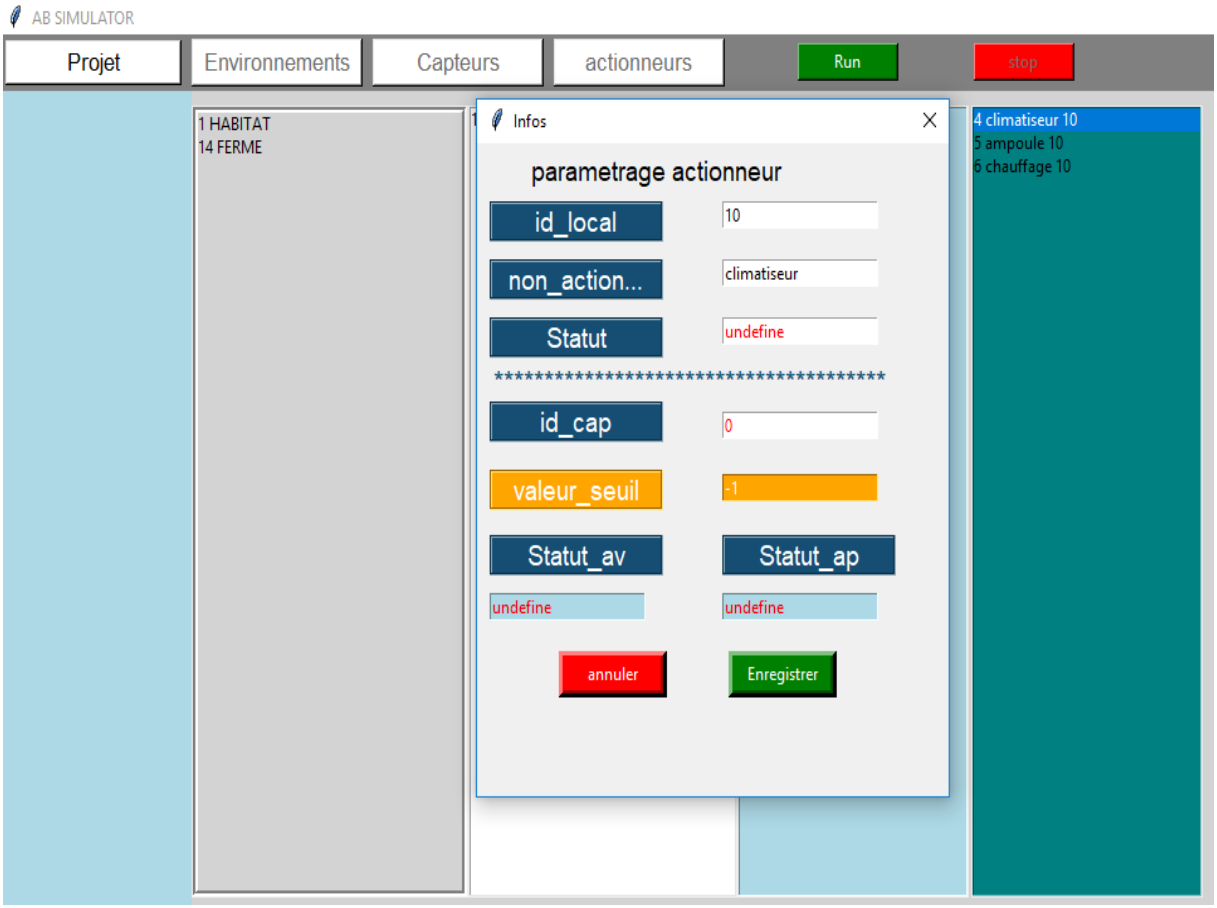

Figure 3.21 : interface de configuration d'un actionneur

Le tableau qui suit va décrire les champs de configuration d'un actionneur.

CHAMP	DESCRIPTIONS
Id_local	Contient l'identifiant du local auquel appartient l'actionneur. Pour changer le local de l'actionneur, il suffit de saisir l'identifiant du nouveau local auquel vous voulez l'attribuer.
Nom_actionneur	Contient le nom de l'actionneur lui-même et donne la possibilité à l'utilisateur de le personnaliser
Statut	Permet de définir le statut par défaut de l'actionneur. Chaque actionneur à deux statuts : - TRUE = en action - FALSE = non action
Id_cap	Permet de définir l'identifiant du capteur auquel sera lié l'actionneur. Sans ce champ, l'actionneur ne pour pas accéder aux données d'un capteur.
Valeur_seuil	Permet d'indiquer la valeur seuil entre les deux états de l'actionneur. C'est la valeur de référence
Statut_av	Permet de définir l'état de l'actionneur lorsque la valeur reçue du capteur est inférieure ou égale à la valeur seuil.
Statut_ap	Permet de définir l'état de l'actionneur lorsque la valeur reçue du capteur est supérieure à la valeur seuil.

Tableau 3.2 *:* description des champs de configuration d'un actionneur

La configuration de l'actionneur est très simple, il suffit d'identifier le capteur auquel il sera lié et aussi se rassurer que l'identifiant du capteur soit valide. Dans la prochaine partie nous allons enfin passer à la simulation pour observer le fruit de notre travail.

3.5.3. Lancer la simulation

L'étape de la simulation est la plus attendue pour observer votre projet IOT. Pour lancer la simulation, il y'a un bouton « Run » dans l'interface principale du système qui permet de lancer la simulation.

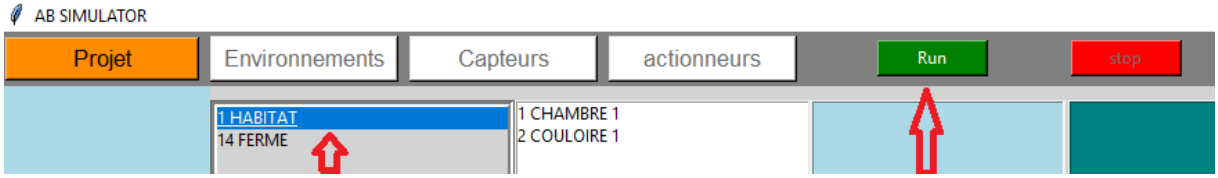

Figure 3.22 : étape de lancement de la simulation

Avant de lancer la simulation, il faut d'abord sélectionner le nom de l'environnement que vous souhaitez lancer en simulation comme indiqué dans la **figure 3.22**. En outre, avant de lancer la simulation, il faut aussi préparer l'interface de visualisation qui est une page web. Pour cela, vous devez ouvrir l'interface de visualisation dans un navigateur web de votre choix. Une fois ouvert, vous pouvez en fin cliquer sur le bouton « Run » pour lancer la simulation.

Lorsqu'on lance la simulation, une requête SQL est envoyée au serveur pour récupérer toutes les informations relatives à l'environnement dans la base de données. Ces informations concernent les locaux appartenant à l'environnement, les capteurs de chaque local ainsi que les actionneurs de chaque local. Ces informations sont ensuite enregistrées dans un fichier json (JavaScript Object Notation) en vue d'une meilleure organisation pour être ensuite envoyées vers le broker MQTT. Json est un langage léger d'échange de données textuelles. Prenons l'exemple des données d'un capteur dans un fichier json dans l'image suivante :

```
[
    {
        "id_table": 17,
        "temps_debut": "0:00:00",
        "temps_fin": "1:00:00",
        "valeur": 45,
        "unité": "°c"
    },
    {
        "id_table": 18,
        "temps_debut": "1:00:00",
        "temps_fin": "2:00:00",
        "valeur": 44,
        "unité": "°c"
    },
    {
        "id_table": 19,
        "temps_debut": "2:00:00",
        "temps_fin": "3:00:00",
        "valeur": 43,
        "unité": "°c"
    },
    {
        "id_table": 20,
        "temps_debut": "3:00:00",
```


Figure 3.23 : données d'un capteur sous un format json

On remarque qu'une structure très bien organisée et facile de lecture. En somme, les données sont toujours envoyées au serveur sous forme json afin d'avoir une meilleure réception des données sur l'interface de visualisation.

Une fois la simulation lancée, on obtient ceci (figure 3.20) sur l'interface secondaire (de visualisation :

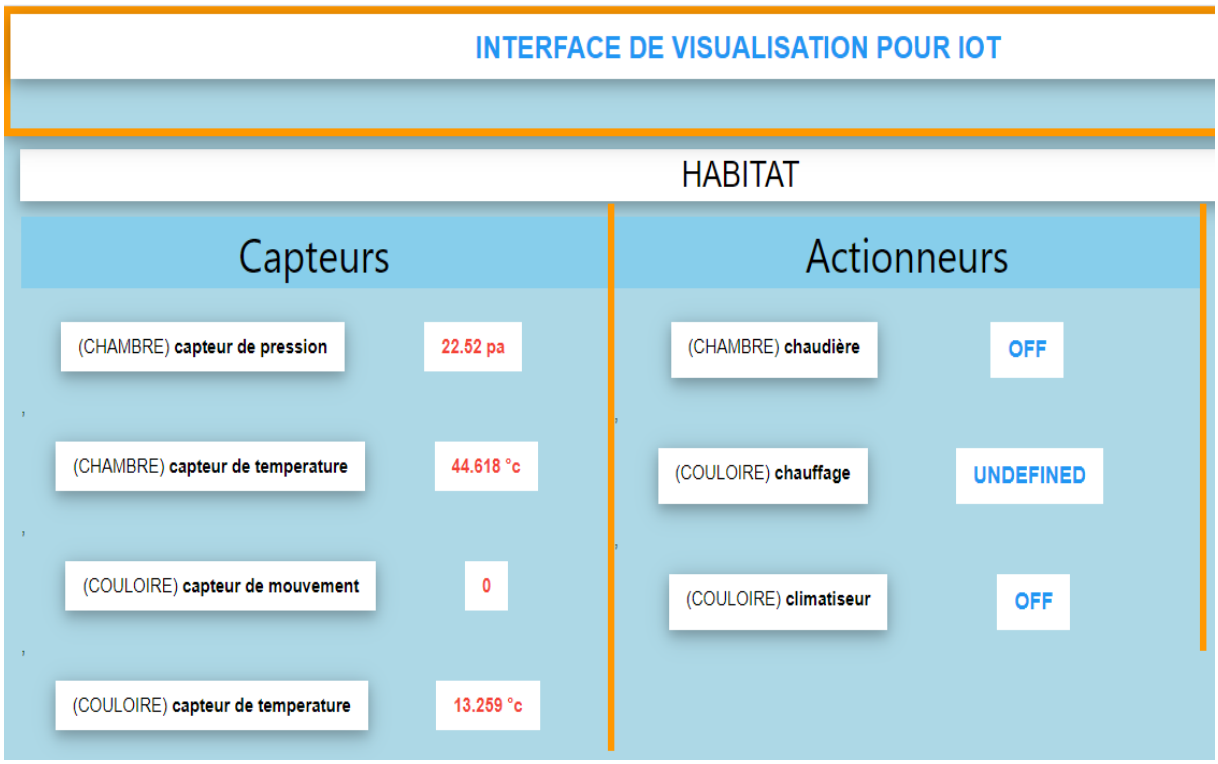

Figure 3.24 : résultat de la simulation

Nous pouvons observer sur l'interface, l'état des capteurs et les actionneurs de l'environnement. Dans l'affichage, il y'a le nom de l'environnement en cours de simulation, une liste de capteurs à gauche et une liste d'actionneurs à droite. On peut observer que chaque capteur ou actionneur est précédé par le nom du local auquel il appartient. Ce qui facilite la visibilité de la simulation et une meilleure observation. On remarque également que les valeurs des capteurs changent dynamiquement. Ce qui montre toute la logique dans le fonctionnement de ces derniers. Les capteurs avec une valeur « -1 » par défaut et les actionneurs avec une valeur « undefine » constituent ceux qui n'ont pas été configurés. On peut observer aussi que les états des actionneurs sont synchronisés avec les données des capteurs comme indiqué dans la configuration de ces derniers.

Pour arrêter la simulation, il suffit de cliquer sur le bouton « stop » en rouge dans l'interface principale.

Dans la prochaine étape, nous allons voir comment sécuriser notre système de simulation.

3.5.4. Sécurité

La sécurité est un point essentiel qui permet de protéger le système de simulation contre des manœuvres indésirables d'intrus au système. En effet, le protocole MQTT étant basé sur l'architecture publier/souscrire sur des topics, n'importe qui pourrait envoyer des données sur votre serveur MQTT s'il connait le nom du topic. Cependant, lorsque des données inconnues sont envoyées sur votre topic, cela pourrait compromettre les résultats de votre simulation.

Pour pallier à ce problème on a pensé à intégrer une solution de sécurité dans le système. Cette solution consiste à demander les identifiants de l'utilisateur pour pouvoir accéder au serveur MQTT. Ainsi un utilisateur qui n'est pas identifié ne pourra jamais publier sur le broker et aussi même s'il s'inscrit à un topic il ne recevra aucune donnée tant qu'il ne s'est pas identifié.

Pour implémenter cette solution, nous avons utilisé le système d'authentification de notre serveur MQTT (Mosquitto) appelé « Mosquitto_passwd ». Il permet à l'utilisateur de bloquer l'accès au serveur par un ou plusieurs mots de passe.

Les étapes pour créer un mot de passe d'authentification :

- **Etape 1 :**

  Commencer par créer un fichier (un fichier texte) dans lequel vous allez enregistrer vos mots de passes. Ce fichier doit être dans le dossier d'installation de Mosquitto.

- **Etape 2 :**

  Ouvrir l'invité de commande de votre ordinateur (cmd sous Windows) puis naviguez dans le fichier d'installation de Mosquitto et taper la commande suivante :

  « **mosquitto_passwd -b nomDuFichier nomUtilisateur motDePasse** », puis ok et vous constaterez dans le fichier texte que vous avez créé, un nom d'utilisateur et un mot de passe crypté. Exemple :

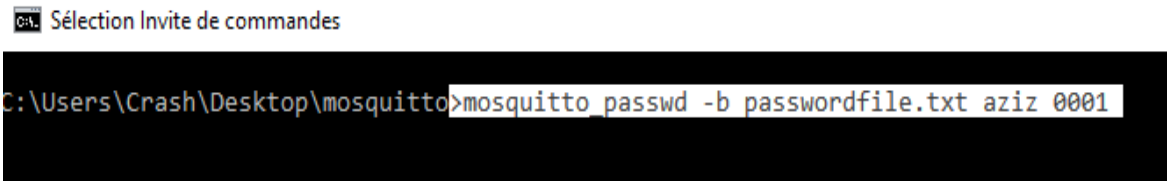

Figure 3.25 : commande de création d'un mot de passe sur Mosquitto

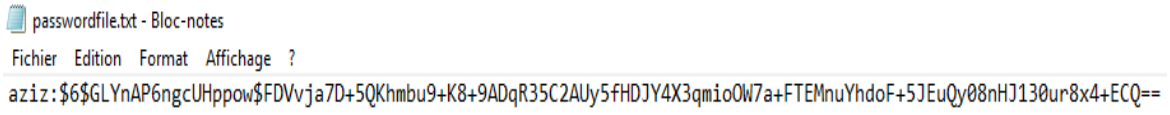

Figure 3.26 : mot de passe crypté sous Mosquitto

- **Etape3 :**

Vous devez en fin configurer Mosquitto pour bloquer les accès anonymes car par défaut, Mosquitto autorise tous les accès. Pour se faire, il faut ouvrir le fichier Mosquitto.conf toujours dans le dossier d'installation de Mosquitto et ajouter les lignes suivantes à la fin :

Figure 3.27 : bloquer les accès anonymes sur Mosquitto

La ligne une force « **allow_anonymous** à false », ce qui permet d'interdire les accès anonymes. Et la ligne deux indique le chemin absolu vers le fichier qui contient les mots de passe afin de n'autoriser que les identifiants qui y sont.

Pour supprimer un mot de passe, il suffit de taper la commande **« mosquitto_passwd -D nomDuFichier nomUtilisateur ».**

Une fois toutes ses étapes effectuées, le système de simulation est en fin prêt à effectuer une connexion sécurisée pour une meilleure expérience de simulation.

Pour une question de rigueur, notre système va toujours vous demander un mot de passe lorsque vous lancer la simulation. Si votre serveur n'exige pas de mot de

passe, vous annulez les demandes de saisie et tout ira bien. Ci-dessous un exemple de simulation sécurisé :

Lorsque vous lancez la simulation, le système demande votre nom d'utilisateur :

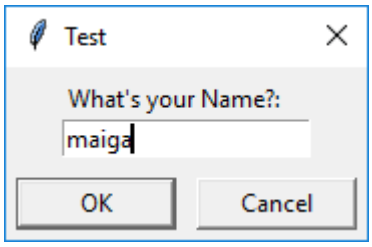

Figure 3.28 : champ de saisie nom d'utilisateur, identification Mosquitto

Ensuite votre mot de passe :

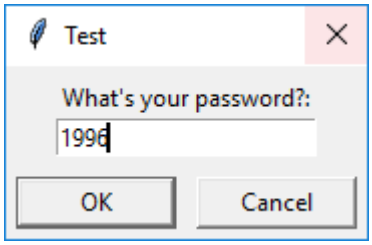

Figure 3.29 : champ de saisie mot de passe, identification Mosquitto

Si le nom d'utilisateur et le mot de passe sont correctes, la simulation se lance et les résultats serons affiché dans l'interface de visualisons.

Si par contre les identifiants sont erronés, vous aurez un message d'erreur :

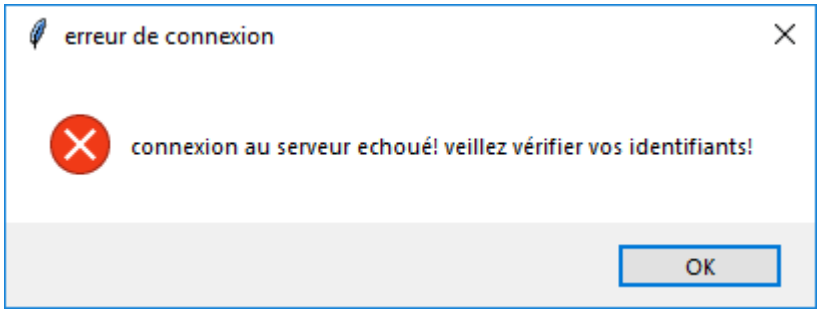

Figure 3.30 : message d'erreur en cas d'authentification incorrecte

Pour plus de sécurité, Mosquitto offre à ses utilisateurs une possibilité de connexion SSL grâce au logiciel openssl de cryptage de données. Ainsi tous les messages seront cryptés lors de la simulation, et l'on ne pourra pas les décrypter même si on arrive à les intercepter.

3.6. Conclusion

Ce chapitre a été consacré à la prise en main du système de simulation en vue de permettre aux futurs utilisateurs de créer facilement leurs projets IOT et avoir une meilleure expérience dans la simulation de leur environnement.

En effet, après avoir montré la gestion des communications via les protocoles MQTT et websocket avec le serveur Mosquitto, nous avons montré comment concevoir un projet de A à Z allant du choix de l'environnement, des locaux, des capteurs aux actionneurs.

Ensuite, nous avons montré comment paramétrer les capteurs et les actionneurs avant une quelconque simulation. Nous avons souligné également que les données étaient en format json lors de l'envoie au serveur afin de permettre une meilleure réception de ces données.

Nous avons aussi montré comment sécuriser le serveur Mosquitto afin d'éviter toute intrusion de personnes non autorisées à publier sur ce serveur. Cela a pour but de protéger la simulation et éviter d'avoir des surprises si seulement un particulier arrivait à envoyer des données indésirables sur notre système.

Conclusion générale

Les recherches menées durant la mise en œuvre de ce projet, nous ont permis de comprendre dans un premier temps comment l'internet des objets est sur le point de révolutionner notre monde.

Le premier chapitre a permis de ressortir les fondamentaux de l'IoT que sont les objets, les réseaux et leurs domaines d'application. Nous pouvons retenir que théoriquement parlant, l'IoT à des beaux jours devant lui. En outre le monde s'active pour faciliter l'insertion de cette technologie dans notre vie quotidienne.

Le chapitre deux a fait l'objet de recherche sur la faisabilité de notre système de simulation. Ainsi nous avons comparé les services que notre système offre par rapport aux technologies de simulation déjà existantes. Il ressort en effet que notre système offres des fonctionnalités sur l'autonomie des capteurs, des actionneurs ainsi que la sécurité que la plupart des systèmes ne prennent pas en charge ou négligent. Ensuite on a conçu les architectures des interfaces de notre système avant de passer à leur programmation grâce aux langages Python, HTML, CSS et JAVASCRIPT. Dans le troisième, nous avons présenté une étude pratique de notre système. Nous avons décrit en effet, la gestion de la communication entre nos différentes interfaces, nous avons montré comment créer un projet IOT de A à Z. Nous avons également montré comment configurer les capteurs ainsi que les actionneurs et enfin comment sécuriser le serveur contre les intrusions.

Dans l'ensemble, les résultats pratiques sont satisfaisants comme nous l'avons prévue dans l'étude théorique. Les capteurs remplissent bien leurs fonctions, les actionneurs également.

En dépits des fonctions que remplit le système, des améliorations futures peuvent être apportés au système. Ainsi comme perspectives nous prévoyons :

- Introduire un algorithme permettant de simuler le déplacement d'une voiture via des données GSP en temps réel.

- Améliorer la présentation graphique des capteurs et actionneurs en y assimilant des images adaptées pour chaque cas afin de faciliter les créations de projets.
- Créer des environnements types intégrés au système comme des parkings, des garages et même des torchons de routes.
- Intégrer des fonctions de sorties afin de réagir avec des équipements électroniques physiques comme des cartes Arduino, des capteurs réels, des actionneurs, etc.

Bibliographie

1. Mike Loukides etIon Brunor: 'What is the internet of things', O'Reilly Media, 2015.

2. 'Introduction à l'IOT', Cisco Systems, 1992–2008.

3. Sabina Jeschke, Christian Brecher, Houbing Song, Danda B. Rawat: 'Industrial Internet of Things', Springer International Publishing Switzerland, 2017

4. 'Le mobile au chevet de l'Afrique', Orange Healthcare, mai 2015.

5. Yassine Haddab :'Introduction à l'internet des Objets',2016

6. D. Jung, MATSNL :' une plate-forme de nœud de capteur sans fil MATLAB ™, Package de prédiction et de simulation à vie', Version 1.0.0, Février 14, 2007

7. Nasser Alshammari, Talal Alshammari, Mohamed Sedky, Justin Champion, Carolin Bauer:'OpenSHS Open Smart Home Simulator', sensors, Ioannis Chatzigiannakis and Georgios Mylonas,2017.

8. G. Kalpana1 1 Asst. Professor, CBIT, Hyderabad :' La technologie, les outils et les simulateurs d'Internet de choses', Revue internationale de recherche en sciences appliquées et technologie de l'ingénierie (IJRASET), Volume 6 Parution III pp2432-2438, 2018.

9. Brambilla, G., Picone, M., Cirani, S., Amoretti, M., & Zanichelli, F :' Une plate-forme de simulation pour les scénarios de l'Internet des objets à grande échelle dans les environnements urbains'. Actes de la première conférence internationale sur l'IoT dans l'espace urbain, 2014, 50-55

10. Bouchard, K., Ajroud, A ., Bouchard, B .,Bouzouane, A :'SIMACT: un simulateur de maison intelligente 3D Open Source pour la reconnaissance d'activité' , Dans le cadre des progrès de l'informatique et des technologies de l'information: Conférences AST / UCMA / ISA / ACN 2010, Miyazaki, Japon, 23-25 juin 2010; Kim, T.H., Adeli, H., Eds .; Springer: Berlin / Heidelberg, Allemagne, 2010; pp. 524-533.

11. Shnayder, M. Hempstead, B. Chen, G.W. Allen et M.Welsh :'Simulation de la consommation électrique d'un réseau de capteurs à grande échelle Applications', Division de l'ingénierie et des sciences appliquées, Université de Harvard, 2003.

12. Lahmar, K., Cheour, R. et Abid, M :'Réseaux de capteurs sans fil : tendances, consommation d'énergie et simulateurs', Sixième Symposium sur la modélisation de l'Asie ,2012.

13. Charles Bell: 'MySQL for the Internet of Things',2016

14. Gastón C. Hillar: ' Internet of Things with Python ', Packt Publishing, 2016.

15. IOTdesign.com

16. Christophe lagane, 8 juillet 2015 : https://www.silicon.fr/simulation-service-de-linternet-objets-121276.html

17. BERRONE Pascual, RICARD Joan Enric, DUCH Ana, CARRASCO Carlos, 2019, « IESE Cities in motion index 2019 », *IESE, ST-509-E, 9*

18. OCDE/FAO (2016), « L'agriculture en Afrique subsaharienne : Perspectives et enjeux de la décennie à venir », dans *Perspectives agricoles de l'OCDE et de la FAO 2016-2025*, Éditions OCDE, Paris

19. Steven, 5 mars 2020 : https://www.linstant-interview.com/174454/ido-agricole-marche-apercu-complet-de-la-taille-et-du-partage-et-potentiel-de-croissance-future-2027/

20. tutorial écrit par Guy Geek sur : http://www.lafabriquediy.com/tutoriel/liste-des-capteurs-229/

21. les systèmes automatisés sur : http://www.technobreizh.eu/4eme/16-seq1-4eme/10-les-systemes-automatises.