



REPUBLIQUE DU SENEGAL
Un Peuple - Un But - Une Foi

Ministère de l'Enseignement Supérieur et de la Recherche
Direction Générale de l'Enseignement Supérieur
Direction de l'Enseignement Supérieur Privé



Ecole Supérieure de Technologie et de Management de Dakar
Département des Sciences et Technologies

Mémoire de fin de cycle

En Vue de L'Obtention de la Licence en Téléinformatique
Option Télécommunications et Réseaux

Thème

**ETUDE ET MISE EN PLACE D'UNE PLATEFORME
INTEGRANT DU CLOUD COMPUTING ET IOT**

Présenté par :

Arona Seye

Encadrant :

Dr Keba Gueye – enseignant à ESTM

Année universitaire : 2021/2022

Remerciements

Au terme de ce travail, je tiens à exprimer ma profonde gratitude et mes sincères remerciements à :

- mon encadreur monsieur le Professeur Dr Kéba Gueye pour m'avoir proposé ce sujet et m'avoir dirigé, conseillé et encouragé tout au long de la réalisation de ce travail et surtout pour sa patience et sa compréhension afin de donner le fruit d'un travail long et difficile. Veuillez croire à l'expression de ma grande admiration et mon profond respect ;
- tous mes camarades de classe avec qui j'ai passé de très bons moments d'échange et d'entraide ;
- tout le cadre professoral et administratif de l'ESTM.

Dédicaces

Je dédie ce modeste travail à :

- mes parents, que Dieu me les gardes ;
- mon maître coranique, que Dieu lui accorde longue vie et bonne santé ;
- mes chers et adorable frères et sœurs ;
- tous mes amis.

Résumé

Avec l'évolution croissante d'internet et les besoins énormes des utilisateurs particuliers les entreprises de nouvelles technologies on fait leur apparitions afin de satisfaire ces nouvelles contraintes, le stockage et le traitement en ligne et distant aussi appelé **CLOUD** domine de plus en plus le marché et devient essentiel dans le monde de **l'internet des objets**.

L'avènement du cloud computing a permis aux solutions IoT d'adopter de nouvelles fonctionnalités telles que : l'interactivité, qui se manifeste par une prise de décision indépendante, une intelligence et une interaction avec l'environnement via des capteurs et dispositifs microélectroniques intégrées.

Dans ce mémoire, nous avons étudié les concepts du cloud computing et de l'IoT, et proposer ensuite une solution IoT capable de s'exécuter sur n'importe quel système d'exploitation grâce à la plateforme de conteneurisation docker. D'où le concept de plateforme intégrant du cloud computing et IoT.

Abstract

With the growing evolution of the internet and the huge needs of individual users, new technologies have emerged to meet these new constraints, online and remote storage and processing also called **CLOUD** is increasingly dominating the market and becoming essential in the world of the **Internet of Things**.

The advent of cloud computing has allowed IoT solutions to adopt new functionalities such as: interactivity, which manifests itself in independent decision making, intelligence and interaction with the environment via embedded sensors and microelectronic devices.

In this memoir, we studied the concepts of cloud computing and IoT, and then propose an IoT solution capable of running on any operating system through the docker containerization platform. Hence the concept of a platform integrating cloud computing and IoT.

Table des matières

Remerciements	i
Dédicaces	ii
Résumé	iii
Abstract	iv
Table des matières	v
Figures	viii

1 - Introduction générale.....	1
1.1. Contexte	1
1.2. Problématique.....	1
1.3. Objectif	1
1.4. Présentation de l'entreprise	2
 2 - Cloud Computing	 3
2.1. Introduction	3
2.2. Définition du cloud computing	3
2.3. Composants du cloud	4
2.4. Les modèles de services cloud	4
2.3.1 Software as a service (SaaS)	5
2.3.2 Platform as a service (Paas)	5
2.3.3 Infrastructure as a service (IaaS)	5
2.5. Les types de cloud	6
2.4.1 Cloud public	6
2.4.2 Cloud privé	7
2.4.3 Cloud hybride	7
2.4.4 Cloud communautaire	7
2.6. Principaux fournisseurs cloud	8
2.5.1 Microsoft Azure	8
2.5.2 Google Cloud Platform	9
2.5.3 Amazone Web service	9
2.7. Caractéristiques du cloud computing	10
2.8. Vue d'ensemble de la virtualisation	11
2.8.1 Avantages de la virtualisation	11
2.8.2 Types de virtualisation	12

2.8.3	Types de ressources virtualisées	13
2.9.	Avantages et inconvénients du cloud computing	14
2.10.	Conclusion	15
3	- Internet des Objets	17
3.1.	Introduction	17
3.2.	Architecture de l'internet des objets	17
3.2.1	Couche perception	18
3.2.2	Couche réseau	18
3.2.3	Couche application	18
3.3.	Objet connecté	19
3.3.1	Objets portés	19
3.3.2	Objets indépendants	19
3.4.	Composant d'un système IoT	20
3.4.1	Les capteurs	20
3.4.2	Les actionneurs	21
3.4.3	Les sources d'énergies	21
3.4.4	La connectivité	22
3.5.	Réseaux IoT	22
3.5.1	Les réseaux personnels	22
3.5.2	Les réseaux bâtiments	23
3.5.3	Les réseaux LPWAN (Low Power Wide Area Network)	23
3.6.	Caractéristique fondamentale de l'internet des objets	25
3.7.	Domaine d'application de l'internet des objets	25
3.7.1	Maison intelligente	25
3.7.2	Santé	25
3.7.3	Industrie	26
3.8.	Protocoles de l'internet des objets	26
3.8.1	HyperText Transfert Protocol (http)	26
3.8.2	Websocket	26
3.8.3	MQ Telemetry Transport (MQTT)	27
3.9.	Systèmes embarqués	27
3.9.1	Définition	27
3.9.2	Caractéristique des systèmes embarqués	27
3.10.	Conclusion	28
4	- Etude de plateforme IoT (Microsoft IoT Central et Amazon AWS IoT)	29
4.1.	Introduction	29
4.2.	Définition d'une plateforme IoT	29

4.3.	Types de plateforme IoT : open source, propriétaires	29
4.4.	Microsoft IoT Central	30
4.4.1	Présentation de Microsoft IoT Central	30
4.4.2	Avantages	30
4.4.3	Fonctionnalités	30
4.4.4	Architecture	31
4.4.5	Intégrations et API	33
4.5.	Amazon AWS IoT	33
4.5.1	Présentation Amazon AWS IoT	33
4.5.2	Avantages	33
4.5.3	Fonctionnalités	33
4.5.4	Architecture	34
4.5.5	Intégrations et API	35
4.6.	Conclusion	36
5	- Système d'arrosage automatique avec l'API OpenWeatherMap	37
5.1.	Introduction	37
5.2.	Architecture générale du système	37
5.3.	Outils et plateformes utilisés	38
5.3.1	Le microcontrôleur esp32	38
5.3.2	L'outil de programmation Node Red	38
5.3.2.1	Installation	39
5.3.2.2	Ajout d'une catégorie de nodes à la palette	39
5.3.2	L'API OpenWeatherMap	40
5.3.3	L'IDE Visual Studio Code et l'extension PlatformIO	41
5.3.4	Machine virtuelle Ubuntu 20.04	41
5.4.	Configurations tests et résultats	42
5.4.1	Exécution du programme sous docker	52
5.4.1.1	Présentation et installation docker	52
5.4.1.2	Mise en place du conteneur	54
5.4.1.3	Push du conteneur vers le référentiel docker	56
5.5.	Conclusion	58
	Conclusion générale et perspectives	59
	Références	60

Figures

2.1.	Piles cloud computing	4
2.2.	Modèles de services cloud	6
2.3.	Types de cloud	8
2.4.	Fournisseurs Cloud	10
2.5.	Provisionnement machine virtuelle	12
2.6.	Types de virtualisations	13
3.1.	Couche de l'architecture IoT	18
3.2.	Composant d'un système IoT	20
3.3.	Réseaux IoT	22
3.4.	Système embarqué	28
4.1.	Fonctionnalité Microsoft IoT Central	31
4.2.	Architecture Microsoft IoT Central	32
4.3.	Fonctionnalité Amazon AWS IoT	34
4.4.	Architecture Amazon AWS IoT	35
5.1	Architecture générale du système	37
5.2	ESP32	38
5.3	Interface node-red	39
5.4	Ajout de la catégorie Dashboard à la palette	40
5.5	Clé api OpenWeatherMap	40
5.6	Requête http GET	40
5.7	PlatformIO	41
5.8	Versions Ubuntu	42
5.9	Installation PlatformIO IDE	43
5.10	Fichier de configuration PlatformIO	43
5.11	Flow Node-red	48
5.12	Dashboard Node-red	49
5.13	Installation du nœud MySQL	50
5.14	Configuration de la base de données	50
5.15	Configuration du nœud MySQL	51
5.16	Enregistrement des données dans MySQL	51
5.17	Dashboard final	52
5.18	Architecture de base de docker	53
5.19	Flow Node-red docker	55
5.20	Dashboard Node-red docker	56
5.21	Docker login	56
5.22	Docker push	57
5.23	Docker hub	57

Chapitre 1

Introduction générale

Etude et mise en place d'une plateforme intégrant Cloud Computing et IOT

1.1. Contexte

L'Internet des objets (IoT) est un écosystème d'objets physiques connectés accessibles via Internet. Ainsi, en termes simples, IOT signifie tout ce qui peut être connecté à Internet et peut être contrôlé, surveillé à l'aide d'Internet à partir de nos appareils intelligents ou PC.

Ces objets doivent s'intégrer dans un système plus global qu'est le monde digital et s'y adapter. L'IoT comprend une grande diversité de dispositifs intégrant capteurs et actuateurs. Via les capteurs, l'IoT observe, mesure l'état du monde réel, ce qui est essentiel pour la prise de décision. Via les actuateurs, l'IoT agit sur le monde réel. Elle doit s'intégrer dans un système plus global qu'est l'écosystème digital.

Actuellement, nous ne pouvons pas intégrer un objet connecté dans une société d'objets (écosystème digital) de manière flexible, car il nécessite une configuration difficile, et la présence d'un groupe d'objets connectés les uns aux autres ne signifie pas qu'ils constituent une société qui travaille pour atteindre des objectifs et des intérêts individuels et collectifs commun et cela est dû à l'interopérabilité et les différentes structures et protocoles qui composent les objets.

1.2. Problématique

L'interaction des éléments de cet écosystème digital entre eux et la transformation des états d'une grandeur physique observée en une grandeur utilisable pour le stockage et le traitement numériques, produit une énorme quantité de données. Les ressources et les capacités de stockage et de traitement limitées des objets ne parviennent pas à gérer cette quantité de données, ce besoin en matière de stockage et de traitement de grande masse de données est pris en charge par l'infrastructure de calcul via le cloud qui de manière simple permet de sauvegarder des données sur internet.

1.3. Objectifs

Actuellement, le développement du cloud et l'automatisation des différentes tâches par l'IoT constituent un atout majeur pour la société, raison pour laquelle nous proposons une solution de mise en place d'une plateforme permettant de mettre les données IoT au niveau du cloud afin de les traiter et d'en prendre des décisions intelligentes.

Pour atteindre cet objectif, les étapes du projet sont les suivantes :

- Faire la revue des technologies comprises dans le concept de l'IoT et du cloud.

- Etudier différentes technologies permettant de traiter des données IoT au niveau du cloud afin de proposer un système IoT capable de s'exécuter dans le cloud.

1.4. Présentation de l'entreprise

ESTM (Ecole Supérieure de Technologie et de Management) de Dakar, est une école privée d'enseignement supérieur, universitaire et professionnel. Elle a été créée en 2001 par les professionnels des secteurs des technologies de l'information, de la communication et de la gestion.

Les enseignements dispensés s'inspirent des normes exigées par le CAMES (Comité Africain et Malagas pour l'enseignement Supérieure) et sont donc superposable à ceux dispensés dans des meilleures écoles tant sur le continent africain que sur le reste du monde.

L'école compte toujours rester à la pointe de la technologie dans un environnement qui se veut compétitif. Ses principaux atouts résident dans la composition conseil pédagogiques, la qualité des enseignants et dans la rigueur. Ce conseil pédagogique consultatif élabore programmes qui sont constamment mise à jour, dans l'optique de les adapter à la réalité toujours suivant les besoins de l'entreprise. C'est ce qui lui a d'ailleurs donné la pleine reconnaissance de ses diplômes par le CAMES.

Cloud computing

2.1. Introduction

Le cloud computing peut être considéré comme une réponse évoluée aux besoins informatiques d'aujourd'hui. De nombreux aspects de la vie quotidienne ont été transformés par l'omniprésence de logiciels fonctionnant sur des réseaux cloud. En tirant parti du cloud computing, les startups et les entreprises sont en mesure d'optimiser les coûts et d'augmenter leurs offres sans acheter et gérer tout le matériel et les logiciels. Les développeurs indépendants sont habilités à lancer des applications et des services en ligne disponibles dans le monde entier. Les chercheurs peuvent partager et analyser des données à des échelles autrefois réservées aux projets fortement financés. Et les internautes peuvent accéder rapidement aux logiciels et au stockage pour créer, partager et stocker des médias numériques en quantités qui vont bien au-delà de la capacité de calcul de leurs appareils personnels.

Dans cette partie, nous fournirons un aperçu général du cloud computing, ses modèles de livraison, ses offres et ses risques.

2.2. Définition du cloud computing

Le cloud computing est la fourniture de ressources informatiques en tant que service, ce qui signifie que les ressources sont détenues et gérées par le fournisseur de cloud plutôt que par l'utilisateur final. Ces ressources peuvent inclure des applications logicielles basées sur un navigateur (telles que Tik Tok ou Netflix), le stockage de données tiers pour les photos et autres supports numériques (tels que iCloud ou Dropbox), ou des serveurs tiers utilisés pour prendre en charge l'infrastructure informatique d'une entreprise, une recherche ou un projet personnel.

Traditionnellement, une organisation qui doit déployer une solution informatique en particulier doit se procurer, configurer et gérer l'infrastructure et l'application. Certaines organisations peuvent décider de développer leurs propres logiciels. Dans ce cas, l'organisation « est propriétaire » de la solution, ce qui lui permet d'exercer un contrôle total, notamment pour en sécuriser l'accès et la personnaliser. Toutefois, être propriétaire d'une solution présente quelques inconvénients :

- Les organisations doivent investir un capital substantiel dans l'acquisition de ressources informatiques pour une solution en particulier.
- Les organisations sont seules responsables de la gestion de leurs solutions informatiques. Elles doivent engager des administrateurs système pour superviser le matériel et les logiciels. Les organisations doivent également payer pour l'alimentation et le refroidissement du matériel. Par conséquent, outre les coûts initiaux, elles doivent budgétiser les coûts récurrents.

- La solution informatique a généralement une taille fixe et doit être modifiée pour être mise à l'échelle quand les besoins augmentent ou diminuent.

Avec la disponibilité croissante des applications, du stockage, des services et des machines basés sur le cloud, les entreprises et les consommateurs ont désormais accès à une multitude de ressources informatiques à la demande en tant que services accessibles sur Internet.

2.3. Composants du cloud

Le cloud computing permet l'utilisation de ressources informatiques en tant que service sur le réseau. Bien entendu, toutes les exigences de service ne sont pas identiques. Chaque cas d'utilisation représente un modèle de service bien défini.

Avant d'évoquer des modèles de service, considérons les couches de matériel et de logiciels qui constituent le fondement de la fourniture de services cloud utilisant les différents modèles de service. La figure qui suit propose une abstraction utile du cloud computing en le divisant en quatre couches : logiciels d'application, plateformes de développement, partage de ressources et infrastructure.

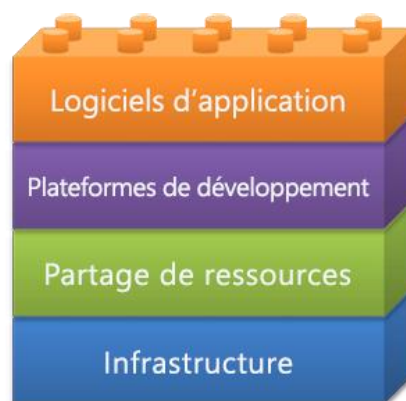


Figure 2.1. - Piles cloud computing

Il est possible de quantifier davantage les quatre couches comme suit :

- Logiciels d'application : La couche supérieure de la pile est celle des logiciels d'application. Il s'agit normalement des composants système dont se sert l'utilisateur final.
- Plateformes de développement : cette couche permet aux développeurs d'applications d'écrire des logiciels d'application à l'aide de l'interface de programmation d'application (API) du fournisseur de cloud.
- Partage de ressources : elle utilise une combinaison de matériel et de logiciels pour faciliter le partage des ressources physiques tout en offrant un certain niveau d'isolation.
- Infrastructure : Les ressources physiques constituent la couche inférieure et, en matière de cloud computing, elles sont principalement déployées du côté fournisseur de cloud.

Ensuite, voyons des modèles de services cloud populaires et la manière dont les abstractions illustrées ci-dessus facilitent leur livraison.

2.4. Les modèles de services cloud

Dans une large mesure, les services cloud varient en fonction des besoins des différents utilisateurs. Cette section passe en revue trois types populaires de services cloud :

- SaaS (software as a service)
- PaaS (platform as a service)
- IaaS (infrastructure as a service)

2.4.1 Software as a Service (SaaS)

Les fournisseurs SaaS sont des applications basées sur le cloud auxquelles les utilisateurs accèdent à la demande depuis Internet sans avoir besoin d'installer ou de maintenir le logiciel. Les exemples incluent GitHub, Google Docs, Slack et Adobe Creative Cloud. Les applications SaaS sont populaires parmi les entreprises et les utilisateurs généraux, car elles sont souvent faciles à adopter, accessibles depuis n'importe quel appareil et disposent de versions gratuites, premium et d'entreprise de leurs applications. Comme le PaaS, le SaaS fait abstraction de l'infrastructure sous-jacente de l'application logicielle afin que les utilisateurs ne soient exposés qu'à l'interface avec laquelle ils interagissent.

2.4.2 Platform as a Service (PaaS)

PaaS fournit une plate-forme informatique où l'infrastructure sous-jacente (telle que le système d'exploitation et d'autres logiciels) est installée, configurée et maintenue par le fournisseur, permettant aux utilisateurs de concentrer leurs efforts sur le développement et le déploiement d'applications dans un environnement testé et standardisé. Le PaaS est couramment utilisé par les développeurs de logiciels et les équipes de développeurs car il réduit la complexité de la configuration et de la maintenance de l'infrastructure informatique, tout en prenant en charge la collaboration entre les équipes distribuées. PaaS peut être un bon choix pour les développeurs qui n'ont pas besoin de personnaliser leur infrastructure sous-jacente, ou ceux qui souhaitent se concentrer sur le développement plutôt que sur DevOps et l'administration système.

2.4.3 Infrastructure as a Service (IaaS)

IaaS est la livraison à la demande d'une infrastructure informatique, y compris les systèmes d'exploitation, la mise en réseau, le stockage et d'autres composants d'infrastructure. Agissant un peu comme un équivalent virtuel des serveurs physiques, IaaS soulage les utilisateurs du cloud de la nécessité d'acheter et de maintenir des serveurs physiques tout en offrant la flexibilité d'évoluer et de payer les ressources selon les besoins. L'IaaS est une option populaire pour les entreprises qui souhaitent tirer parti des avantages du cloud et qui disposent d'administrateurs système capables de superviser l'installation, la configuration et la gestion des systèmes d'exploitation, des outils de développement et des autres infrastructures sous-jacentes qu'elles souhaitent utiliser.

Cependant, IaaS est également utilisé par les développeurs, les chercheurs et autres qui souhaitent personnaliser l'infrastructure sous-jacente de leur environnement informatique.

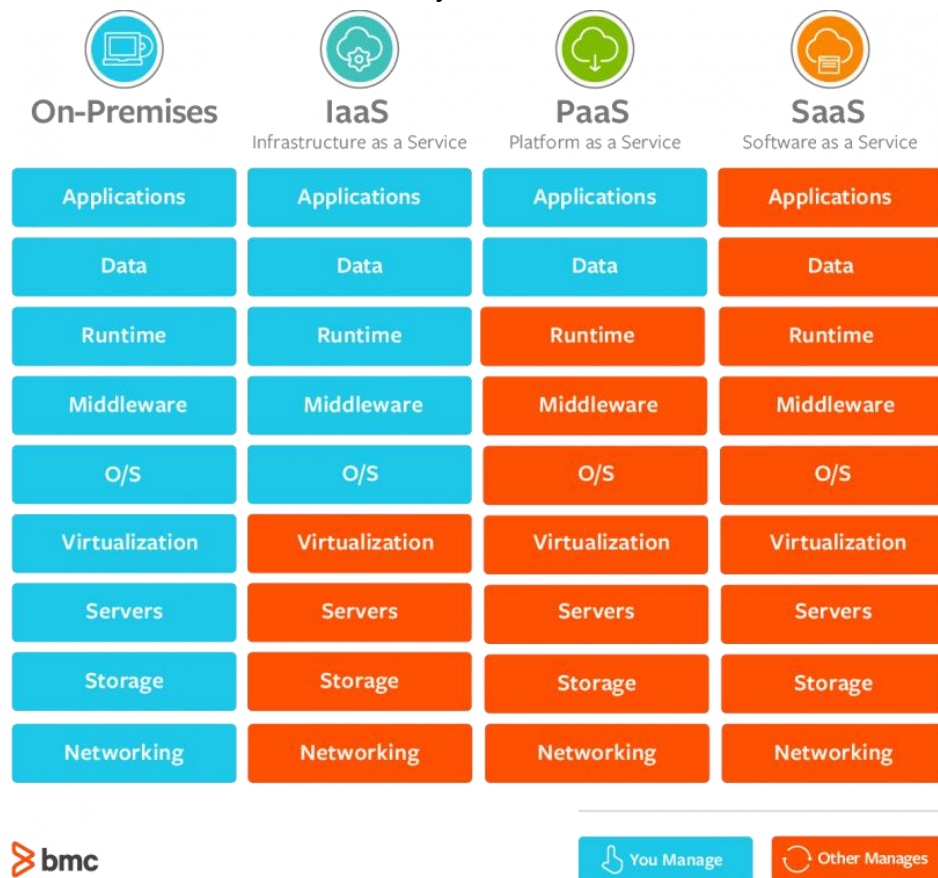


Figure 2.2. - Modèles de services cloud

2.5. Les types de cloud

Il existe quatre modèles connus de déploiement de cloud computing : les clouds public, privé, hybride et communautaire. Un cloud public appartient à un fournisseur de cloud, mais est mis à la disposition du public. Un cloud privé appartient généralement à une organisation. Un cloud communautaire appartient généralement à plusieurs organisations. Un cloud hybride est une combinaison de clouds public, privé et communautaire. Nous présentons les différents types dans cette partie.

2.5.1. Cloud public

Dans un cloud public, l'infrastructure cloud appartient à un fournisseur de cloud et est accessible au public via Internet. Le fournisseur de cloud héberge l'infrastructure cloud, et les utilisateurs finaux peuvent y accéder à distance sans avoir à acheter et configurer un environnement de travail (par exemple, acheter du matériel et des logiciels). Les ressources cloud publiques sont partagées entre différents utilisateurs finaux. Les utilisateurs de cloud public sont généralement facturés pour la durée d'utilisation de ces services. Toutefois, les modèles de facturation de cloud public varient selon les fournisseurs. La sécurité et les conditions d'utilisation

sont définies par le fournisseur et, par conséquent, les utilisateurs finaux doivent respecter les contraintes du fournisseur quand ils utilisent ses services.

2.5.2. Cloud privé

Dans ce deuxième type de cloud, l'infrastructure cloud appartient à une organisation. Cette infrastructure est accessible à des utilisateurs spécifiques via l'intranet de l'organisation. L'environnement cloud doit être acquis, configuré, exploité et géré par l'organisation elle-même. Les ressources d'un cloud privé sont généralement partagées entre les utilisateurs finaux d'une organisation. Contrairement au cloud public, la sécurité et les conditions d'utilisation d'un cloud privé sont définies par l'organisation. L'ensemble de l'infrastructure étant située au sein de l'organisation, la sécurité peut être conforme aux stratégies de celle-ci.

2.5.3. Cloud hybride

Dans un cloud hybride, l'infrastructure comprend un cloud privé en pleine propriété et un cloud public loué. Les clouds hybrides ouvrent la voie au « cloud bursting » (éclatement de cloud). C'est-à-dire qu'une organisation utilise son cloud privé pour la plupart de ses besoins et provisionne des ressources dans le cloud public de manière dynamique quand l'utilisation dépasse la capacité de son cloud privé.

2.5.4. Cloud communautaire

Le cloud communautaire permet à plusieurs organisations de partager des ressources en mode Cloud, qui sont alors exclusivement dédiées à ces organisations. Le Cloud communautaire peut être géré par des organisations membres ou par un fournisseur externe. Le cloud communautaire peut également permettre à plusieurs utilisateurs de construire un cloud aux caractéristiques d'un cloud privé en termes de sécurité et de ressources dédiées, à moindre coût et avec une garantie d'indépendance vis-à-vis d'un prestataire de services de cloud public

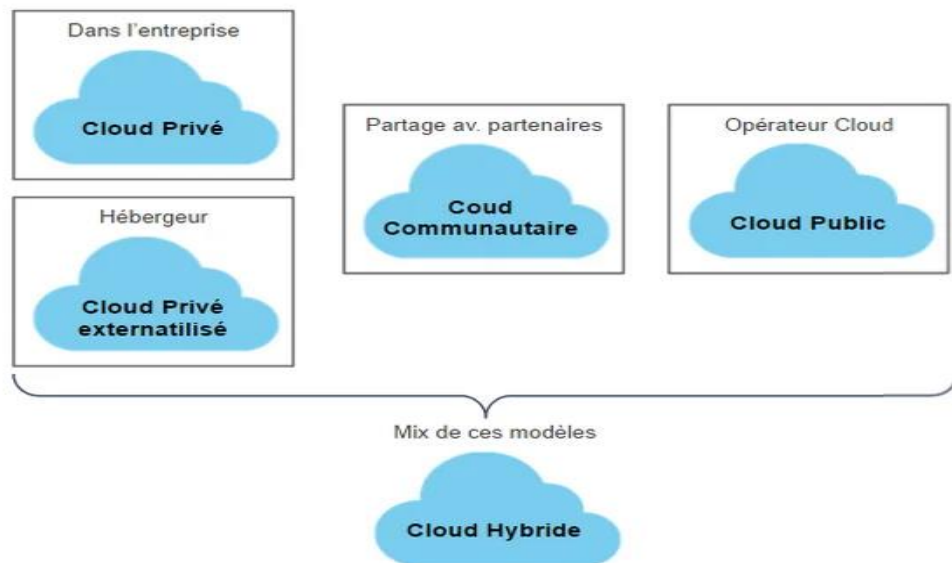


Figure 2.3. - Types de cloud

2.6. Principaux fournisseur cloud

Nous allons maintenant passer en revue brièvement des piles cloud actuellement populaires sur le marché. Nous examinerons rapidement les services proposés par les principaux fournisseurs de cloud, dont, Microsoft Azure, Google Cloud Platform et , Amazon Web Services.

2.6.1. Microsoft Azure

Microsoft Azure est la plateforme cloud qui connaît la croissance la plus rapide du marché, avec un chiffre d'affaires impressionnant et un portefeuille de services en constante expansion. Azure est disponible dans une centaine de pays et possède des centres de données réparties un peu partout dans le monde. Des sous-ensembles d'Azure sont disponibles via Azure Stack. Cela permet à une organisation de créer un cloud privé pouvant se connecter au cloud public Azure et interagir avec celui-ci sans difficulté. Ces sous-ensembles permettent d'automatiser amplement les centres de données internes, en utilisant des ressources partagées capables de répondre à des pics soudains de demande.

Azure propose plus de 100 services IaaS, PaaS et SaaS, dont les suivants :

- **Calcul** : Microsoft propose le service Azure Virtual Machines, qui peut être configuré pour exécuter Windows ou différentes versions de Linux.
- **Stockage** : La famille de services appelée Stockage Azure propose plusieurs solutions de stockage, à savoir : Blobs Azure, Files d'attente Azure, Azure Files.
- **Mise en réseau** : Microsoft propose également des services de réseau privé virtuel via Azure Virtual Network.
- **Produits PaaS** : Azure propose plusieurs produits PaaS, dont Azure App Service, Azure SQL Database et Azure Cosmos DB.

- **Produits SaaS** : Microsoft propose une gamme de services SaaS, qui incluent notamment Microsoft 365 et OneDrive.

2.6.2. Google Cloud Platform

À l'origine, Google Cloud Platform ne proposait que des produits et API PaaS intégrés dans les produits les plus puissants de Google, tels que l'API de traduction. La plateforme GCP s'est depuis diversifiée dans de multiples services en réponse aux offres de ses concurrents. Elle compte environ 60 centres de données répartis dans 20 régions autour du monde.

- **Calcul** : La principale plateforme IaaS de Google est le moteur de calcul Google Compute Engine (GCE) qui prend en charge des machines virtuelles Linux de différentes tailles.
- **Stockage** : Google propose trois services de stockage principaux, dont Google Cloud Storage, un service de stockage d'objets similaire à AWS S3 et aux blobs Azure.
- **Mise en réseau** : Google propose plusieurs produits de réseau permettant de gérer les connexions entre les services cloud de Google et le monde extérieur, notamment des services d'équilibrage de charge, d'interconnexion et DNS.
- **Produits PaaS** : La principale offre PaaS de Google est Google App Engine (GAE), plateforme sans serveur complètement managée gérée pour l'hébergement d'applications similaires à AWS Elastic Beanstalk et Azure App Service.
- **Produits SaaS** : Google propose certains des services SaaS les plus populaires au monde, dont Gmail et Google Drive.

2.6.3. Amazon Web Service

AWS est un leader du marché dans plusieurs segments du cloud computing, en particulier dans le domaine de l'IaaS. Amazon Web Services a commencé par la banalisation et la location de services développés en interne par l'équipe d'ingénierie d'Amazon pour le grand public. AWS a initialement proposé S3, service de stockage d'objets, puis EC2, nuage de calcul élastique. AWS est actuellement la plus grande plateforme cloud, rapportant plus de 25 milliards de dollars de revenus en 2018. Sa croissance est de près de 50 % par an. AWS comprend plus de 100 services répartis entre IaaS, PaaS et SaaS. Ses principaux composants sont les suivants :

- **Calcul** : La solution de calcul principale d'Amazon est Elastic Compute Cloud (EC2), qui fournit aux utilisateurs des machines virtuelles, ou instances, de différentes capacités louées à l'heure ou à long terme.
- **Stockage** : AWS propose plusieurs produits dans cet espace. Le stockage de blocs est effectué sur des volumes Elastic Block Storage (EBS), qui peuvent être attachés à des instances EC2 et détachés de celles-ci.
- **Mise en réseau** : Les solutions Virtual Private Cloud (VPC), Elastic Load Balancer (ELB) et Route 53 d'Amazon sont des services réseau utilisables pour gérer la connectivité entre vos instances et services déployés dans AWS et dans le monde extérieur.

- **Produits PaaS** : Les plateformes d’AWS sont vastes et variées pour répondre aux différents besoins des applications. AWS fournit une suite de plateformes d’analyse telles que Elastic MapReduce (EMR), Amazon Kinesis et Redshift.
- **Produits SaaS** : AWS inclut un vaste éventail de produits SaaS, dont beaucoup ciblent les domaines en constante évolution de l’apprentissage automatique et de l’intelligence artificielle. Il s’agit, par exemple, d’Amazon Polly pour la conversion de texte en parole, d’Amazon Recommendations pour l’amélioration des applications de commerce.

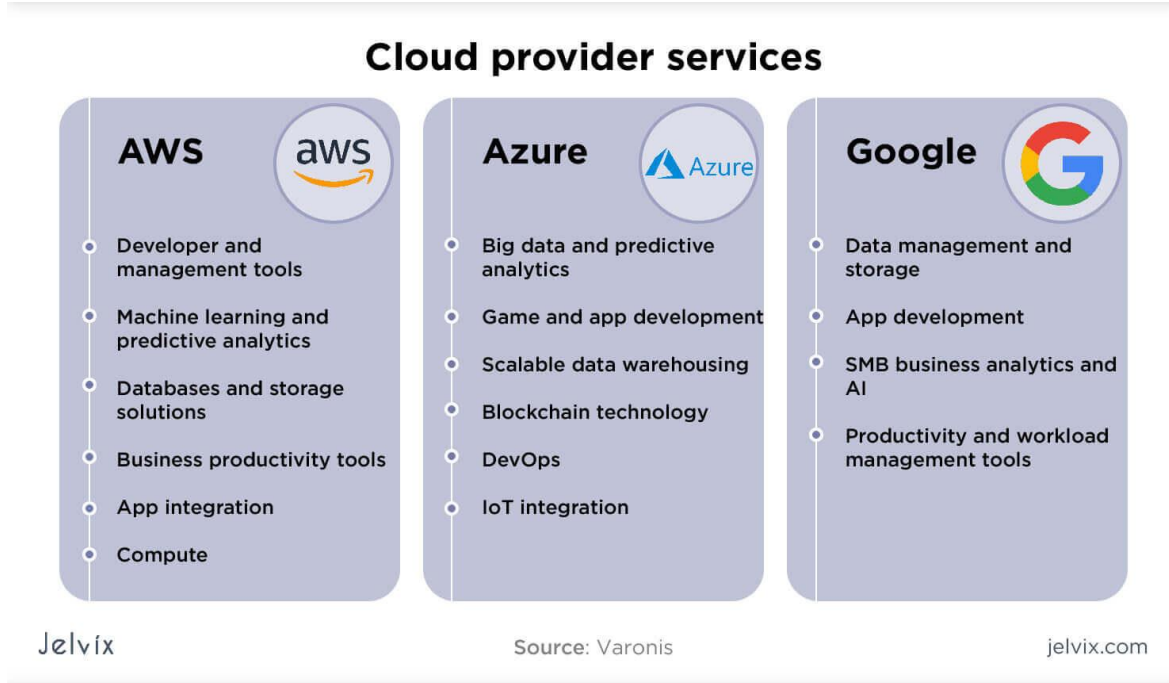


Figure 2.4. - Fournisseurs Cloud

2.7. Caractéristique du cloud computing

Le National Institute of Standards and Technology (NIST), une agence du Département du commerce des États-Unis répertorie les cinq caractéristiques essentielles du cloud computing :

- **Libre-service à la demande** : les ressources cloud peuvent être consultées ou provisionnées sans interaction humaine. Avec ce modèle, les consommateurs peuvent obtenir un accès immédiat aux services cloud lors de leur inscription.
- **Large accès au réseau** : les utilisateurs peuvent accéder aux services et aux ressources cloud via n'importe quel appareil et dans n'importe quel emplacement du réseau à condition qu'ils en aient l'autorisation.
- **Mise en commun des ressources** : les ressources du fournisseur de cloud sont partagées par plusieurs locataires tout en gardant les données des clients individuels cachées aux autres clients.
- **Élasticité rapide** : contrairement au matériel et aux logiciels sur site, les ressources de cloud computing peuvent être rapidement augmentées, réduites ou modifiées en fonction des besoins changeants de l'utilisateur du cloud.

- **Service mesuré** : l'utilisation des ressources cloud est mesurée afin que les entreprises et les autres utilisateurs du cloud n'aient à payer que pour les ressources qu'ils utilisent au cours d'un cycle de facturation donné.

Ces caractéristiques offrent une grande variété d'opportunités de transformation pour les entreprises et les particuliers, dont nous parlerons plus tard dans la section Avantages du Cloud Computing.

2.8. Vue d'ensemble de la virtualisation

La virtualisation est une technologie qui vous permet de créer plusieurs environnements simulés ou ressources dédiées à partir d'un seul système physique. Son logiciel, appelé hyperviseur, est directement relié au matériel et vous permet de fragmenter ce système unique en plusieurs environnements sécurisés distincts. C'est ce que l'on appelle les machines virtuelles. Ces dernières exploitent la capacité de l'hyperviseur à séparer les ressources du matériel et à les distribuer convenablement.

Le matériel physique doté d'un hyperviseur est appelé « hôte », tandis que toutes les machines virtuelles qui utilisent ses ressources sont appelées « invités ». Ces invités traitent les ressources de calcul (processeur, mémoire, stockage) à la manière d'un pool de ressources qui peut être déplacé sans difficulté. Les opérateurs peuvent contrôler les instances virtuelles du processeur, de la mémoire, du stockage et des autres ressources, de sorte que les invités reçoivent les ressources dont ils ont besoin, lorsqu'ils en ont besoin.

2.8.1. Avantages de la virtualisation

La virtualisation est principalement utilisée par les programmeurs pour simplifier le développement et le test de logiciels. Elle est utilisée par les centres de données informatiques pour convertir les serveurs dédiés en matériel plus économique et par le cloud pour isoler les utilisateurs qui partagent une seule couche matérielle et offrir une élasticité, entre autres fonctionnalités.

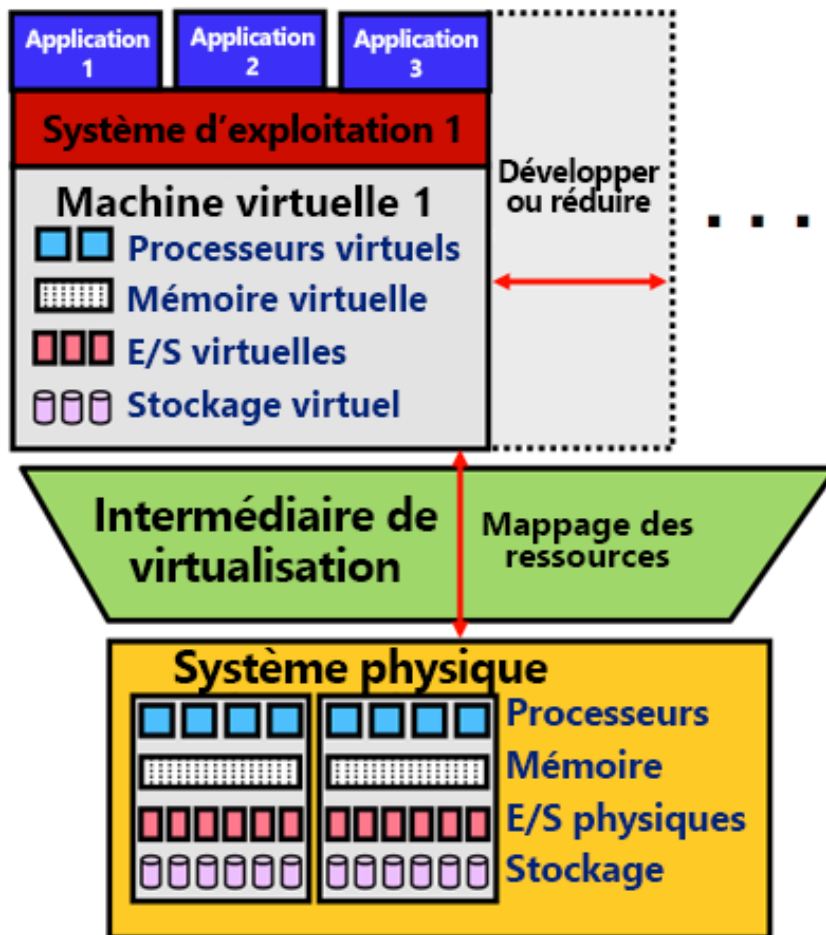


Figure 2.5. - Provisionnement machine virtuelle

2.8.2. Les types de virtualisation

- Virtualisation de type 1 :

C'est une technique de virtualisation qui présente à la machine invitée une interface logicielle similaire mais non identique au matériel réel. Ainsi, il permet aux systèmes d'exploitation invités d'interagir directement avec le système d'exploitation hôte et, par conséquent, ils seront conscients de la virtualisation.

- Virtualisation de type 2 :

Il s'agit d'une technique de virtualisation qui crée un environnement entièrement virtuel. En utilisant ces techniques, le système d'exploitation invité n'interagit pas directement avec le système d'exploitation hôte et pense donc qu'il s'exécute sur une vraie machine physique. Cette technique de virtualisation permet uniquement la virtualisation des SE avec la même architecture matérielle que l'hôte.

- L'isolation :

L'isolation : il est possible de diviser un système d'exploitation en plusieurs espaces mémoire ou plusieurs contextes. Chaque contexte est géré par le système d'exploitation hôte.

Cette isolation permet d'exécuter plusieurs fois la même application pour ne s'exécuter qu'une seule fois par machine. Les programmes de chaque contexte ne peuvent communiquer qu'avec les processus et les ressources associés à leur contexte. L'isolement n'est lié qu'aux systèmes Linux.

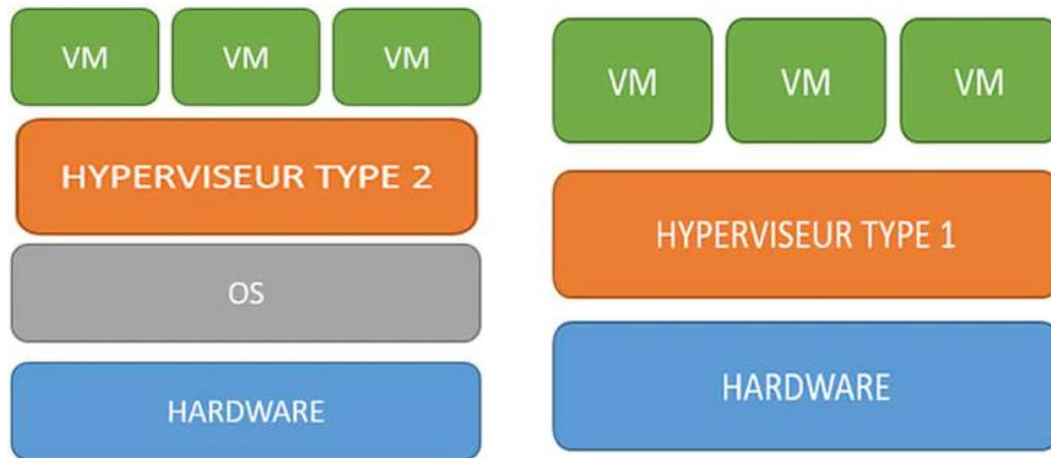


Figure 2.6. - Types de virtualisations

2.8.3. Types de ressources virtualisées

- Virtualisation des données

Les données éparpillées dans un environnement peuvent être regroupées sous la forme d'une source unique. La virtualisation des données permet aux entreprises d'utiliser les données comme une source dynamique. Ainsi, elles profitent de fonctionnalités de traitement capables de rassembler les données issues de plusieurs sources, d'héberger facilement de nouvelles sources de données et de transformer les données pour répondre aux besoins des utilisateurs.

- Virtualisation des serveurs

Les serveurs sont des ordinateurs conçus pour traiter un volume élevé de tâches spécifiques afin de permettre aux autres ordinateurs d'effectuer diverses autres tâches. La virtualisation d'un serveur permet d'optimiser l'exécution de ces fonctions spécifiques et implique son partitionnement, de sorte que les composants puissent être utilisés pour exécuter différentes fonctions.

- Virtualisation des S/E

La virtualisation des systèmes d'exploitation intervient au niveau du noyau, qui correspond au gestionnaire de tâches central de ces derniers. Cette approche permet notamment d'exécuter des environnements Linux et Windows côte à côte.

- Virtualisation des postes de travail

Souvent confondue avec la virtualisation des systèmes d'exploitation, qui vous permet de déployer plusieurs systèmes d'exploitation sur une seule machine, la virtualisation des postes de travail permet à un administrateur central de déployer des environnements de postes de travail simulés sur des centaines de machines physiques en même temps. Contrairement aux environnements de postes de travail classiques que vous devez installer, configurer et mettre à jour physiquement sur chaque machine, les postes de travail virtualisés peuvent être configurés, mis à jour et vérifiés simultanément par un administrateur.

- Virtualisation des fonctions réseaux

La virtualisation des fonctions réseau (NFV) sépare les fonctions clés d'un réseau (telles que les services d'annuaire, le partage de fichiers et la configuration des adresses IP) afin de les répartir entre les différents environnements. Lorsque les fonctions logicielles sont indépendantes des machines physiques sur lesquelles elles étaient hébergées, il est possible de regrouper des fonctions spécifiques dans un nouveau réseau et de les assigner à un environnement. La virtualisation des réseaux réduit le nombre de composants physiques nécessaires à la création de plusieurs réseaux indépendants, tels que les commutateurs, les routeurs, les serveurs, les câbles et les hubs. Elle est particulièrement répandue dans le secteur des télécommunications.

2.9. Avantages et inconvénient du cloud computing

- Avantages du cloud

Le cloud computing offre une variété d'avantages aux particuliers, aux entreprises, aux développeurs et à d'autres organisations. Ces avantages varient en fonction des objectifs et des activités des utilisateurs du cloud.

- ✓ **Modèle économique** : Les organisations estiment généralement leurs besoins informatiques pour une période de 1 à 5 ans à l'avance selon un processus appelé planification de la capacité. La planification de la capacité amène les organisations à estimer les investissements informatiques en tenant compte des pics de charge. Cette approche qui peut parfois entraîner une surcapacité ou une capacité insuffisante (ce qui peut entraîner une dégradation du service). Avec le modèle économique de paiement à l'utilisation, les organisations paient les ressources dont elles ont besoin. Elles n'ont plus à payer de coûts initiaux, à investir dans l'achat d'une infrastructure informatique onéreuse pour gérer leur infrastructure.
- ✓ **Gestion informatique simplifiée** : Les utilisateurs de services cloud n'ont pas besoin de consacrer du temps à la configuration, à l'exploitation et à la gestion de leurs ressources informatiques. De son côté, le fournisseur de cloud, en concurrence pour l'acquisition de clients, investit des ressources conséquentes dans la gestion et la maintenance de ses offres avec une fiabilité élevée.
- ✓ **Scalabilité** : Dans un environnement informatique interne traditionnel, les organisations peuvent prendre beaucoup de temps pour acquérir, configurer et commencer à exploiter une infrastructure informatique. Les fournisseurs de services cloud louent des ressources informatiques en quelques minutes. Les clients peuvent augmenter l'échelle des ressources à la demande et la réduire pendant les périodes d'accalmie afin de réaliser des économies. C'est pourquoi les clouds offrent la propriété importante d'**élasticité** qui permet de provisionner et de dé provisionner des ressources de manière dynamique ou programmatique, afin de s'adapter aux charges de travail changeantes.
- ✓ **Utilisation améliorée** : Le cloud computing améliore sensiblement l'utilisation des ressources physiques, car celles-ci sont partagées entre tous les utilisateurs (elles sont multilocataires). Grâce à la virtualisation, les serveurs sont désormais consolidés en tant qu'images de système d'exploitation partageant les mêmes ressources système.

L'utilisation est ainsi améliorée. Cela permet de réaliser des économies globales au niveau de l'alimentation et du refroidissement, ainsi que de réduire l'empreinte carbone.

- ✓ **Déploiement rapide et global** : En recourant aux services de fournisseurs de cloud computing qui ont une présence mondiale au travers de leurs centres de données, des start-ups peuvent rivaliser avec des acteurs déjà établis en déployant rapidement des applications et services vers un public mondial. C'est particulièrement important pour les start-ups de réseaux sociaux, qui peuvent être confrontées à des tendances de croissance virale à mesure que leurs services deviennent populaires dans plusieurs pays.

- Inconvénients du cloud

En adoptant des services cloud computing, les utilisateurs et les organisations peuvent bénéficier des avantages ci-dessus. Cependant, l'utilisation de ces services présente plusieurs risques, tels que les suivants :

- ✓ **Enfermement propriétaire** : le cloud computing s'est progressivement standardisé. OpenStack est une plateforme de cloud computing open source qui vise à standardiser la pile logicielle du cloud computing. Toutefois, elle n'est pas entièrement compatible avec tous les clouds publics. L'absence de standardisation peut conduire à une situation d'enfermement propriétaire, par exemple, quand un client s'inscrit à un service cloud non standard, développe des applications pour celui-ci et y déploie des données. En l'absence d'une standardisation, il est peu probable que le client puisse passer aisément à un autre fournisseur.
- ✓ **Risques liés à la sécurité** : étant donné que le cloud computing avec des clouds publics peut entraîner l'envoi de données d'une organisation à l'extérieur de son périmètre, la sécurité devient une préoccupation majeure. Pour certains domaines, c'est tout simplement inacceptable pour divers utilisateurs ou organisations.
- ✓ **Risques liés à la confidentialité** : L'utilisation du cloud soulève également de nombreuses préoccupations en lien avec la confidentialité. En fonction de la législation applicable à un fournisseur de services cloud, les gouvernements ont parfois le pouvoir de rechercher et de saisir des données du fournisseur sans consentement explicite ni notification du client. De plus, les clients ne peuvent pas être totalement assurés de la confidentialité des données lorsqu'ils utilisent des clouds publics.
- ✓ **Risques liés à la fiabilité** : les clouds connaissent eux aussi des problèmes de fiabilité. Les clouds publics font leur possible pour optimiser la durée de bon fonctionnement et la fiabilité. Néanmoins, des pannes peuvent se produire. Les clients doivent concevoir dans la perspective de défaillances possibles, et utiliser des fonctionnalités telles que des zones de disponibilité multiples, dans lesquelles ils pourront configurer des infrastructures redondantes vers lesquelles basculer en cas de défaillance.

2.10. Conclusion

Dans ce chapitre, nous avons donné un aperçu sur la technologie du Cloud Computing. Comme nous avons vu que le Cloud pense à résoudre les problèmes posés dans les autres technologies. De plus, le Cloud représente une révolution car il offre des services de plusieurs types. Grâce aux

bénéfices offerts par le fournisseur du Cloud, les utilisateurs finaux et les entreprises trouvent que le Cloud est un bon choix pour l'utilisation de ces services. Avec toutes ces solutions et les avantages produits dans le Cloud, il existe toujours des limites que nous avons détaillés dans ce chapitre.

Internet des Objets

3.1. Introduction

Un objet connecté est un objet communicant intelligent, capable de capter de l'information, la traiter pour un objectif donné et la retransmettre par un réseau de communication ou vers d'autres objets connectés.

Une fois la donnée captée et prétraitée par l'objet connecté, elle est communiquée, puis stockée sur des serveurs (privés ou dans le cloud). Elle pourra être traitée en vue d'une action d'amélioration d'un processus interne ou d'un service aux utilisateurs qu'ils soient consommateurs, entreprises, collectivités ou gouvernements. On parle alors d'Internet des Objets ou Internet of Things (IoT).

Le premier objet connecté daterait du début des années 80. Le concept « d'internet of Things » est né en 1999 au centre Auto-ID du MIT. L'idée était d'associer un tag RFID à chaque objet du monde réel pour pouvoir à tout instant et à distance : l'identifier, l'inventorier et tracer ses déplacements.

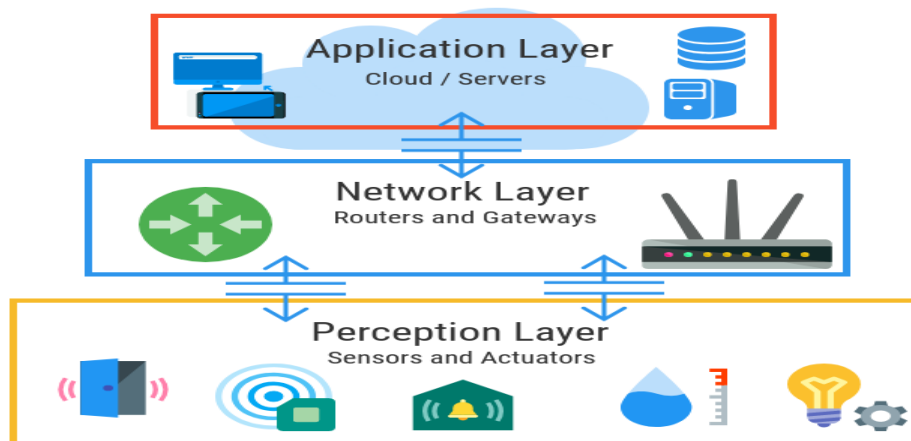
Par la suite, la miniaturisation de l'électronique, sa faible consommation et les performances croissantes des réseaux de communication ont permis d(e) :

- Identifier l'objet à plus grande distance,
- Transmettre une information plus riche issue des capteurs embarqués de plus en plus nombreux.

La collecte de ces données et la possibilité d'effectuer des actions à distance permet d'améliorer les activités opérationnelles et la rentabilité, ainsi que de créer des nouveaux services.

3.2. Architecture de l'internet des objets

L'architecture d'un système IoT est composée de plusieurs niveaux qui communiquent entre eux pour relier le monde tangible des objets au monde virtuel des réseaux et du cloud. Tous les projets n'adoptent pas une architecture formellement identique. Donc chaque couche est définie par ses fonctions et les périphériques utilisés dans cette couche. L'architecture de l'IoT est généralement divisée en trois couches, la couche perception, la couche réseau et la couche application.



3.1. Couche de l'architecture IoT

3.2.1. Couche perception

La couche de perception est également connue sous le nom de couche « Capteurs » dans l'IoT. Le but de cette couche est d'acquérir les données de l'environnement à l'aide de capteurs et d'actionneurs. Cette couche détecte, collecte et traite les informations, puis les transmet à la couche réseau.

3.2.2. Couche réseau

La couche réseau de l'IoT sert à la fonction de routage et de transmission des données vers différents hubs et appareils IoT sur Internet. À cette couche, les plates-formes de cloud computing, les passerelles Internet, les dispositifs de commutation et de routage ...etc.

3.2.3. Couche application

La couche application est considérée comme une couche supérieure de l'architecture IoT conventionnelle. Cette couche fournit des services personnalisés en fonction des besoins des utilisateurs. La responsabilité principale de cette couche est de relier l'écart majeur entre les utilisateurs et les applications.

En résumé...

Un capteur commence par relever une mesure.

La mesure relevée par le capteur est ensuite envoyée à la passerelle grâce à un premier protocole de communication. Le rôle de la passerelle est de traduire les protocoles pour établir une communication entre les objets et le réseau (souvent le cloud). Parfois, une intelligence embarquée dans la passerelle permet le traitement et le stockage de la donnée, ainsi que des fonctions de pilotage de l'objet. Pour les protocoles courte portée (Zigbee, Z-Wave, WiFi, ...), la passerelle est locale et se connecte souvent à la box du fournisseur d'accès à internet. Pour les protocoles longue portée (LoRa, LTEM, 3G/4G, ...), la passerelle se trouve sur le réseau de l'opérateur télécom.

La plateforme IoT est une plateforme technique qui collecte les données, surveille, et contrôle les objets connectés. Les mesures peuvent être stockées dans l'objectif de créer une historisation permettant la prédiction. La plateforme IoT délivre un service digital à l'utilisateur final restituant les données collectées des objets, permettant des actions de pilotage et apportant une couche d'intelligence (alertes, conseils, ...).

L'utilisateur bénéficie, d'une interface dédiée sur son terminal. Il peut alors piloter son objet, accéder à l'historique des mesures relevées, et utiliser les différentes fonctionnalités prévues par la solution IoT.

3.3. Objet connecté

Les objets connectés sont des capteurs ou des objets dotés de capteurs capables de communiquer des données à travers un réseau. Les données sont généralement envoyées à un ordinateur, une tablette, un smartphone ou tout autre appareil électronique et parfois via Internet pour que l'information soit accessible sur tous les appareils pouvant s'y connecter.

Les informations que l'appareil peut envoyer sont par exemple la température ambiante de votre maison, des images de votre intérieur, votre rythme cardiaque, etc. Tout cela dépend de l'objet, de si c'est une caméra, une montre ou autre.

Ces appareils peuvent aussi recevoir des informations, pour par exemple vous donner la possibilité d'allumer votre four cinq minutes avant votre rentrée du travail, de lancer votre machine à café à distance, d'allumer votre aspirateur robot, etc.

La plupart des objets connectés ont besoin d'information pour être utile. Une information, c'est tout ce qui entoure, que ce soit l'humidité dans l'air, le battement de votre cœur, le mouvement exercé sur eux, la température ou un signal électrique transmis depuis votre téléphone en appuyant sur un bouton.

Il y a plusieurs critères de classification des types des objets, Il existe deux grands modèles des objets connectés qui sont :

3.3.1. Les objets portés

De manière générale, les termes « technologie wearable », « dispositifs wearable » et « vêtements intelligents » se réfèrent tous à des technologies électroniques ou à des micro-ordinateurs qui sont incorporés dans des vêtements et des accessoires, qui peuvent être portés sur le corps.

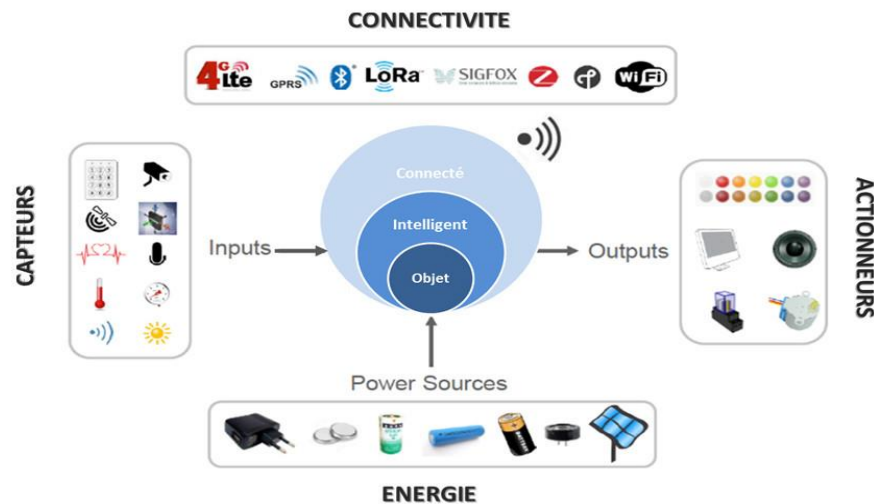
3.3.2. Les objets indépendants

Tous les objets non portables avec des capteurs déployés à plusieurs endroits, comme les stations météo, les voitures intelligentes, les drones...etc.

Il existe une autre classification pour déterminer les types des objets connectés, ils peuvent être classés en grande partie soit passifs soit actifs en fonction de la source d'énergie détectée.

3.4. Composants d'un système IoT

Un système IoT réunit de nombreux acteurs et composants technologiques. Il est composé de capteurs, d'actionneurs, de la connectivité et d'une source d'énergie pour assurer l'alimentation du système.



3.2. Composant d'un système IoT

3.4.1. Les capteurs

Les capteurs sont des dispositifs permettant de transformer une grandeur physique observée (température, luminosité, mouvement etc...) en une grandeur digitale utilisable par des logiciels. Il existe une très grande variété de capteurs de tous types.

Exemple de capteurs : lumière, présence, position, déplacement, accélération, température, humidité, son, vibration, électrique, magnétique, chimique, gaz, flux, force, pression, niveau, ...

Il existe de nombreux capteurs disponibles pour l'IoT et un certain nombre de façons de les catégoriser. Les catégories décrites ci-dessous ne sont qu'un petit échantillon des façons dont les capteurs peuvent être regroupés. Les capteurs peuvent être divisés par :

- Leurs besoins en alimentation externe

Type	Définition	Exemple
Passive	Ne nécessite pas d'alimentation externe pour fonctionner. Ils répondent aux entrées de leur environnement.	Un capteur de température qui modifie la résistance en réponse aux changements de température
Active	Nécessite une alimentation externe pour fonctionner.	Une camera

- Type de signal produit par le capteur

Type	Définition	Exemple
Analogique	Émet un signal analogique continu	Accéléromètres, capteurs de température
Numérique	La sortie est convertie en valeurs discrètes (1 et 0 numériques) avant de transmettre à un appareil	Capteur de pression numérique, capteur de température numérique

- Type d'appareil de mesure

Type	Définition	Exemple
Chimique	Répond aux changements chimiques dans son environnement	Capteur de gaz
Mécanique	Répond aux changements physiques de son environnement	Micro-interrupteur
Electrique	Répond aux changements électriques dans son environnement	Capteur optique

3.4.2. Les actionneurs

Les actionneurs sont des dispositifs qui transforment une donnée digitale en phénomène physique pour créer une action, ils sont en quelque sorte l'inverse du capteur.

Exemple d'actionneurs : Afficheurs, Alarmes, Caméras, Haut-parleurs, Interrupteurs, Lampes, Moteurs, Pompes, Serrures, Ventilateur, ...

3.4.3. Les sources d'énergies

Les sources d'énergie sont de 4 types : alimentation filaire pour les objets ayant accès à une prise de courant, piles ou batteries pour ceux qui n'y ont pas accès ou de manière occasionnelle (recharge), capteurs d'énergie ou « Energy harvesting » (photovoltaïque, thermoélectrique, cinétique...) pour rallonger la durée de vie des objets à très faible consommation, et enfin les objets passifs sans piles qui sont alimentés par les ondes électromagnétiques des lecteurs (RFID, NFC...).

L'énergie est un des grands défis des objets connectés, tant pour garantir la plus grande durée de vie possible sans maintenance, que pour garantir un respect environnemental malgré la multiplication des objets connectés énergivores qui envahissent nos espaces personnels et professionnels.

3.4.4. La connectivité

La connectivité de l'objet est assurée par une petite antenne Radio Fréquence qui va permettre la communication de l'objet vers un ou plusieurs réseaux (qui sont détaillés dans la section « réseaux IoT »). Les objets pourront d'une part remonter des informations telles que leur identité, leur état, une alerte ou les données de capteurs, et d'autre part recevoir des informations telles que des commandes d'action et des données. Le module de connectivité permet aussi de gérer le « cycle de vie de l'objet », c'est-à-dire, l'authentification et l'enregistrement dans le réseau, la mise en service, la mise à jour et la suppression de l'objet du réseau.

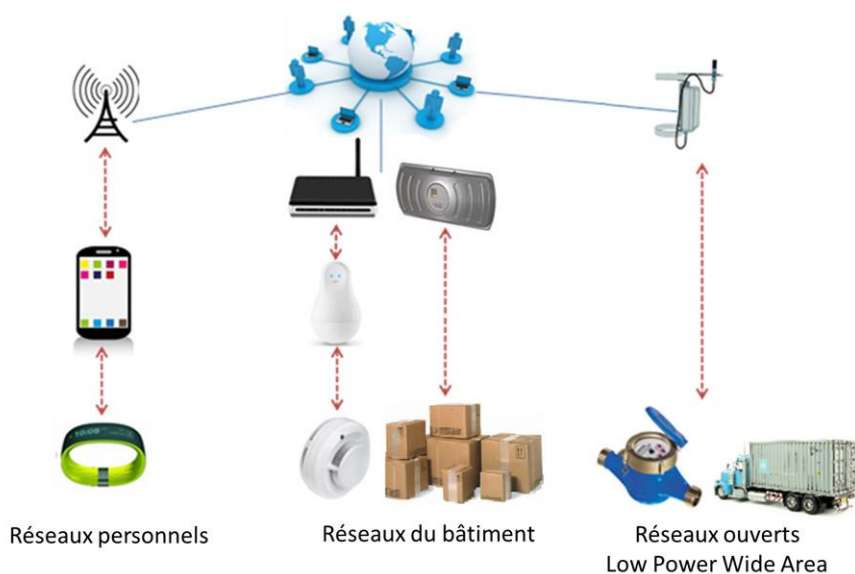
3.5. Réseaux IoT

Depuis les années 2010, la multiplication des objets connectés grand public et professionnels a entraîné la création ou la transformation de multiples réseaux dédiés : les réseaux attachés à la personne, « indoor » pour les bâtiments et « outdoor » à l'échelle des territoires, des pays ou même mondiaux.

Les réseaux IoT se sont focalisés sur la performance énergétique pour augmenter :

- ✓ La durée de vie des objets sur batterie,
- ✓ La distance de communication pour diminuer les coûts d'infrastructure
- ✓ Le coût de service pour permettre de connecter plus d'objets.

Un réseau sera choisi en fonction de nombreux critères tels que : disponibilité, déploiement, maturité, espérance de vie, débit de données, distance et type de communication, objets fixes ou mobiles, localisation des objets, niveau de sécurité, type d'opérateur de réseau, durée de vie des batteries, disponibilité des modules, type de plateforme IoT, coût (OPEX/CAPEX), maintenance, interopérabilité....



3.3. Réseaux IoT

3.5.1. Les réseaux personnels

Ce sont les réseaux grand public attachés à la personne, orientés autour du sport & bien-être, des loisirs, des médias, des réseaux sociaux et de l'efficacité personnelle. Ces réseaux sont donc très souvent liés aux smartphones ou à des dispositifs miniaturisés et portables. Les principaux réseaux sont :

- Bluetooth (BLE, BLE Smart, etc...) qui permet une connectivité facile à quelques mètres en utilisant un protocole de communication très largement diffusé et disponible nativement sur de très nombreux appareils.
- LiFi, La technologie LiFi appartient à la famille des VLC (Visible Light Communication). Il s'agit d'un réseau optique sans fil qui utilise des LED pour transmettre des données. L'information est codée à travers la fluctuation d'un signal lumineux émis par une LED. Les informations communicantes sont très rapides et ne sont pas détectées par l'œil humain (la vitesse de transmission est de l'ordre du MHz).

3.5.2. Les réseaux du bâtiment

Les principaux réseaux du bâtiment sont :

- BLE industriel, issu du Bluetooth grand public fiabilisé pour l'industrie, permet la création de réseaux denses dans les entrepôts ou le tertiaire pour suivre les biens et les capteurs, avec une distance de communication typique de 20 à 200m. Pouvant associer les étiquettes BLE avec des balises, des répéteurs et des routeurs, ces réseaux permettent de très nombreuses applications allant jusqu'à la géolocalisation à l'intérieur des bâtiments.
- RFID, Le sigle RFID signifie Radio Frequency Identification, comprenez radio-identification. Il désigne une méthode utilisée pour stocker et récupérer des données à distance en utilisant des balises métalliques, les « Tags RFID ». Ces balises, qui peuvent être collées ou incorporées dans des produits, réagissent aux ondes radio et transmettent des informations à distance.
- Zigbee : Le Zigbee est un protocole de communication similaire au Bluetooth mais dédié à l'IoT. Il consomme peu, et est fait pour envoyer de petits volumes de données (entre 20 et 250 Kb/s). Il permet d'utiliser chaque objet comme "rallonge de connexion".
- WiFi, c'est une technologie mise au point en 1997. Elle assure la connexion de plusieurs appareils (ordinateurs portables ou de bureau, tablette, smartphone, imprimante, box) entre eux et à Internet sans passer par des fils ou des câbles. Il émet des fréquences situées entre 2,4 GHz à 5 GHz. Il est régi par les normes IEEE 802.11.
- 6LoWPAN, Le protocole 6LoWPAN a été développé pour définir l'adaptation d'IPv6, ainsi que la manière de transporter les datagrammes IP sur des liaisons IEEE 802.15.4 et d'exécuter les fonctions de configurations nécessaires pour former et maintenir un sous-réseau IPv6 (Internet Protocol version 6).

3.5.3. Les réseaux LPWAN (Low Power Wide Area Network)

Ce sont les nouveaux réseaux (cellulaires ou non) dédiés aux objets connectés en extérieur. Ils sont idéaux pour remonter des informations simples de capteurs et de localisation vers des plateformes de gestion d'objets, sur le cloud ou sur des réseaux privés. Ces réseaux sont basés sur des protocoles privilégiant :

- ✓ La très basse consommation (autonomie de 5 à 10 ans)
- ✓ La très longue distance de communication (plusieurs km)
- ✓ La bonne couverture à l'intérieur des bâtiments
- ✓ La très forte densité d'objets connectables
- ✓ Les faibles coûts d'opération

La contrepartie de ces avantages est le débit limité de communication et la limitation du nombre de communications bidirectionnelles (à partir de l'objet ou vers l'objet).

Les principaux réseaux LPWAN sont :

- LoRa, réseau cellulaire concurrent de Sigfox dédié aussi aux objets connectés, qui a vu ses premiers déploiements massifs à partir de 2017. Il utilise un protocole de communication ouvert géré par la LoRa Alliance, qui permet aux utilisateurs de créer leur propre réseau, interne ou bien d'utiliser les différents réseaux nationaux et internationaux des opérateurs.
- Les réseaux cellulaire, GSM : supportant uniquement les appels et SMS, 2G : rendant possible l'envoi de MMS, 3G : initiant l'utilisation de l'Internet mobile, 4G : permettant le haut débit sur mobile, par exemple le streaming vidéo HD), 5G en cours de commercialisation (particulièrement adapté à l'IoT très gourmand en data, comme les voitures autonomes).
- Réseaux propriétaires ou marché : ingenu, RPMA, Wimax, Qowisio, Weightless

	Standard	Fréquence	Couverture	Débit
Lifi	Similar to 802.11	400–800 THz	<10m	<224 Gbps
Wifi	802.11a/b/g/n/ac	2.4 GHz and 5 GHz	~50 m	~400 Mbps
Cellular	GSM/GPRS/EDGE (2G), UMTS/HSPA (3G), LTE (4G), 5G	900, 1800, 1900, and 2100MHz 2.3, 2.6, 5.25, 26.4, and 58.68 GHz	<200 Km	<500 kps (2G) <2Mbps (3G) ~100 Mbps (4G) ~1000 Mbps (5G)
Bluetooth	Bluetooth 4.2	2.4 GHz	15 m	1 - 3 Mbps
RFID/NFC	ISO/IEC 18000-3	13.56 MHz	10 cm	100–420 kbps
6LowPAN	RFC6282	2.4 GHz and ~1 GHz	<20 m	20–250 kbps
ZigBee	ZigBee 3.0 based on IEEE 802.15.4	2.4 GHz	10–100 m	20 - 250 kbps
Z-Wave	Z-Wave Alliance ZAD12837 / ITU-T G.9959	868.42 MHz and 908.42 MHz	<100 m	<100 kbps
Lora	LoRaWAN	868 MHz and 915 MHz	<15m	0.3–50 kbps

3.6. Caractéristiques fondamentales de l'internet des objets

Les caractéristiques fondamentales de l'IoT sont les suivantes :

La connectivité : est une exigence importante de l'infrastructure IoT. Les objets de l'IoT doivent être connectés à l'infrastructure IoT. N'importe qui, n'importe où, n'importe quand, la connectivité doit être garantie à tout moment Sans connexion, rien n'a de sens.

Détection intelligente : les appareils connectés à l'IoT auront des capacités de détection intelligente. Par exemple, l'utilisation de détecteurs de mouvement pour allumer ou éteindre les lumières. La technologie de détection aide à créer des expériences qui reflètent une véritable conscience du monde physique, des personnes et des objets.

Intelligence : les appareils connectés à l'IoT peuvent être dotés d'intelligence. Par exemple, les Nest Learning Thermostats sont compatibles Wi-Fi, pilotés par capteur et dotés de capacités d'auto- apprentissage. Le Misfit Shine est un tracker de fitness avec des capacités de surveillance du sommeil. Le Misfit Shine peut distribuer des tâches de calcul entre un smartphone et le cloud.

Expressive : les appareils connectés à l'IoT ont une capacité unique de dire l'état actuel aux autres appareils connectés dans les environs. Ils donnent un meilleur flux de communication entre l'homme et les machines.

Sécurisé : les appareils connectés à l'IoT peuvent contribuer à garantir la sécurité de la vie individuelle. Par exemple, un pneu de voiture en mouvement peut indiquer son état actuel au propriétaire de la voiture ayant des tableaux de bord de voiture intelligents, cela aidera à prévenir les accidents dus à l'éclatement des pneus de voiture en raison d'une surchauffe, ...etc.

3.7. Domaine d'application de l'internet des objets

IoT est utilisé dans plusieurs domaines, nous citons comme exemples :

3.7.1. Maison intelligente

La maison intelligente, ou domotique, est une extension de l'automatisation du bâtiment, avec laquelle nous pouvons surveiller et contrôler le chauffage, la ventilation et la climatisation (CVC), l'éclairage, les appareils électroménagers, les systèmes de sécurité à bande. En connectant tous les appareils électroménagers, nous pouvons automatiser de nombreuses routines quotidiennes, comme allumer et éteindre automatiquement les lumières et le chauffage, démarrer ou arrêter la cuisson et le lavage, etc. Avec le réseau intelligent et les compteurs intelligents, nous pouvons réduire les consommations d'énergie et les factures de services publics, et avec les systèmes de sécurité, nous pouvons rendre la maison plus sûre en détectant automatiquement et en dissuadant, espérons-le, les intrusions en utilisant divers capteurs infrarouges, de mouvement, sonores, de vibration ainsi que des systèmes d'alarme.

3.7.2. Santé

L'IoT permet la surveillance à distance de la santé et les systèmes de notification d'urgence. Une approche très populaire consiste à utiliser des techniques portables. Ces appareils portables peuvent collecter une gamme de données de santé, telles que la fréquence cardiaque, la température corporelle et la pression artérielle, qui peuvent ensuite être transmises sans fil à un site distant pour le stockage et une analyse plus approfondie. Cela permet également à la télésanté/télémédecine, c'est-à-dire de diagnostiquer ou de traiter les patients à distance.

3.7.3. Industrie

L'application de l'IoT dans l'industrie est souvent appelée Industrie 4.0 ou quatrième révolution industrielle. La première révolution industrielle a eu lieu au XVIIIe siècle lorsque la machine à vapeur a mobilisé la production industrielle. La deuxième révolution industrielle a eu lieu au début du XIXe siècle, lorsque l'énergie électrique alimentait la production de masse. La troisième révolution industrielle, ou révolution numérique, a eu lieu à la fin du XIXe siècle lorsque l'électronique et l'informatique ont encore automatisé la production. L'industrie 4.0 s'appuie sur des systèmes cybers - physiques qui intègrent étroitement les machines, les logiciels, les capteurs, Internet et les utilisateurs. Il créera des usines intelligentes, dans lesquelles les machines pourront utiliser l'auto optimisation, l'auto configuration et même l'intelligence artificielle pour effectuer des tâches complexes afin de fournir des économies de coûts largement supérieures et des biens ou services de meilleure qualité.

3.8. Protocoles de l'internet des objets

Les protocoles, ou les protocoles de communication, sont un ensemble de règles qui permettent aux appareils de communiquer entre eux. Les protocoles définissent la syntaxe, la sémantique et la synchronisation de la communication. Une analogie étroite avec les protocoles est celle des langues humaines. Il existe de nombreux protocoles de communication disponibles pour les applications IoT. Les protocoles suivants sont couramment utilisés : HTTP, Websocket et MQTT.

3.8.1. Hypertext Transfer Protocol

Le protocole de transfert hypertexte (Hypertext Transfer Protocol : HTTP) est le protocole de communication derrière le World Wide Web (WWW). Il est basé sur une architecture client-serveur et fonctionne à la manière des demandes et des réponses. HTTP utilise TCP (protocole de contrôle de transmission) pour fournir des connexions fiables.

HTTP est un protocole sans état, car le client et le serveur ne maintiennent pas de connexion pendant la communication.

3.8.2. Websocket

WebSocket est un protocole de communication conçu pour les navigateurs Web et les serveurs Web, mais contrairement à HTTP, WebSocket fournit une communication en duplex intégral sur une seule connexion TCP. WebSocket est avec état, car le client et le serveur maintiennent une connexion pendant la communication. Le WebSocket permet une plus grande interaction entre un navigateur et un serveur Web, permet un transfert de données en temps réel et des flux de messages. À ce jour, WebSocket est implémenté dans tous les principaux navigateurs Web, par exemple Firefox 6, Safari 6, Google Chrome 14, Opera 12.10 et Internet Explorer 10.

3.8.3. MQ Telemetry transport (MQTT)

MQ Telemetry transport (MQTT) est un protocole de communication machine à machine légère conçue pour les appareils IoT par IBM. MQTT est basé sur un modèle publisher-subscriber, où publisher publie des données sur un serveur (également appelé broker), et le subscriber s'abonne au serveur et reçoit des données du serveur. Le broker MQTT est responsable de la distribution des messages et peut-être quelque part dans le Cloud.

3.9. Système embarqué

3.9.1. Définition

Un système embarqué est un système inclus dans un autre système, il est défini comme un système électronique et informatique autonome souvent temps réel, spécialisé dans une tâche bien précise. Ses ressources sont généralement limitées, le système comprend une partie matérielle et une autre logicielle qui sont conçues spécifiquement pour réaliser une fonction dédiée. Il contient généralement un ou plusieurs microprocesseurs destinés à exécuter un ensemble de programmes définis lors de la conception et stockés dans des mémoires.

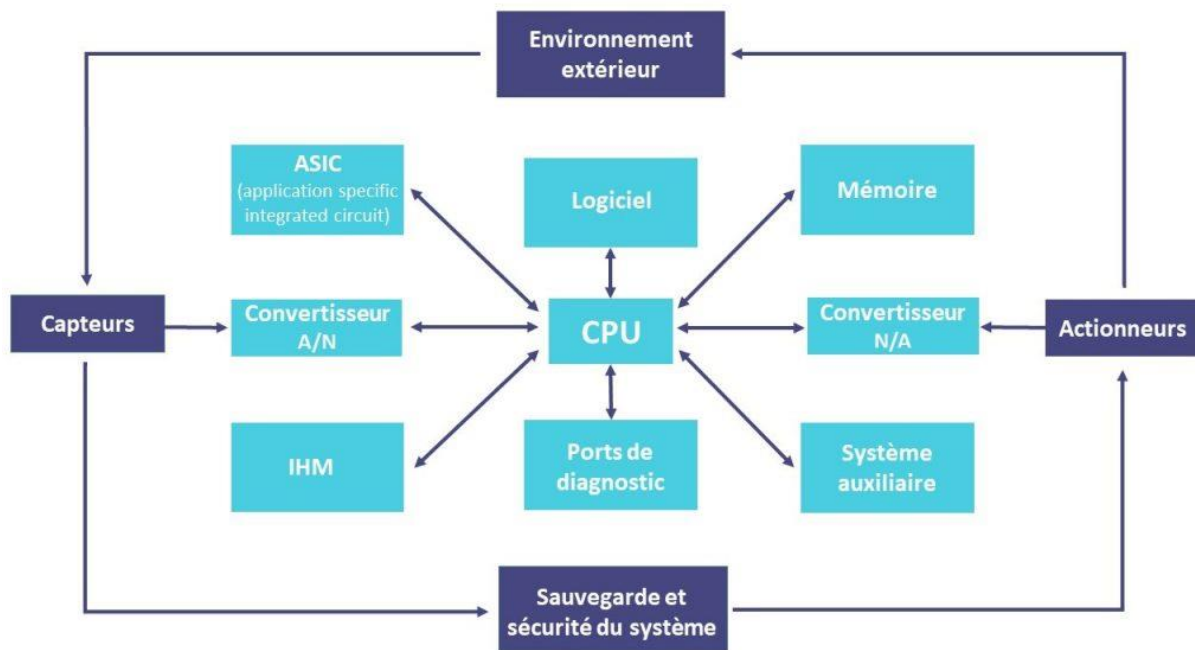
3.9.2. Caractéristiques des systèmes embarqués

Les systèmes embarqués ont pour but de permettre aux objets usuels de réagir à l'environnement, ils peuvent aussi apporter une interface avec l'utilisateur. Généralement ils sont caractérisés par :

- Coût réduit, maximisation rapport performance/prix.
- Capacité mémoire adaptée.
- Volume restreint (compact, pas modulaire).
- Exécution temps réel (souvent).
- Fiabilité et sécurité de fonctionnement.
- Consommation d'énergie maîtrisée (très faible en cas d'utilisation sur batterie)
- Capacité de calcul appropriée à l'application.

La structure de base de ces systèmes est donnée par la figure qui suit : l'environnement est mesuré par divers capteurs. L'information des capteurs est échantillonnée pour être traitée par le

cœur du système embarqué. Puis le résultat du traitement est converti en signaux analogiques qui génèrent les actions sur l'environnement (afficheur d'informations pour l'utilisateur, actionneurs, transmission d'informations ... etc.).



3.4. Système embarqué

3.10. Conclusion

Dans ce chapitre nous avons fait un survol sur l'internet des objets en général en fournissant la définition d'un objet connecté et de l'architecture de l'IoT. On a vu aussi l'historique de l'IoT et les raisons de cette évolution. Ensuite on a expliqué les technologies et les domaines d'application qui seront touchés par l'émergent d'internet des objets ainsi que les systèmes embarqués.

Etude de plateforme IoT (Microsoft IoT Central et Amazon AWS IoT)

4.1. Introduction

De nombreuses organisations font face à des pressions concurrentielles et à l'évolution rapide des besoins des clients. En réponse, les organisations s'engagent dans une transformation numérique : elles repensent leurs activités en rassemblant les personnes, les données et les processus. Dans ce contexte, l'IoT (Internet of Things) est le « ciment » qui permet la transformation numérique dans une organisation.

Cependant, pour que les grandes entreprises puissent bénéficier des avantages de la transformation numérique, il peut être difficile de déployer et de gérer des appareils IoT dans différentes zones géographiques. En outre, le véritable enjeu de l'IoT se situe dans la donnée. Les capteurs connectés enregistrent des informations en local ou les envoient directement vers un datacenter. Il faut pouvoir traiter ce flux afin d'en tirer toute l'intelligence nécessaire pour une activité. C'est là qu'intervient la plateforme IoT. Sa structure repose majoritairement sur les technologies du Cloud. Basiquement, elle fait le pont entre l'IoT et le Big Data. En réalité, son fonctionnement est bien plus complexe. La plateforme peut intervenir au niveau des équipements réseau, des serveurs, des systèmes embarqués, du logiciel et même au niveau du hardware d'un objet connecté.

Dans cette partie, nous faisons une étude comparative de deux des principales plateformes IoT à savoir Microsoft IoT Central et Amazon AWS IoT.

4.2. Définition d'une plateforme IoT

Une plateforme IoT, ou IoT platform, est une technologie multicouche qui permet d'approvisionner, de gérer et d'automatiser facilement les appareils connectés de l'Internet des objets. Elle connecte le matériel des entreprises au cloud en utilisant des options de connectivité flexibles, des mécanismes de sécurité de niveau entreprise et de larges puissances de traitement des données.

Les plateformes IoT contribuent à :

- Connecter le matériel, comme les capteurs et les dispositifs,
- Gérer différents protocoles de communication hardware et software,
- Assurer la sécurité et l'authentification des dispositifs et des utilisateurs,
- Collecter, visualiser et analyser les données recueillies par les capteurs et les appareils,
- Intégrer tout ce qui précède avec les systèmes d'entreprise existants et d'autres services web.

4.3. Types de plateforme IoT : open source, propriétaire

Il existe deux types de plateformes IoT : les plateformes open source et les plateformes propriétaires.

- Les plateformes open-source : le logiciel est distribué librement et le code-source est ouvert à tous. Vous devez développer l'ensemble des services et vous occuper de la maintenance de la plateforme vous-même, il vous faut donc assez bien de connaissances et de ressources pour faire cela. Cela peut prendre un certain temps, mais heureusement, vous pouvez compter sur la communauté pour vous aider.
- Les plateformes propriétaires : généralement, vous souscrivez aux services et vous partagez la responsabilité avec le prestataire qui fournit et maintient les environnements.

4.4. Microsoft IoT Central

4.4.1. Présentation de Microsoft IoT Central

Microsoft IoT Central est une solution SaaS dans le cloud qui aide les organisations à créer, utiliser et gérer des produits intelligents. Logiciel en mode SaaS, Microsoft IoT Central est donc compatible avec la plupart des systèmes d'informations d'entreprises ainsi qu'avec la plupart des systèmes d'exploitation (OS) comme Windows, Mac OS, et Linux car il est accessible depuis un navigateur web (Chrome, Firefox ...). Ce progiciel est également accessible à distance (bureau, domicile, en déplacement...) depuis de nombreux appareils mobiles comme une tablette ou un smartphone iPhone (plateforme iOS) ou Android, et dispose sans doute d'une application mobile disponible sur le Play Store / App Store. Pour l'utiliser, il est important d'avoir une connexion Internet correcte ainsi qu'un navigateur à jour.

4.4.2. Avantages

Parmi les avantages de Microsoft IoT Central on peut citer :

- Communication entre objets connectés
- Gestion à distance
- Surveillance des objets connectés
- Accessibilité 24-7
- Import – Export des données

4.4.3. Fonctionnalités

Les fonctionnalités clés de l'intégration d'Azure IoT Hub sont les suivantes :

- Gestion des messages de liaison montante : The Things Stack publie des messages de liaison montante vers une application Azure IoT Central
- Approvisionnement automatique des appareils : les appareils finaux sont automatiquement créés dans l'application Azure IoT Central, à l'aide des informations du référentiel d'appareils LoRaWAN afin d'approvisionner le modèle d'appareil final

- Mise à jour de l'état de l'appareil dans Device Twin : mettez à jour les propriétés signalées par l'appareil en fonction des charges utiles décodées et planifiez des liaisons descendantes en fonction des propriétés souhaitées de l'appareil

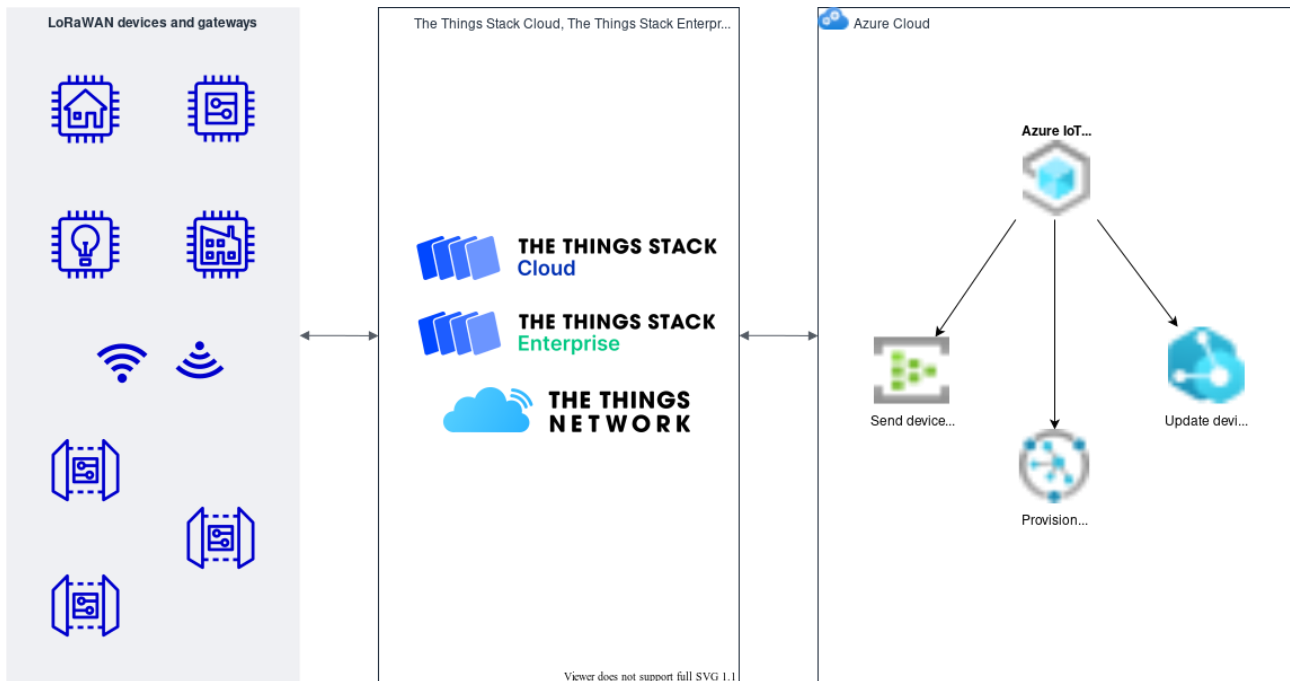


Figure - 4.1. Fonctionnalité Microsoft IoT Central

4.4.4 Architecture

L'intégration d'Azure IoT Central ne nécessite aucune ressource physique supplémentaire dans votre compte Azure. Il se connecte à l'application Azure IoT Central à l'aide du service de provisionnement d'appareils Azure IoT sous-jacent, puis soumet le trafic à l'aide du hub Azure IoT dans lequel l'application a été provisionnée.

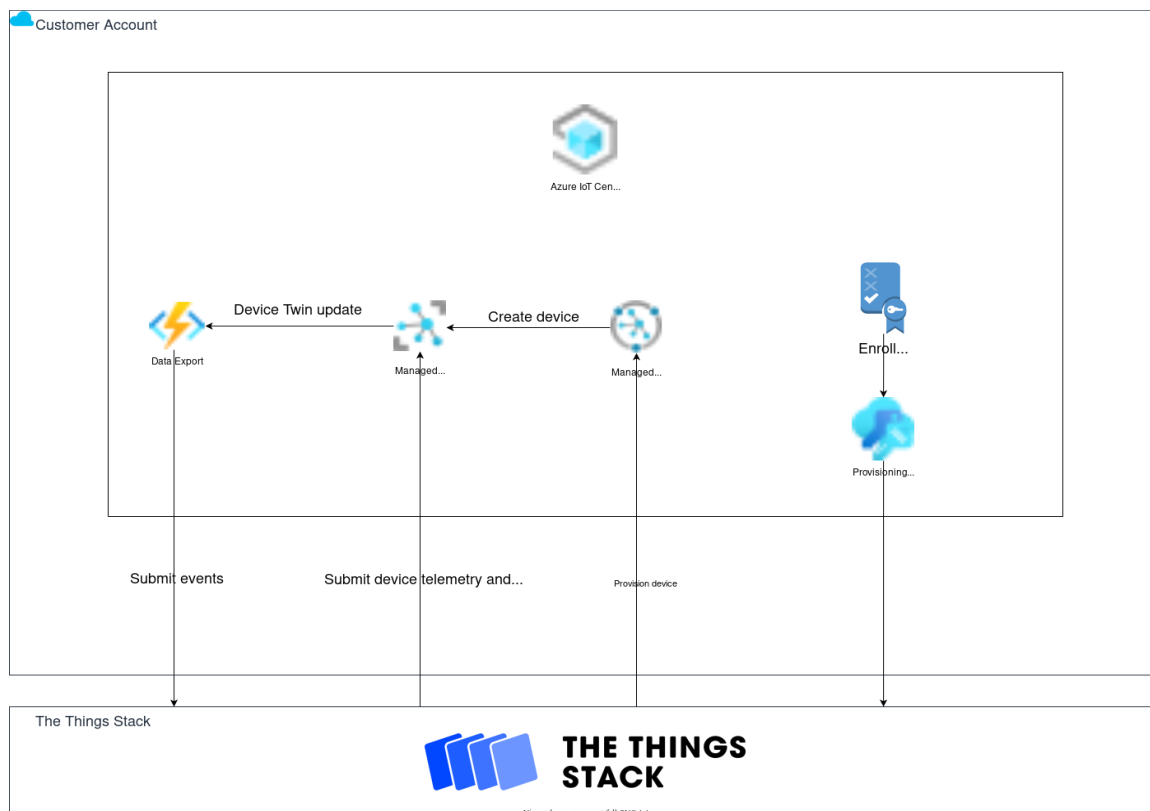


Figure - 4.2. Architecture Microsoft IoT Central

Azure IoT Hub est conçu autour d'appareils finaux autonomes communiquant directement avec le hub. Chaque terminal doit se connecter au concentrateur via l'un des protocoles de communication pris en charge (MQTT / AMQP). Ces protocoles sont par nature avec état - chaque appareil final individuel doit avoir une connexion toujours ouverte afin d'envoyer et de recevoir des messages à partir d'Azure IoT Hub.

Le trafic de liaison descendante, qui correspondrait aux messages cloud-à-appareil d'IoT Hub, se produit rarement au niveau de la couche d'application pour la plupart des cas d'utilisation. En tant que tel, garder une connexion ouverte par périphérique final est à la fois inutile et difficile à mettre à l'échelle, car les deux protocoles de communication mentionnés ci-dessus imposent que chaque périphérique terminal ait sa propre connexion individuelle, et aucune sémantique de groupes d'abonnement n'est disponible.

Sur la base des arguments ci-dessus, l'intégration d'Azure IoT Central préfère utiliser un style de communication asynchrone et sans état. Lorsque des messages de liaison montante sont reçus d'un appareil final, l'intégration se connecte à la demande à Azure IoT Hub et soumet le message, et met également à jour le Device Twin. Le protocole de plan de données utilisé entre The Things Stack et Azure IoT Hub est MQTT, et les connexions sont toujours sécurisées à l'aide de TLS 1.2. Les mises à jour des propriétés souhaitées de Device Twin et les événements de création ou de suppression d'appareils sont reçus par The Things Stack à l'aide d'une exportation de données IoT Central. L'exportation de données soumet les données via des requêtes HTTP qui sont authentifiées à l'aide de la clé API fournie lors du provisionnement de l'intégration, et les connexions sont toujours effectuées via TLS. Ce pipeline permet à The Things Stack d'éviter les longues connexions à Azure IoT Hub.

4.4.5 Intégrations et API

Microsoft IoT Central propose des APIs afin de s'intégrer à d'autres applications informatiques comme Microsoft Azure, Microsoft Dynamics, et Microsoft Excel. Ces intégrations permettent par exemple de se connecter à une base de données, d'échanger des données, ou bien encore de synchroniser des fichiers entre plusieurs programmes informatiques via une extension, un plugin, ou une API (application programming interface / interface de programmation). Le logiciel Microsoft IoT Central peut se connecter à plus de 9 plateformes logicielles afin de faciliter les échanges de données entre applications, d'améliorer votre flux de travail, et de gagner en productivité.

- Microsoft Exchange – Microsoft Office 365
- Microsoft Outlook – Microsoft SharePoint
- Microsoft OneDrive
- CRM

4.5. Amazon AWS IoT

4.5.1. Présentation Amazon AWS IoT

AWS IoT Device Management permet de faire évoluer un parc de périphériques et de réduire les coûts et les efforts liés à la gestion de déploiements de grands périphériques IoT. Logiciel en mode SaaS, Amazon AWS IoT est donc compatible avec la plupart des systèmes d'informations d'entreprises ainsi qu'avec la plupart des systèmes d'exploitation (OS) comme Windows, Mac OS, et Linux car il est accessible depuis un navigateur web (Chrome, Firefox ...). Ce logiciel est également accessible à distance (bureau, domicile, en déplacement...) depuis de nombreux appareils mobiles comme une tablette ou un smartphone iPhone (plateforme iOS) ou Android, et dispose sans doute d'une application mobile disponible sur le Play Store / App Store. Pour l'utiliser, il est important d'avoir une connexion Internet correcte ainsi qu'un navigateur à jour.

4.5.2 Avantages

Voici la liste des principaux avantages de ce logiciel :

- Communication entre objets connectés
- Gestion à distance
- Surveillance des objets connectés
- Accessibilité 24-7
- Import – Export des données

4.5.3 Fonctionnalités

Les fonctionnalités clés de l'intégration par défaut sont :

- Gestion des messages de liaison montante et de liaison descendante : The Things Stack publie des messages de liaison montante sur IoT Core et s'abonne aux messages de liaison descendante que vous avez envoyés depuis IoT Core. En outre, vous recevez tous les événements de liaison descendante en file d'attente, envoyés, en échec, reconnus et non reconnus dans IoT Core
- Création et revendication d'appareils : gérez des éléments dans IoT Core ou revendiquez des éléments en toute sécurité sur The Things Join Server en prouvant la propriété
- Cryptage et décryptage des charges utiles des messages : tirez parti du véritable cryptage LoRaWAN de bout en bout en déballant l'AppSKey du serveur de jonction
- Mise à jour de l'état de l'appareil dans l'ombre : surveillez les appareils à l'aide de mesures utiles signalées dans l'état de l'ombre, comme la qualité du signal et le nombre de passerelles de couverture

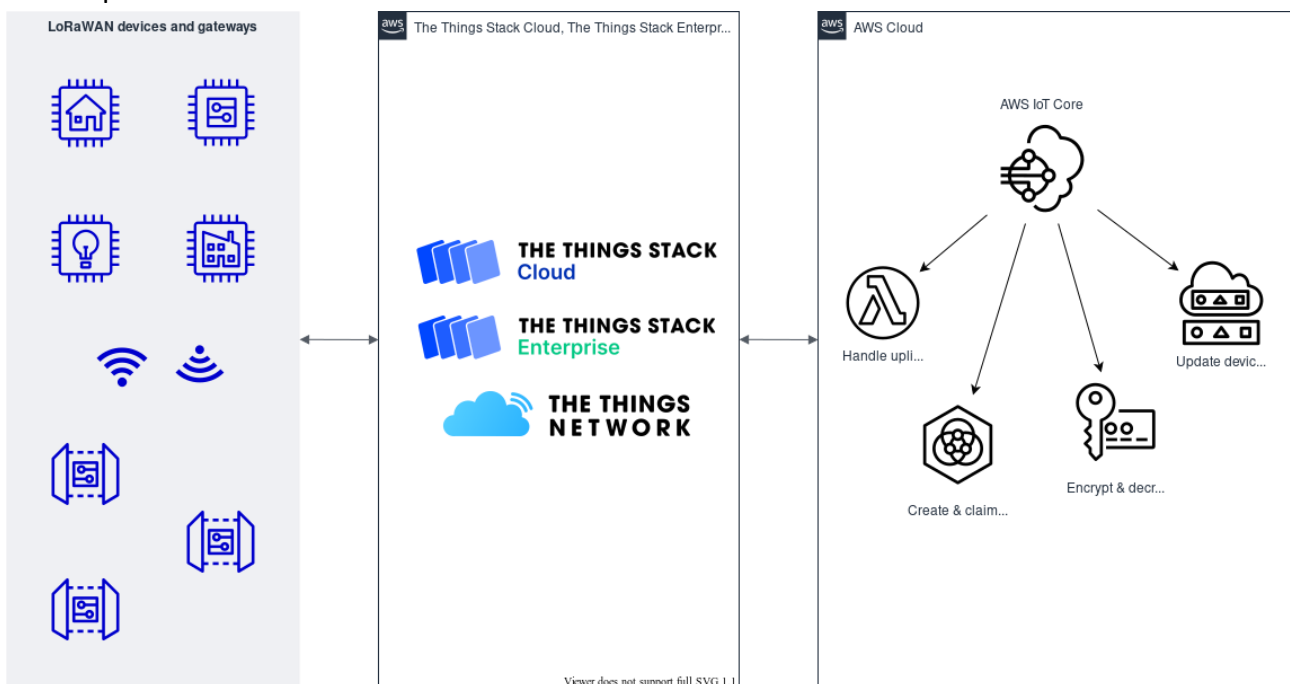


Figure - 4.3. Fonctionnalité Amazon AWS IoT

4.5.4 Architecture

L'intégration AWS IoT Core est un déploiement sans serveur qui évolue automatiquement à mesure que votre déploiement se développe.

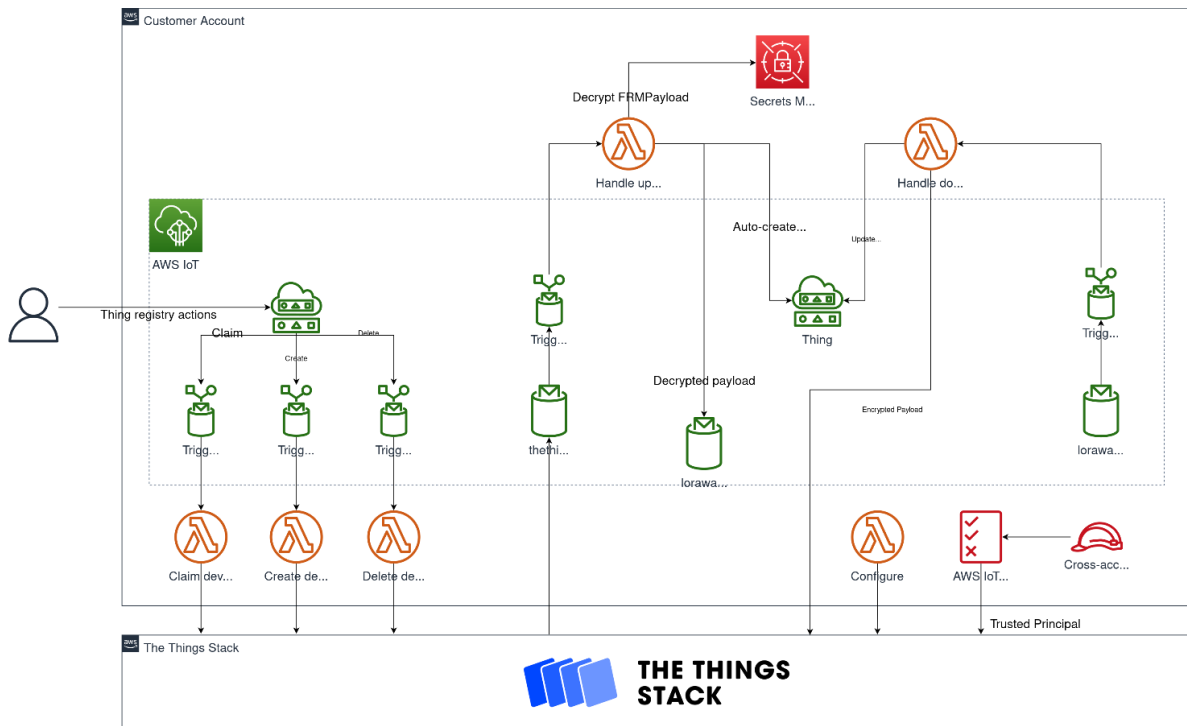


Figure - 4.4. Architecture Microsoft IoT Central

Les principales ressources déployées dans votre compte AWS sont :

- Rôle entre comptes pour que The Things Stack se connecte à votre point de terminaison AWS IoT Core MQTT
- Fonctions AWS Lambda pour créer le type d'objet et configurer l'intégration en tant que pub/sub dans The Things Stack
- Fonctions AWS Lambda pour réclamer et créer des appareils, et pour gérer les messages de liaison montante et descendante
- Secret avec clé de chiffrement à clé (KEK) pour tirer parti du chiffrement de bout en bout LoRaWAN
- Règles IoT Core pour déclencher les fonctions Lambda en fonction des rubriques et des attributs

Il s'agit d'un déploiement sans serveur : aucune ressource de calcul n'est déployée. AWS ne facture que l'utilisation, qui dépend du trafic. Les seuls frais continus sont facturés par la connectivité IoT Core de The Things Stack à votre compte AWS. Toutes les autorisations sont les autorisations minimales pour que l'intégration fonctionne.

4.5.5 Intégrations et API

Amazon AWS IoT propose des APIs afin de s'intégrer à d'autres applications informatiques comme AWS – Amazon Web Services, Amazon EC2, et Amazon S3. Ces intégrations permettent par exemple de se connecter à une base de données, d'échanger des données, ou bien encore de synchroniser des fichiers entre plusieurs programmes informatiques via une extension, un plugin, ou une API (application programming interface / interface de programmation).

Le logiciel Amazon AWS IoT peut se connecter à plus de 5 plateformes logicielles afin de faciliter les échanges de données entre applications, d'améliorer votre flux de travail, et de gagner en productivité.

4.6. Conclusion

Les plateformes IoT aident les entreprises à surmonter les défis techniques sans avoir à rémunérer et à gérer des équipes d'ingénieurs spécialisés dans les différents domaines d'ingénierie requis par l'IoT (la mécanique, l'électricité, les logiciels, etc.). Pour les développeurs, une plateforme IoT offre un ensemble de fonctionnalités prêtes à l'emploi qui accélèrent considérablement le développement d'applications pour les appareils connectés et assurent l'évolutivité et la compatibilité entre les appareils.

Les plateformes IoT sont une composante essentielle de l'écosystème IoT et représentent un marché en pleine croissance, qui devrait dépasser les 22 milliards de dollars d'ici 2023. Elles apportent une valeur ajoutée considérable aux entreprises, en leur permettant de réduire les coûts de développement, d'accélérer le lancement et de rationaliser les processus.

Système d'arrosage automatique avec l'API OpenWeatherMap

5.1. Introduction

Dans le processus du développement, l'implémentation vient après un enchainement de plusieurs étapes et son but principal est de réaliser un produit capable de résoudre les problèmes posés en utilisant des outils et des algorithmes.

Dans le but de réaliser et valider les idées du problème, le présent chapitre montre les outils et les configurations utilisées afin de développer ce système.

Dans ce chapitre nous allons présenter l'architecture du système, les outils et les plateformes utilisés pour développer notre approche. Ensuite, nous présentons les configurations et les résultats obtenus. Enfin, nous discutons sur les résultats pour chaque évaluation.

5.2. Architecture générale du système

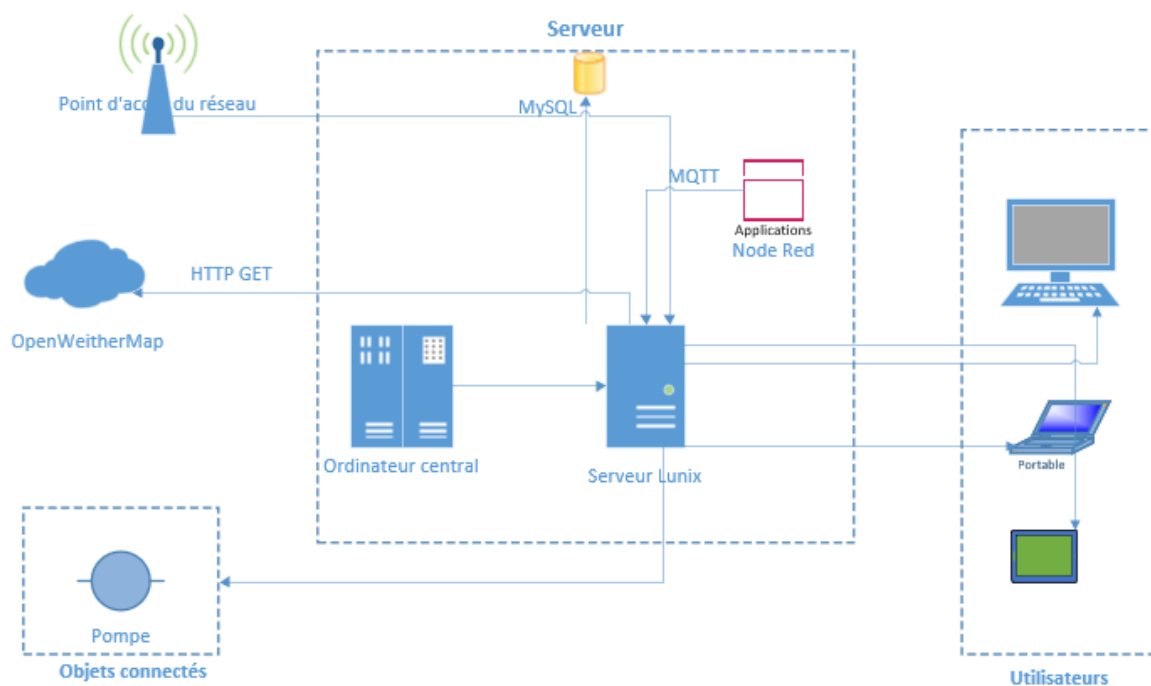


Figure 5.1 - Architecture générale du système

L'idée de ce projet est de réaliser un système d'arrosage automatique qui utilise comme capteur de température et de luminosité les données d'OpenWeatherMap et l'actuateur (pompe dans notre cas) sera représenté par une LED. Les données seront récupérées par l'ESP32 par la méthode HTTP GET et envoyées à node-red par le protocole MQTT. Le tableau de bord de node-red sera utilisé pour visualiser la température, la pression et l'état de la pompe. L'ensemble des mesures seront stockées dans une base de données MySQL.

5.3. Outils et plateforme utilisées

Dans cette section, nous allons présenter les outils et plateforme utilisés pour réaliser notre système. Les implémentations ont été réalisées sur une machine virtuelle Ubuntu sur Windows ; comme microcontrôleur j'ai utilisé un esp32 et Node Red pour visualiser les résultats des tests

5.3.1 Le microcontrôleur esp32

L'ESP32 développé par la société Espressif, est une carte de développement à faible coût dédié à l'internet des objets (IoT) et les applications embarquées. C'est un (SoC) system on a chip doté de communications sans fil Wifi et Bluetooth. Nos capteurs et nos actionneurs seront reliés à l'ESP32 dans notre système.



Figure 5.2 – esp32

Caractéristiques :

- Alimentation : 5 Vcc via micro-USB 3,3 Vcc via broches Vin
- Microprocesseur : Tensilica LX6 Dual-Core
- Fréquence : 240 MHz
- Mémoire SRAM, Flash: 512 kB, 4 Mb
- E/S disponibles : 15 E/S digitales dont 10 compatibles PWM 2 x sorties analogiques (DAC) 15 x entrées analogiques (ADC)
- Interfaces : I2C, SPI, 2 x UART
- Interface Wifi 802.11 b/g/n 2,4 GHz
- Bluetooth : Classique / BLE
- Antenne intégrée

5.3.2 L'outil de programmation Node-Red

Node-red est un outil puissant pour construire des applications de l'Internet des Objets en mettant l'accent sur la simplification de la programmation qui se fait grâce à des blocs de code prédéfinis, appelés « nodes » pour effectuer des tâches. Il utilise une approche de programmation visuelle qui permet aux développeurs de connecter les blocs de code ensemble.

Les nœuds connectés, généralement une combinaison de nœuds d'entrée, de nœuds de traitement et de nœuds de sortie, lorsqu'ils sont câblés ensemble, constituent un flow.

Node-red est construit sur Node.js, tirant pleinement parti de son modèle non bloquant piloté par les événements. Cela le rend idéal pour fonctionner en périphérie du réseau sur un serveur d'application ou un serveur dans le cloud.

5.3.2.1 Installation

Pour installer Node-RED localement, j'aurai besoin d'une version prise en charge de Node.js. Pour cela, je peux utiliser la commande npm fournie avec node.js :

```
#sudo npm install -g --unsafe-perm node-red
```

Si vous utilisez Windows, ne démarrez pas la commande avec sudo.

Cette commande installera Node-RED en tant que module global avec ses dépendances.

Une fois installé en tant que module global, je peux utiliser la commande node-red pour démarrer Node-RED dans mon terminal. Ctrl-C ou fermer la fenêtre du terminal pour arrêter Node-RED.

Je peux ensuite accéder à l'éditeur Node-RED en pointant votre navigateur sur <http://iPServeur:1880>.

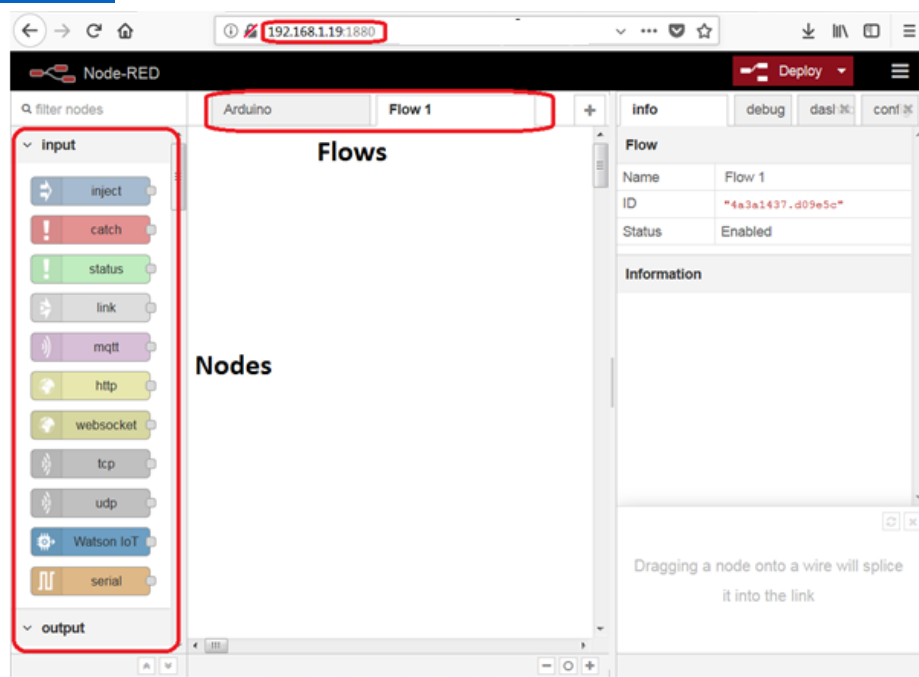


Figure 5.3 – Interface node-red

5.3.2.2 Ajout d'une catégorie de nodes à la palette

La palette de nodes d'origine est déjà bien fournie mais il manque au moins une catégorie essentielle qui va permettre de réaliser l'interface web. Elle se nomme Dashboard. Dans le menu, je clique sur Manage Palette puis recherche la catégorie de nodes à installer. Une nouvelle catégorie est ensuite disponible.

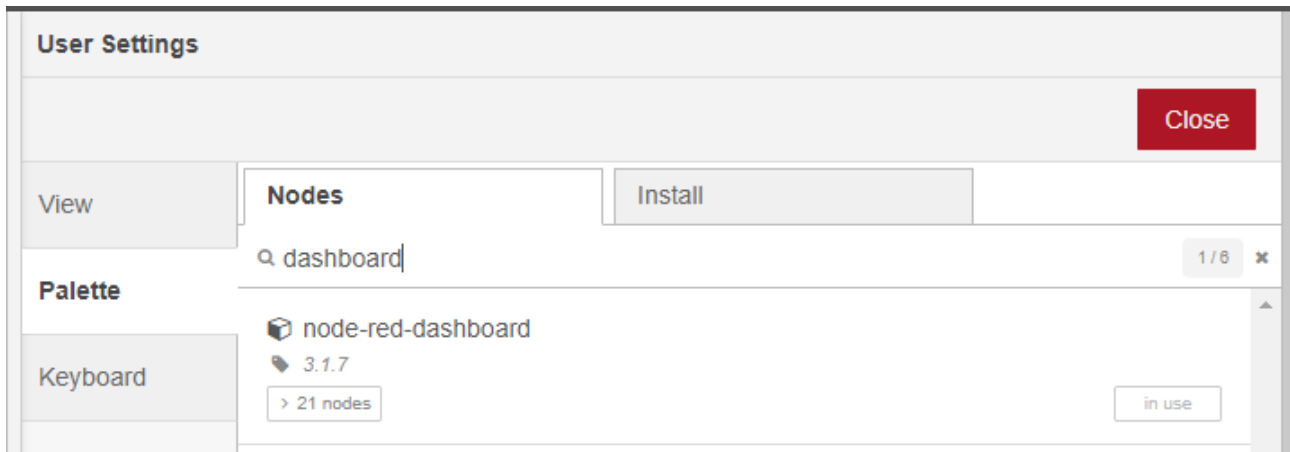


Figure 5.4 – Ajout de la catégorie Dashboard à la palette

5.3.2 L'API OpenWeatherMap

Une interface de programmation d'application (API) est un ensemble de fonctions écrites par des développeurs de logiciels pour permettre à quiconque d'utiliser leurs données ou leurs services. Le projet OpenWeatherMap dispose d'une API qui permet aux utilisateurs de demander des données météorologiques. Dans ce projet, j'utiliserai cette API pour demander les prévisions météorologiques du jour pour l'emplacement la ville de Dakar.

Le plan gratuit d'OpenWeatherMap fournit tout ce dont j'ai besoin pour mener à bien ce projet. Pour utiliser l'API, j'ai besoin d'une clé API, appelée APIID. Pour obtenir l'APIID : J'ouvre un navigateur et j'accède à <https://openweathermap.org/appid/> J'appuie sur le bouton S'inscrire et je crée un compte gratuit ensuite je vais sur ce lien : https://home.openweathermap.org/api_keys et j'obtiens ma clé API.



Clé	Nom	Statut	Actions
47df0de449bf1050cf7a8bc73a30589b	Défaut	Actif	 

Figure 5.5 – Clé api OpenWeatherMap

Dans cet exemple, nous apprendrons à effectuer des requêtes API pour accéder aux données. À titre d'exemple, nous utiliserons l'API OpenWeatherMap. Cette API a un plan gratuit et fournit de nombreuses informations utiles sur la météo dans presque tous les endroits du monde.

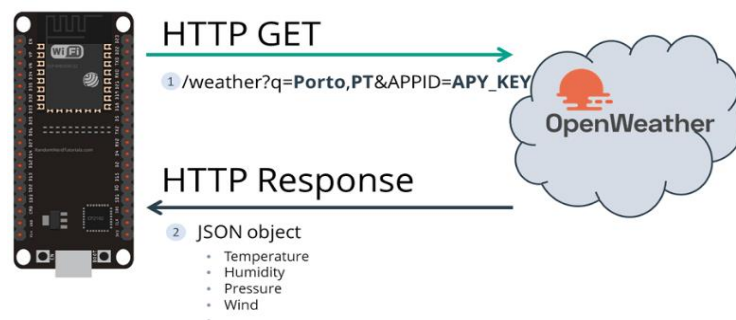


Figure 5.6 – Requête http GET

5.3.3 l'IDE Visual Studio Code et l'extension PlatformIO

Le Visual Studio Code est un éditeur de code source léger mais puissant s'exécutant sur votre bureau et est disponible sur les systèmes d'exploitation Windows, macOS et Linux. Il est livré avec un support intégré pour JavaScript, TypeScript et Node.js et dispose d'un riche écosystème d'extensions pour d'autres langages (tels que C++, C#, Java, Python, PHP, Go) et des environnements d'exécution (tels que .NET et Unity).

PlatformIO est un écosystème open source dédié au développement IoT. PlatformIO IDE est l'environnement de développement C/C++ pour les systèmes embarqués supportés. Il est multi-plateformes (Windows, Mac et GNU/Linux) et il fournit une extension à un éditeur de texte existant : soit Atom soit VSCode.

Je pourrai bien utiliser Arduino IDE mais j'ai choisi de travailler sur PlatformIO du fait de sa simplicité et de la disponibilité de l'autocomplétions.

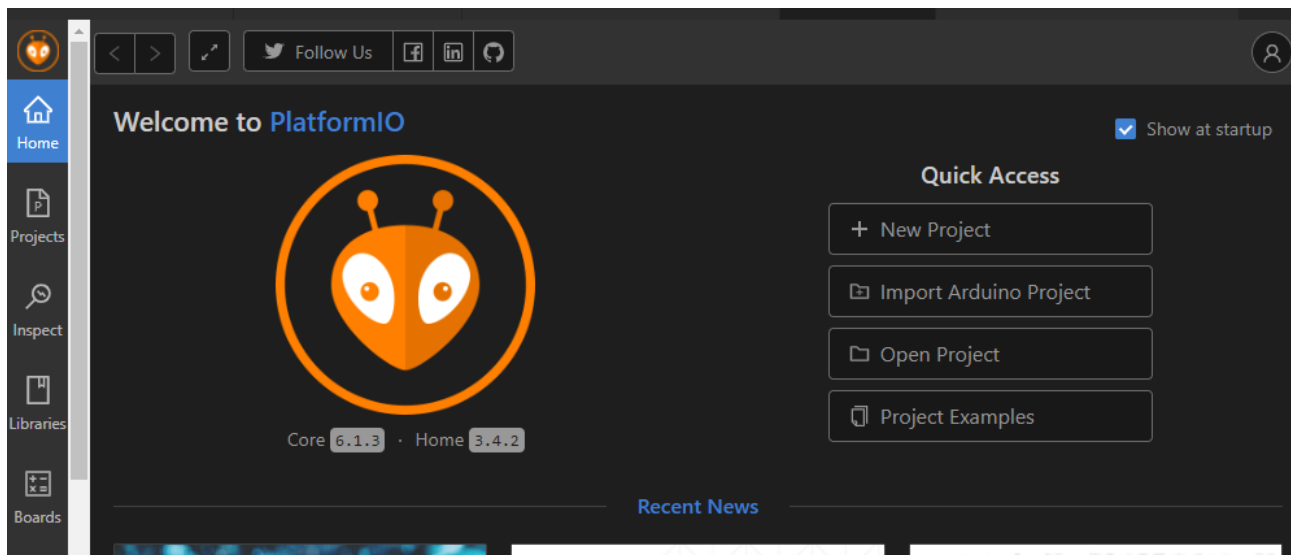


Figure 5.7 - PlatformIO

5.3.4 Machine virtuelle Ubuntu 20.04

Ubuntu 20.04 est la meilleure porte d'entrée pour découvrir, adopter puis utiliser Linux au quotidien. Développé par Canonical, Ubuntu est une distribution Linux accessible et simple d'utilisation. Ubuntu 20.04 est une version LTS (Long Term Support) dont le support technique sera assuré pendant cinq ans (jusqu'en 2025). C'est est la 8è version LTS d'Ubuntu, elle succède à la 18.04 :

Version	Nom de code	Date de sortie	Fin du support	Noyau Linux
22.04 LTS	Jammy Jellyfish	21/04/2022	04/2027	5.15
20.04 LTS	Focal Fossa	23/04/2020	04/2025	5.4
18.04 LTS	Bionic Beaver	26/04/2018	04/2023	4.15
16.04 LTS	Xenial Xerus	21/04/2016	04/2021	4.4
14.04 LTS	Trusty Tahr	17/04/2014	30/04/2019	3.13
12.04 LTS	Precise Pangolin	26/04/2012	28/04/2017	3.2
10.04 LTS	Lucid Lynx	29/04/2010	09/05/2013	2.6.32
8.04 LTS	Hardy Heron	24/04/2008	12/05/2012	2.6.24
6.06 LTS	Dapper Drake	01/06/2006	14/07/2009	2.6.15

Figure 5.8 – Versions Ubuntu

C'est dans ce serveur Ubuntu qu'on installera Node-RED en tant que module global avec ses dépendances et le broker Mosquitto pour publier ou s'inscrire à des topics avec le protocole MQTT.

5.4. Configurations tests et résultats

Dans cette partie, nous ferons un résumé des différentes configurations qui ont été fait côté client et côté serveur.

L'idée de ce projet est de réaliser un système d'arrosage automatique qui utilise comme capteur de température et de luminosité les données d'OpenWeatherMap et l'actuateur (pompe dans notre cas) sera représenté par une LED. Les données seront récupérées par l'ESP32 par la méthode HTTP GET et envoyées à node-red par le protocole MQTT. Le tableau de bord de node-red sera utilisé pour visualiser la température, la pression et l'état de la pompe.

- Dans la section 3.2.1.2, nous avons montré comment installer node-red sur le serveur Ubuntu. Pour installer le broker on tape la commande suivante :

```
#sudo apt install mosquitto
```

- je télécharge puis j'installe Microsoft Visual Studio Code officiel. PlatformIO IDE est construit dessus.

Ouvrir le gestionnaire de packages VSCode

Rechercher l'extension officielle PlatformIO ide

Installer PlatformIO IDE.

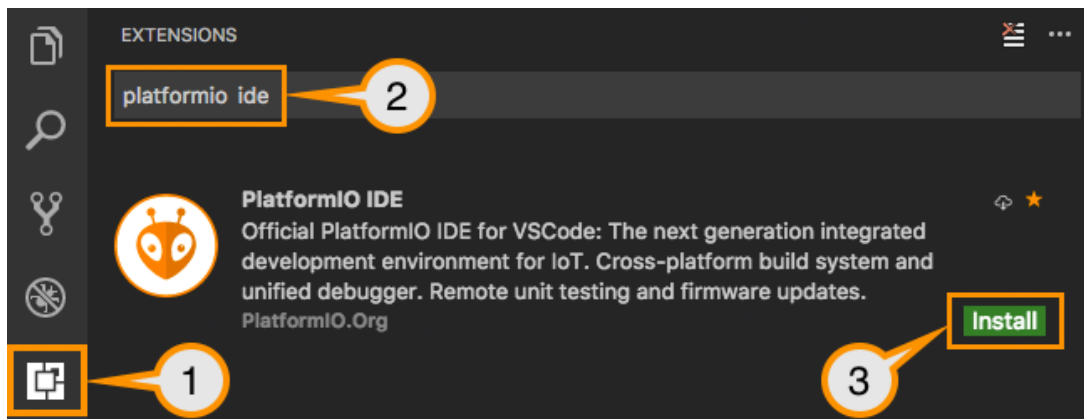


Figure 5.9 – Installation PlatformIO IDE

- Passons maintenant à l'explication du code.

Avant de poursuivre, je dois m'assurer d'avoir rempli les conditions préalables suivantes.

- **Bibliothèque Wifi**
- **Bibliothèque Arduino JSON**

Une bibliothèque JSON simple et efficace pour le C++ embarqué. ArduinoJson prend en charge la sérialisation, la désérialisation, MessagePack, l'allocation fixe, la copie zéro, les flux, le filtrage, etc.

- **Bibliothèque PubSubClient**

Une bibliothèque cliente pour la messagerie MQTT. MQTT est un protocole de messagerie léger idéal pour les petits appareils. Cette bibliothèque vous permet d'envoyer et de recevoir des messages MQTT. Il prend en charge le dernier protocole MQTT 3.1.1 et peut être configuré pour utiliser l'ancien MQTT 3.1

Pour l'installer, j'ouvre platformio.ini, un fichier de configuration de projet situé à la racine du projet PlatformIO. J'ajoute la ligne suivante : lib_deps [env:] :

knolleary/PubSubClient@^2.8

PlatformIO installera automatiquement les dépendances dans les projets

- **Bibliothèque HTTPClient**

Cette bibliothèque peut être utilisée pour les requêtes HTTP (GET, POST, PUT, DELETE) vers un serveur Web. L'installation se fait de la même façon que PubSubClient

- **Une clé API OpenWeatherMap valide**

Pour avoir une clé OpenWeatherMap il faut d'abord s'inscrire et créer un compte gratuit (voir la section 5.3.2).

```
platformio.ini http_openweither x main.cpp mqtt1 - src main.cpp http_openweither - src 1
http_openweither > platformio.ini
11 [env:esp32dev]
12 platform = espressif32
13 board = esp32dev
14 framework = arduino
15 monitor_speed = 115200
16 lib_deps =
17   mbed-kazushi2008/HTTPClient @ 0.0.0+sha.cf5d7427a9ec
18   arduino-libraries/Arduino_JSON @ ^0.1.0
19   knolleary/PubSubClient @ ^2.8
```

Figure 5.10 – Fichier de configuration PlatformIO

Explication du code :

Inclure les bibliothèques

```
#include <WiFi.h>
#include <HTTPClient.h>
#include <Arduino_JSON.h>
#include <PubSubClient.h>
```

Déclaration des constantes et des variables

```
const char* ssid = "redmi";
const char* password = "passer123";
const int ledPin = 5;
//Adresse du serveur mqtt
char *mqttServer = "192.168.231.46";
//Port du serveur mqtt
int mqttPort = 1883;
const char* mqtt_client_name = "ESPYS2111";
//Le sujet sur lequel notre client va publier
const char* mqtt_pub_topic = "/ys/testpub";
//Le sujet à laquelle notre client s'abonnera
const char* mqtt_sub_topic = "/ys/testsub/";
```

Ma clé API, le nom de la ville et code du pays

```
String openWeatherMapApiKey = "47df0de449bf1050cf7a8bc73a30589b";
String city = "Dakar";
String countryCode = "SN";
```

La minuterie par défaut est réglée sur 10 secondes.

Pour une application finale, vérifiez les limites d'appel API par heure/minute pour éviter d'être bloqué/interdit.

```
unsigned long lastTime = 0;
unsigned long timerDelay = 10000;
```

```
String jsonBuffer;
WiFiClient client;
PubSubClient mqttClient(client);
```

La fonction httpGETRequest() fait une requête à OpenWeatherMap et elle récupère une chaîne avec un objet JSON qui contient toutes les informations sur la météo de votre ville.

```
String httpGETRequest(const char* serverName) {

    HTTPClient http;
    //Nom de domaine avec le chemin de l'URL ou adresse IP avec le chemin.
    http.begin(client, serverName);
    //Envoyé une requête HTTP GET
    int httpResponseCode = http.GET();
    String payload = "{}";

    if (httpResponseCode>0) {
        Serial.print("HTTP Response code: ");
```

```

    Serial.println(httpResponseCode);
    payload = http.getString();
}
else {
    Serial.print("Error code: ");
    Serial.println(httpResponseCode);
}
// Free resources
http.end();

return payload;
}

```

Dans la fonction callback() , l'ESP32 reçoit les messages MQTT des sujets souscrits. Selon le sujet et le message MQTT

```

void callback(char* topic, byte* message, unsigned int length) {
    Serial.print("Message arrived on topic: ");
    Serial.print(topic);
    Serial.print(". Message: ");
    String messageTemp;

    for (int i = 0; i < length; i++) {
        Serial.print((char)message[i]);
        messageTemp += (char)message[i];
    }
    Serial.println();
}

```

Les lignes suivantes commencent la connexion Wi-Fi avec WiFi.begin(ssid, mot de passe), attendez une connexion réussie et imprimez l'adresse IP ESP dans le moniteur série.

```

void setup() {
    Serial.begin(115200);
    pinMode(ledPin, OUTPUT);
    WiFi.begin(ssid, password);
    Serial.println("Connecting");
    while(WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.print(".");
    }
    Serial.println("");
    Serial.print("Connected to WiFi network with IP Address: ");
    Serial.println(WiFi.localIP());

    Serial.println("Attendre 10 secondes avant de publier les premier
résultats.");
    mqttClient.setServer(mqttServer, mqttPort);
    mqttClient.setCallback(callback);
}

```

Dans la boucle(), appelons httpGETRequest() fonction pour faire la requête HTTP GET
Pour accéder aux valeurs, décodez l'objet JSON et stockez toutes les valeurs dans jsonBuffer

On doit convertir la variable flottante de température en un tableau de caractères, afin de pouvoir publier la lecture de la température dans le sujet esp32/temperature idem pour l'humiditer.

Si la température est supérieure à 298 kelvin, soit 25 degré Celsius et l'humidité inférieure à 79% on active la pompe. Si l'humidité inférieure à 50% aussi on active la pompe. Sinon la pompe est désactivée.

La boucle se répète après chaque 10 secondes.

```
void loop() {
  if ((millis() - lastTime) > timerDelay) {
    //Vérifié le statut de la connexion wifi
    if(WiFi.status()== WL_CONNECTED){
      String serverPath = "http://api.openweathermap.org/data/2.5/weather?q=" +
city + "," + countryCode + "&APPID=" + openWeatherMapApiKey;
      jsonBuffer = httpGETRequest(serverPath.c_str());

      JSONVar myObject = (JSON.parse(jsonBuffer));
      // JSON.typeof(jsonVar) est utilisé pour obtenir le type de la variable.
      if (JSON.typeof(myObject) == "undefined") {
        Serial.println("Parsing input failed!");
        return;
      }
      int temp = myObject["main"]["temp"];
      int hum = myObject["main"]["humidity"];
      if (!mqttClient.connected()){
        while (!mqttClient.connected()){
          if(mqttClient.connect(mqtt_client_name)){
            Serial.println("MQTT Connected!");
            mqttClient.subscribe(mqtt_sub_topic);
          }
          else{
            Serial.print(".");
          }
        }

        char tempString[8];
        dtostrf(temp, 1, 2, tempString);
        mqttClient.publish("esp32/temperature", tempString);
        char humString[8];
        dtostrf(hum, 1, 2, humString);
        mqttClient.publish("esp32/humidity", humString);
      }
    }

    Serial.print("JSON object = ");
  }
```

```

    Serial.println(myObject);
    Serial.print("Temperature: ");
    Serial.println(temp);
    Serial.print("Humidity: ");
    Serial.println(hum);
    if((temp > 298) && (hum <= 79)){
        digitalWrite(ledPin, HIGH);
        Serial.println("Pompe activ   ");
        mqttClient.publish("etat/pompe", "activ   ");
        if(hum <= 50){
            digitalWrite(ledPin, HIGH);
        }
    }
    else{
        digitalWrite(ledPin, LOW);
        Serial.println("Pompe desactiv   ");
        mqttClient.publish("etat/pompe", "eteinte");
    }

}
else {
    Serial.println("WiFi Disconnected");
}
lastTime = millis();
}
}

```

D  monstration HTTP GET

Apr  s avoir t  l  charg   le code, j'ouvre le Serial Monitor et vous verrez qu'il re  oit les donn  es JSON suivantes :

```

Connection.....
Connecter au WiFi avec l'adresse: 192.168.231.180
Attendre 10 secondes avant de publier les premier r  sultats.
HTTP Response code: 200
MQTT Connected!
JSON object = {"coord":{"lon":-17.4441,"lat":14.6937},"weather":[{"id":804,"main":"Clouds","description":"overcast clouds","icon":"04n"}],"base":"stations","main":{"temp":298.64,"feels_like":299.46,"temp_min":298.64,"temp_max":298.64,"pressure":1011,"humidity":85,"sea_level":1011,"grnd_level":1010},"visibility":10000,"wind":{"speed":9.44,"deg":209,"gust":10.9},"clouds":{"all":99},"dt":1658610336,"sys":{"type":1,"id":2410,"country":"SN","sunrise":1658559025,"sunset":1658605305},"timezone":0,"id":2253354,"name":"Dakar","cod":200}

```

Ensuite, il imprime l'objet JSON d  cod   dans le moniteur s  rie PlatformIO IDE pour obtenir les valeurs de temp  rature (en Kelvin), d'humidit   et l'  tat de la pompe :

```

Temperature: 298
Humidity: 85
Pompe desactiv   

```


Principe de programmation node-red

La programmation consiste à glisser-déplacer les nodes sur le flow et à les relier entre eux. Les données échangées sont des objets JSON et doivent posséder la propriété payload (charge utile). Le payload peut être un nombre, une chaîne de caractères ou un objet JSON. Certains nodes peuvent avoir plusieurs sorties. Dans ce cas, le node retourne un tableau d'objets JSON dont le premier élément sort par la première sortie, le deuxième par la seconde, ... Pour modifier la configuration par défaut des nodes, il suffit de double-cliquer dessus.

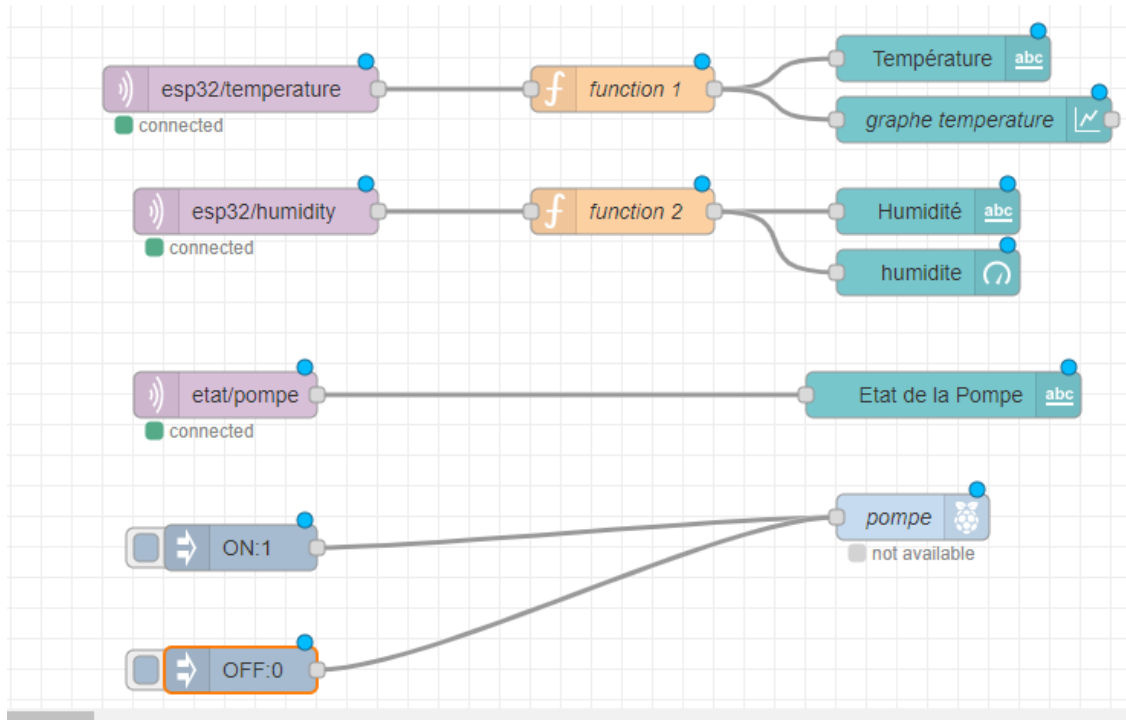
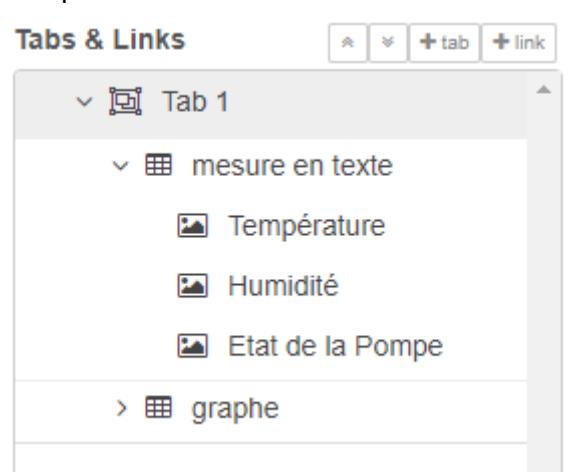


Figure 5.11 – Flow Node-red

Pour que des composants du Dashboard soient bien disposés sur la page web, il faut les mettre dans un Tab, puis dans un Groupe.



- Souscription MQTT

Pareil pour les souscriptions des autres topics.

Je clique ensuite sur le bouton Done, puis déployer le flow en cliquant sur le bouton Deploy.

On accède au site web généré par l'url : <http://IPServeur:1880/ui>

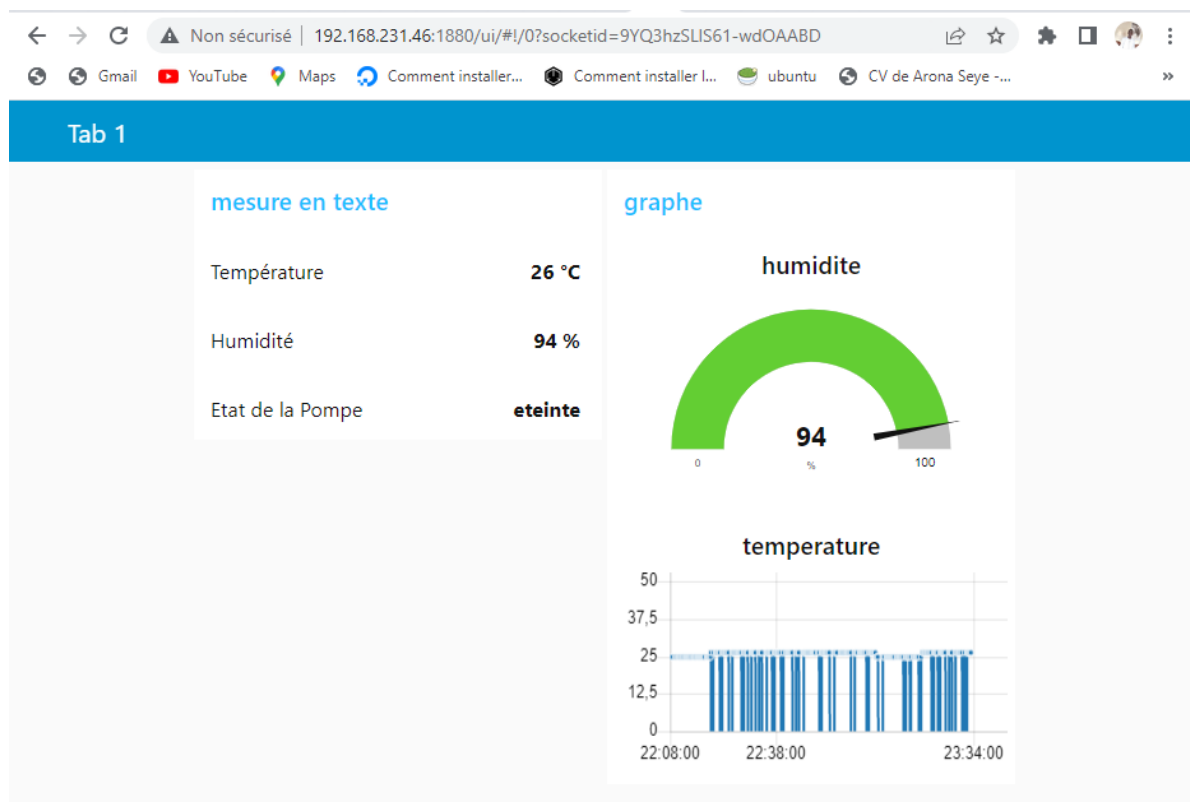


Figure 5.12 – Dashboard Node-red

On voit bien les données sous forme de texte et de graphe ainsi que l'état de la pompe (éteinte dans ce cas).

Enregistrement des données

Il est possible d'enregistrer les données dans un fichier texte ou dans une base de données. Il suffit d'utiliser le node approprié.

Base de données MySQL

Un node de gestion du classique SGBD MySQL est disponible. Commençons par installer le node MySQL :

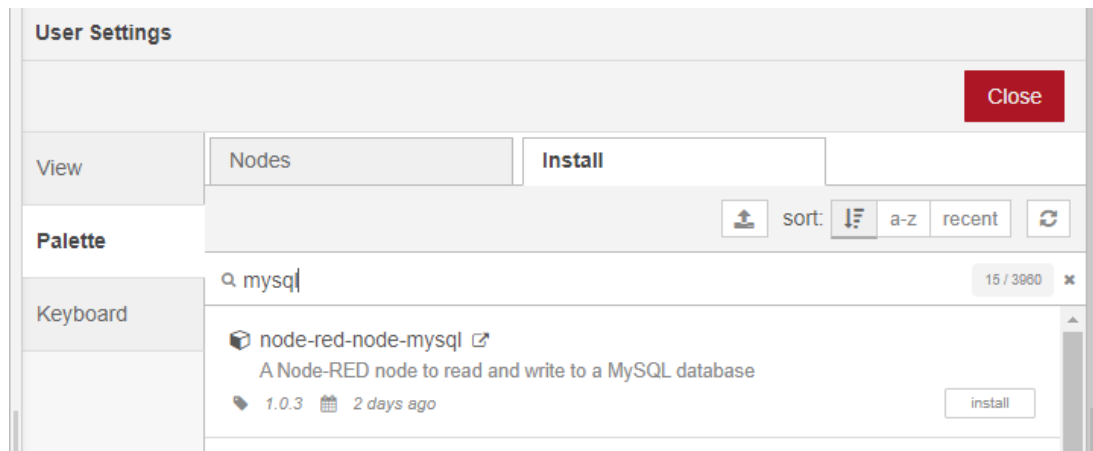


Figure 5.13 - Installation du nœud MySQL

Glissez le node MySQL sur le flow et configurez le pour se connecter à la base de données. Cela sous-entend que nous ayons MySQL installé et démarré avec une base de données accessible.

Création de la base de données et de l'utilisateur

```
mysql> create database capteur;
Query OK, 1 row affected (0,02 sec)

mysql> create user arona identified by 'passer';
Query OK, 0 rows affected (0,03 sec)

mysql> grant all privileges on capteur.* to arona;
Query OK, 0 rows affected (0,07 sec)

mysql> flush privileges;
Query OK, 0 rows affected (0,08 sec)

mysql> use capteur;
Database changed
mysql> create table openweathermap(id int primary key auto_increment, humidite
varchar(255), temperature varchar(255));
Query OK, 0 rows affected (0,06 sec)

mysql> 
```

Figure 5.14 – Configuration de la base de données

Configuration du nœud MySQL

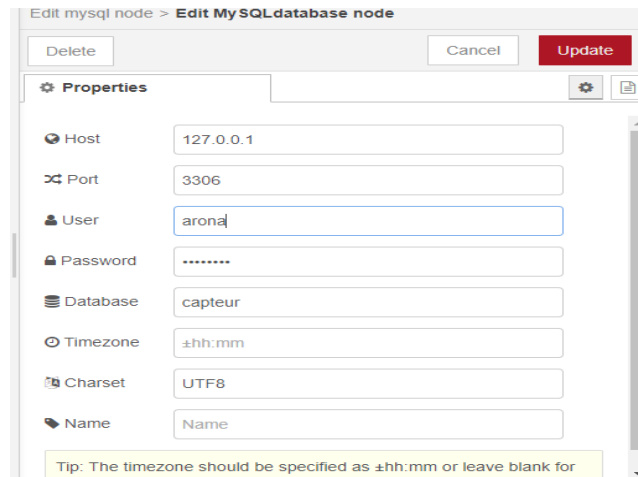


Figure 5.15 – Configuration du nœud MySQL

La requête SQL est injectée au node MySQL sous la forme d'un objet. C'est une chaîne de caractères qui constitue la propriété topic de l'objet.

Ajoutez une fonction entre les nodes Température, Humidité et le node MySQL. Saisissez le code ci-dessous qui sous-entend qu'il existe une table openweathermap et les champs humidite, temperature dans ma base de données.

Code des fonctions :

```
var query = "INSERT INTO `openweathermap` (`humidite`) VALUES  
('"+msg.payload+"'"); msg.topic = query; return msg;  
var query = "INSERT INTO `openweathermap` (`temperature`) VALUES  
('"+msg.payload+"'"); msg.topic = query; return msg;
```

Observons maintenant dans MySQL l'insertion des données :

```
mysql> select * from openweathermap;
```

id	humidite	temperature
1	94 %	NULL
2	NULL	24 °C 24/07/2022 à 15:01:13
3	94 %	NULL
4	NULL	24 °C 24/07/2022 à 15:01:23
5	NULL	24 °C 24/07/2022 à 15:01:34
6	94 %	NULL
7	94 %	NULL
8	NULL	24 °C 24/07/2022 à 15:01:45
9	94 %	NULL
10	NULL	24 °C 24/07/2022 à 15:01:55
11	94 %	NULL
12	NULL	24 °C 24/07/2022 à 15:02:06

Figure 5.16 – enregistrement des données dans MySQL

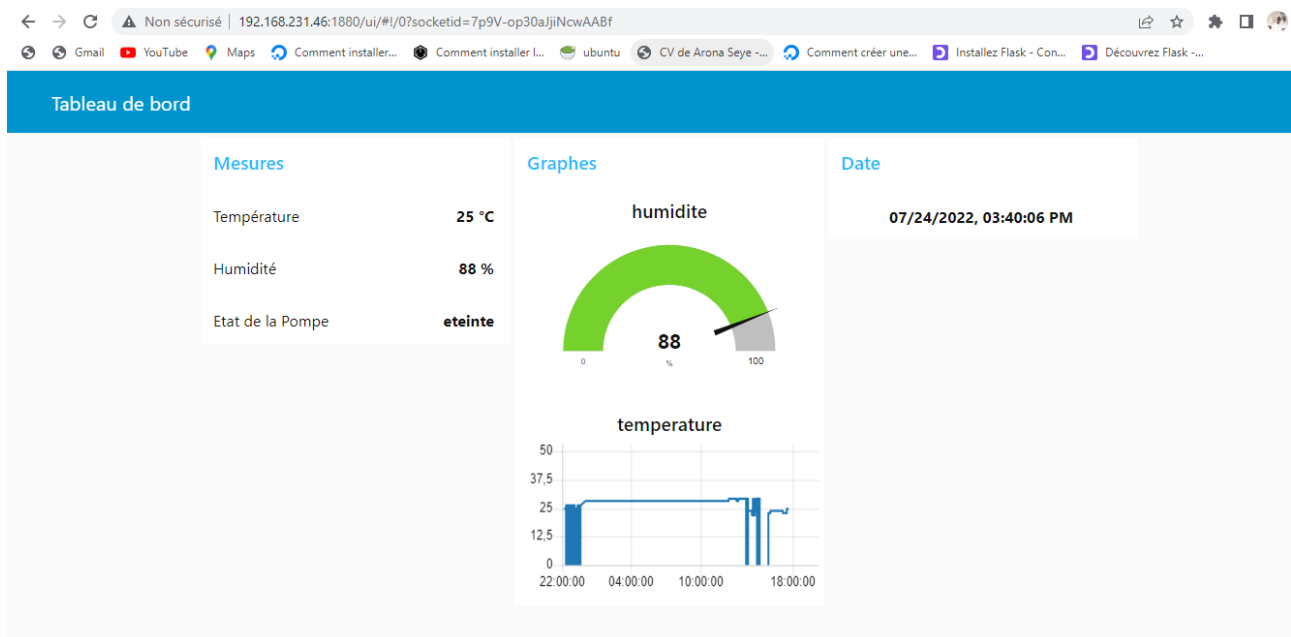


Figure 5.17 – Dashboard final

5.4.1 Exécution du programme sous docker

Pour que le mémoire se rapporte au cloud computing, j'ai décidé d'utiliser une plateforme de conteneurisation. La conteneurisation consiste à rassembler le code du logiciel et tous ses composants de manière à les isoler dans leur propre « conteneur ».

Le logiciel ou l'application dans le conteneur peut ainsi être déplacé et exécuté de façon cohérente dans tous les environnements et sur toutes les infrastructures, indépendamment de leur système d'exploitation. Le conteneur fonctionne comme une sorte de bulle, ou comme un environnement de calcul qui enveloppe l'application et l'isole de son entourage. C'est en fait un environnement de calcul portable complet.

La « légèreté » ou portabilité des conteneurs découle de leur capacité à partager le noyau du système d'exploitation de la machine hôte. Ils n'ont pas besoin d'un système d'exploitation propre et les applications peuvent s'exécuter.

Dans la suite, je vais utiliser la plateforme de conteneurisation docker.

5.4.1.1 Présentation et installation de docker

Il s'agit d'une plateforme logicielle open source permettant de créer, de déployer et de gérer des containers d'applications virtualisées sur un système d'exploitation. Les services ou fonctions de l'application et ses différentes bibliothèques, fichiers de configuration, dépendances et autres composants sont regroupés au sein du container. Chaque container exécuté partage les services du système d'exploitation.

Les containers sont donc proches des machines virtuelles, mais présentent un avantage important. Alors que la virtualisation consiste à exécuter de nombreux systèmes d'exploitation sur

un seul et même système, les containers se partagent le même noyau de système d'exploitation et isolent les processus de l'application du reste du système.

Initialement créé pour fonctionner avec la plateforme Linux, Docker fonctionne désormais avec d'autres OS tels que Microsoft Windows ou Apple MacOS. On compte également des versions de la plateforme conçues pour Amazon Web Services et Microsoft Azure.

- Fonctionnalités

La plateforme de conteneurisation repose sur sept composants principaux. Le Docker Engine est un outil client-serveur sur lequel repose la technologie de container pour prendre en charge les tâches de création d'applications basées container. Le moteur crée un processus daemon server-side permettant d'héberger les images, les containers, les réseaux et les volumes de stockage. Ce daemon fournit aussi une interface SLI client-side permettant aux utilisateurs d'interagir avec le daemon via l'API de la plateforme.

Les containers créés sont appelés Dockerfiles. Le composant Docker Compose permet de définir la composition des composants au sein d'un container dédié. Le Docker Hub est un outil SaaS permettant aux utilisateurs de publier et de partager des applications basées container via une bibliothèque commune.

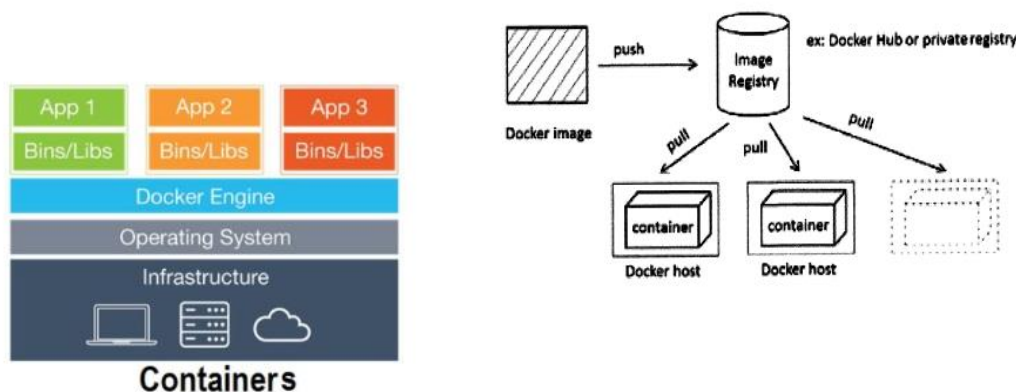


Figure 5.18 - Architecture de base de docker

- Installation

Tout d'abord, mettons à jour notre liste existante de packages :

```
$ sudo apt update
```

Ensuite, installons quelques packages prérequis qui permettent à apt d'utiliser des packages via HTTPS :

```
$ sudo apt install apt-transport-https ca-certificates curl  
software-properties-common
```

Ajoutons ensuite la clé GPG du référentiel Docker officiel à votre système :

```
$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo  
apt-key add -
```

Ajoutons le dépôt Docker aux sources APT :

```
$ sudo add-apt-repository "deb [arch=amd64]  
https://download.docker.com/linux/ubuntu focal stable"
```

Cela mettra également à jour notre base de données de packages avec les packages Docker du dépôt nouvellement ajouté. Assurons-nous que nous sommes sur le point d'installer à partir du référentiel Docker au lieu du référentiel Ubuntu par défaut :

```
$ apt-cache policy docker-ce
```

Enfin, installons Docker :

```
$ sudo apt install docker-ce
```

5.4.1.2 Mise en place du conteneur

Les conteneurs Docker sont construits à partir d'images Docker. Par défaut, Docker extrait ces images de Docker Hub, un registre Docker géré par Docker, la société à l'origine du projet Docker. N'importe qui peut héberger ses images Docker sur Docker Hub, donc la plupart des applications et des distributions Linux dont vous aurez besoin auront des images hébergées là-bas.

Les images dont j'ai besoin pour construire le conteneur sont : eclipse-mosquitto pour le broker MQTT, mysql pour la base de données et nodered/node-red.

Une fois que nous avons identifié les images que nous souhaitons utiliser, nous pouvons les télécharger sur notre ordinateur à l'aide de la sous-commande pull.

Exécutez les commandes suivantes pour télécharger les images officielles sur notre ordinateur :

```
#docker pull eclipse-mosquitto #docker pull mysql #docker pull nodered/node-red
```

Pour vérifier si les images sont bien téléchargées, on tape la commande :

```
#docker images
```

```
root@arona:~# docker images
REPOSITORY          TAG             IMAGE ID        CREATED         SIZE
nodered/node-red    latest         5d049d395dd3   4 days ago     474MB
eclipse-mosquitto   latest         39f30088c30d   6 days ago     11.8MB
mysql               latest         33037edcac9b   13 days ago    444MB
root@arona:~#
```

Nous pouvons exécuter les images sur notre ordinateur à l'aide de la sous-commande run.

```
#docker run --name broker -p 18:1883 eclipse-mosquitto
```

L'option `--name` permet de donner un nom au conteneur, `-p` définit le port d'écoute.

```
#docker run --name mysql -e MYSQL_ROOT_PASSWORD=passer mysql
```

passer est le mot de passe à définir pour l'utilisateur root de MySQL

Après avoir démarré ces deux conteneurs, j'ai maintenant besoin de les lier dans un même conteneur avec node-red. Pour cela je mets la commande :

```
#docker run --name nodered -v nodereddata:/data --link broker:eclipse-mosquitto --link mysql:mysql -p 188:1880 nodered/node-red
```

Cette commande crée un conteneur contenant des instances de mosquitto, mysql et de node-red.

```

root@arona:~# docker ps
CONTAINER ID   IMAGE                  COMMAND                  CREATED        STATUS        PORTS
c061b06a5943   nodered/node-red       "/entrypoint.sh"        4 hours ago   Up 4          0.0.0.0:188->1880/tcp, :::188->1880/tcp
4552d2bbe3fa   eclipse-mosquitto      "/docker-entrypoint...." 4 hours ago   Up 4          0.0.0.0:18->1883/tcp, :::18->1883/tcp
d0bb1e8f5867   mysql                  "docker-entrypoint.s..." 9 hours ago   Up 9          3306/tcp, 33060/tcp
root@arona:~#

```

Pour accéder aux services du conteneur je tape sur la barre de recherche du navigateur :

<http://adresseServeur:188>

Les configurations faites sur la base de données sont les mêmes que j'ai fait dans la partie figure 5.10. Donc ce qui me reste à faire c'est seulement copier le flow sur cette nouvelle page node-red et y installer les palettes nécessaires.

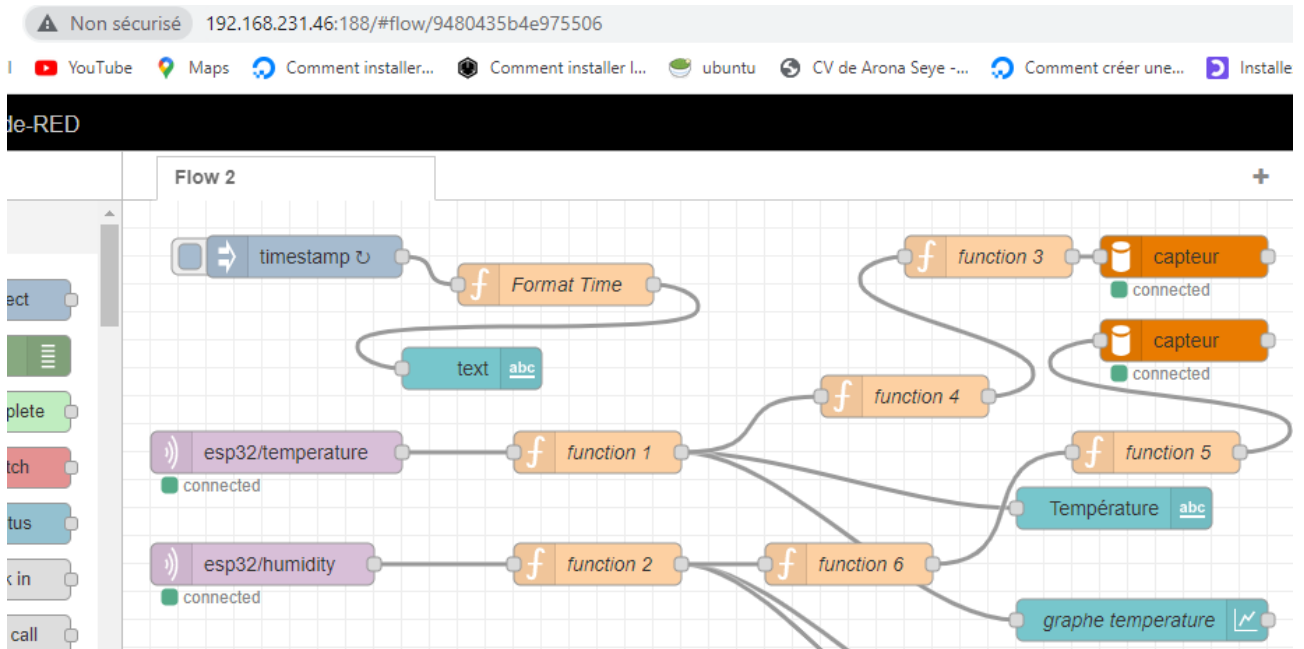


Figure 5.19 – Flow Node-red docker

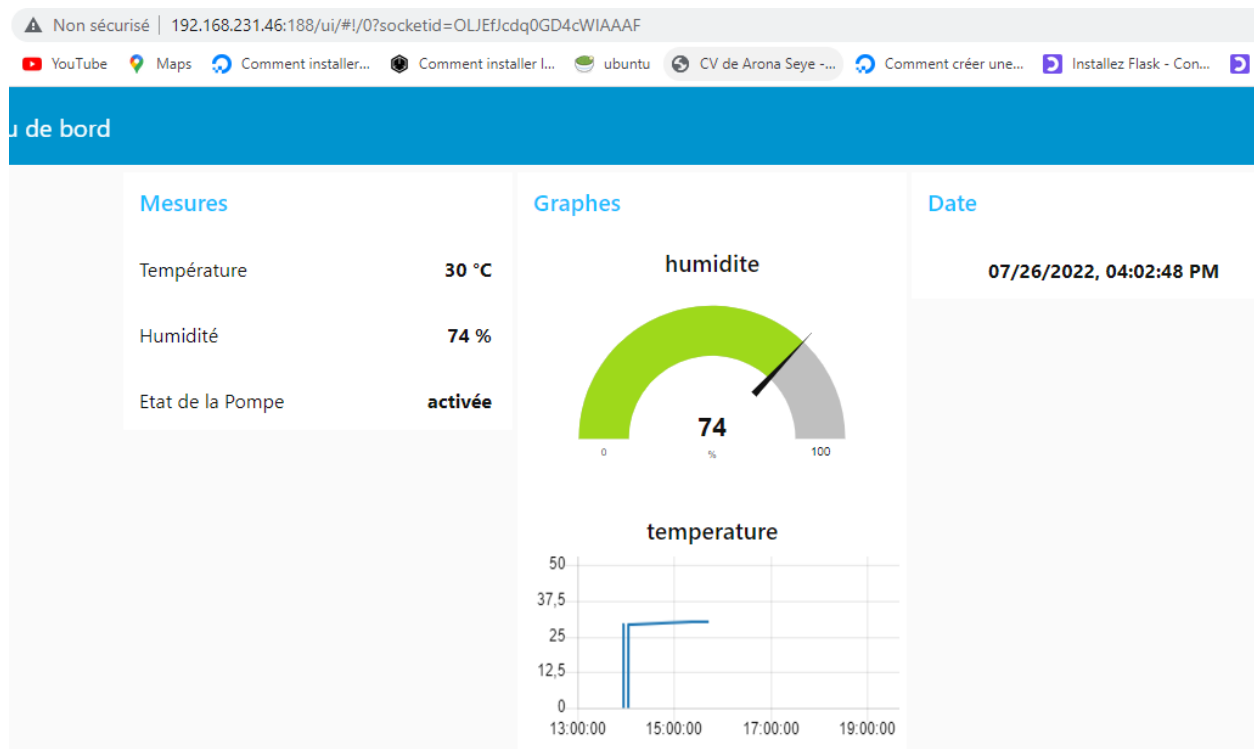


Figure 5.20 – Dashboard Node-red docker

Ainsi j'ai tout mon système sur un container docker.

5.4.1.3 Push du conteneur vers le référentiel docker

La prochaine étape logique après la création d'une nouvelle image à partir d'une image existante consiste à la partager avec quelques-uns de mes amis, le monde entier sur Docker Hub ou un autre registre Docker auquel nous avons accès. Pour pousser une image vers Docker Hub ou tout autre registre Docker, je dois y avoir un compte.

Après avoir mis en place le conteneur, j'ai maintenant un conteneur fonctionnant à partir d'une image, mais le conteneur est différent de l'image que j'ai utilisée pour le créer. Mais je voudrai peut-être réutiliser ce conteneur comme base pour de nouvelles images plus tard. Validons les modifications dans une nouvelle instance d'image Docker à l'aide de la commande suivante.

```
#docker commit -m "Node-red, MySQL, MQTT" -a "aron" c061b06a5943
aron/nodered
```

Pour envoyer mon image, je me connecte d'abord à Docker Hub.

```
root@aron:~# docker login
Authenticating with existing credentials...
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
root@aron:~#
```

Figure 5.21 - Docker login

Puisque nom d'utilisateur de registre Docker est différent du nom d'utilisateur local que j'ai utilisé pour créer l'image, je dois baliser mon image avec mon nom d'utilisateur de registre en tapant :

```
#docker tag arona/nodered aronaseye/nodered
```

Ensuite, je peux pousser ma propre image en utilisant :

```
#docker push aronaseye/nodered
```

Le processus peut prendre un certain temps car il télécharge les images, mais une fois terminé, la sortie ressemblera à ceci :

```
root@arona:~# docker push aronaseye/nodered
Using default tag: latest
The push refers to repository [docker.io/aronaseye/nodered]
0649f7d776be: Pushed
57730b95baa5: Mounted from nodered/node-red
39a4c7ff1cca: Mounted from nodered/node-red
aae7f38b7af3: Mounted from nodered/node-red
5b71dc6b5235: Mounted from nodered/node-red
1bffe8b0d14a: Mounted from nodered/node-red
6dac605a6e52: Mounted from nodered/node-red
7b96954e90ca: Mounted from nodered/node-red
86019687002e: Mounted from nodered/node-red
5f70bf18a086: Mounted from nodered/node-red
9235aeff5c43: Mounted from nodered/node-red
1ef0848ec719: Mounted from nodered/node-red
0250c895df6f: Mounted from nodered/node-red
cfe6e35f03d6: Mounted from nodered/node-red
9410c6971c57: Mounted from nodered/node-red
3697d1909d84: Mounted from nodered/node-red
ec34fcc1d526: Mounted from nodered/node-red
latest: digest: sha256:68441ab5a2cccf7a636aae939bd755a5efb1efcc06a68ce867a9c894f2a35ede size: 3866
root@arona:~#
```

Figure 5.22- docker push

Après avoir poussé une image vers un registre, elle devrait être répertoriée sur le tableau de bord de mon compte, comme cela apparaît dans l'image ci-dessous :

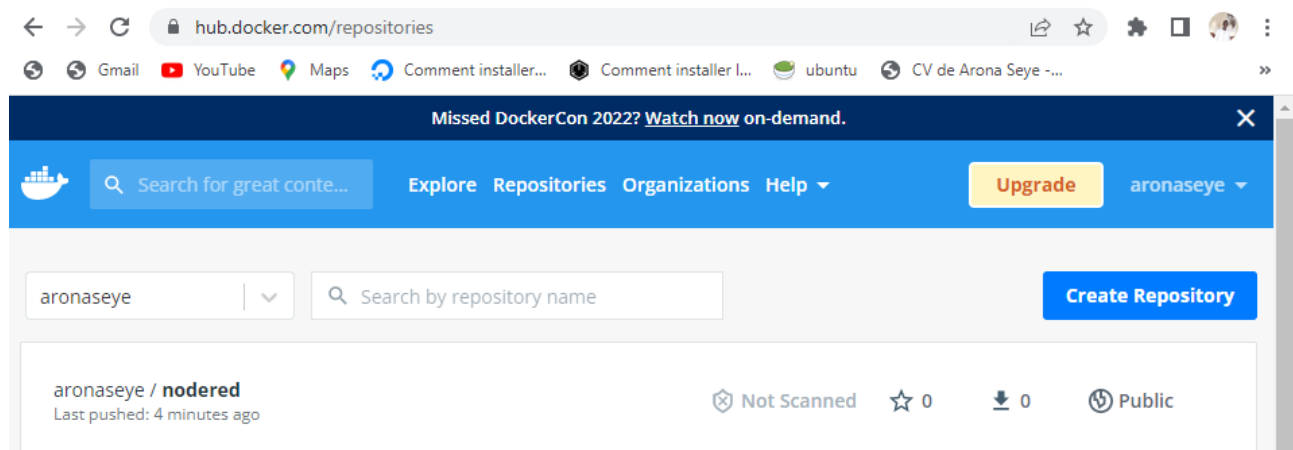


Figure 5.23 - Docker hub

N'importe qui peut maintenant utiliser pour extraire l'image vers une nouvelle machine et l'utiliser pour exécuter un nouveau conteneur :

```
#docker pull aronaseye/nodered
```

5.5. Conclusion

Ce chapitre a été consacré à l'implémentation de notre solution IoT. Au début nous avons présenté l'architecture du système et les outils et plateformes utilisés.

Nous avons ensuite fait les configurations nécessaires sur ces outils et plateformes pour qu'ils fonctionnent selon nos besoins.

Enfin, nous avons terminé ce chapitre par l'implémentation du système sur docker.

Conclusion générale et perspectives

Conclusion

La nouvelle informatique se caractérise par des nouveaux domaines qui représentent la tendance de la technique comme le cloud computing et l'internet des objets et le big data prenant une grande partie de cette nouvelle informatique. Le Big Data représente un changement significatif dans les technologies de l'information. Techniquement, nous vivons un véritable phénomène de rupture. En effet, au-delà de quelques dizaines de téraoctets, les technologies traditionnelles sont inadéquates, elles ne permettent plus d'analyser la volumétrie élevée des données, et la question qui se pose alors est de savoir comment et où se traiter cette quantité de donnée, au niveau de l'objet ou cloud, chacune a des avantages et des limites.

Dans ce mémoire, nous avons décrit en détail nos contributions :

Notre première contribution consiste à donner un aperçu clair des différentes technologies qui composent le cloud computing et l'internet des objets. Nous les avons définies, donné leurs caractéristiques, leur architecture ...

Dans la deuxième contribution, nous avons étudié quelques plateformes intégrant du cloud computing et l'internet des objets en donnant leurs avantages, leurs fonctionnalités, l'architecture et les intégrations API.

Pour la dernière contribution, nous avons, dans une première partie, mis en place une solution IoT autonome qui récupère des données météorologiques sur l'API OpenWeatherMap par la méthode GET du protocole http et qui les envoie à l'interface de programmation Node-red par le protocole MQTT. Dans la seconde partie, nous avons mis dans un conteneur l'ensemble des outils et plateformes utilisés et l'hébergé dans le registre docker hub pour que ce soit accessible et utilisable par les personnes souhaitées.

Toutes ces contributions ont été implémentées sur un serveur linux et un microcontrôleur ESP32.

Perspectives

Dans le futur et comme perspectives de ce travail, nous considérerons les extensions suivantes :

- faire un couplage avec le modèle de Big-Data ;
- développer une stratégie basée sur nos contributions pour la sécurité de l'agriculture intelligente.

Références

Hogan, Brian. How to Install and Use Docker on Ubuntu 20.04. Publié le 19 mai 2020 · Mis à jour le 29 septembre 2021 dans Digitalocean.

Dr. Majd Sakr, l'université Carnegie Mellon. Fondements du cloud computing pour les développeurs. Microsoft Learn.

Dr. Majd Sakr, l'université Carnegie-Mellon. Share cloud ressources. Microsoft Learn.

Dr. Majd Sakr, l'université Carnegie-Mellon. Understand virtualization. Microsoft Learn.

Université d'Oxford – Ajit Jaokar Artificial Intelligence : Cloud and Edge Implementations. Introduction to Azure IoT Hub.

Erin, Glass. A General Introduction to Cloud Computing. Publié le 15 octobre 2020 · Mis à jour le 2 novembre 2020 dans Digitalocean.

Meftah ZOUAI. Une approche cloud computing basée IoT pour le smart House, mémoire master 2 « Faculté des Sciences Exactes et des Sciences de la Nature et de la Vie Département d'informatique ». Université Mohamed Khider – BISKRA, 2019-2020.

Bourhim. La communication et le placement de conteneurs docker dans le fog computing. Mémoire de maîtrise, « Administration Réseaux » Université du Québec à Montréal, 2019-2020.

K. Maioua et A. Mansouri. Approche basée agents mobiles intelligents dans un environnement de cloud computing, mémoire master 2 " intelligence artificielle". Université Kasdi Mrabah Ourgla, 2014.

Julio Arauz and Tony Fynn-Cudjoe. Actuator quality in the internet of things. Dans 2013 IEEE International Conference on Sensing, Communications and Networking (SECON), 2013.

le cloud computing une nouvelle filière fortement structurante. www.idf.directe.gouv.fr, Mis en ligne 2013, consulté Avril 2017.

Akshayan Sinha , Commun9899 , Akarsh Agarwal. WebServer on ESP32. Published March 28, 2022. Dans hackster.io.

Ic_lab. Système d'arrosage automatique pour mes plantes. Publié le 24 novembre 2020. Dans hackster.io.

<https://nodered.org/docs/>

<https://openweathermap.org/guide>

<https://docs.docker.com/>

<https://randomnerdtutorials.com/esp32-http-get-open-weather-map-thingspeak-arduino/>

<https://randomnerdtutorials.com/esp32-mqtt-publish-subscribe-arduino-ide/>