**RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE**

**Ministère de l'Enseignement Supérieur et de la Recherche Scientifique**

---


**UNIVERSITÉ ECHAHID HAMMA LAKHDAR EL OUED**
**FACULTÉ DES SCIENCES EXACTES**
**Département D'informatique**


**Mémoire de Fin D'étude**
**Présenté pour l'obtention de Diplôme de**

# MASTER ACADEMIQUE

Domaine : **Mathématique et Informatique**
Filière : **Informatique**
Spécialité : **Systèmes Distribués et Intelligence Artificielle**


Présenté par :

- **BALI Rahma**
- **HANI Hala**


## *Thème*

# Une approche d'annotation sémantique et légère pour minimiser la taille de données dans une environnement IoT

Soutenue le 04/06/ 2018, Devant le jury :

| | | |
|---|---|---|
| **M. LAOUID Abdelkader** | **MAA** | **Président** |
| **M. MEDILEH Saci** | **MAA** | **Rapporteur** |
| **M. OTHMANI Samir** | **MAB** | **Encadreur** |
| **M. BALI Mouadh** | **MAB** | **Encadreur** |

**Année Universitaire : 2017-2018**

# Remerciements

Avant tous, nous remercions Dieu le tout puissant de nous avoir donné le courage et la patience pour réaliser ce travail malgré toutes les difficultés rencontrées.

Nous remercions infiniment nos encadreurs *M. OTHMANI SAMIR* et *M. BALI MOUADH* qui nous ont aidé à dépasser tous les obstacles et qui n'ont pas cessé de nous donner les conseils et les bonnes orientations. Un grand Merci pour votre temps consacré à nous.

Nous remercions *M. BALI AHMED* qui nous a aidé beaucoup pendant toute la période de préparation de ce travail malgré ses occupations. En fin, nous remercions tous ceux qui nous ont aidés de prés et de loin pour la réalisation de notre mémoire.

## Résume

Les applications de l'Internet des objets (IdO) reposent sur un réseau de dispositifs hétérogènes et homogène comprenant des capteurs et des passerelles. Ces appareils sont dotés de la capacité de détecter en continu l'environnement et de collecter des données, qui peuvent ensuite être transférées via des passerelles vers le cloud. Les données générées par les systèmes IdO sont souvent massives et hétérogènes, par conséquent, les passerelles sont les plus concernées à cause de leur limite de ressources. Ainsi, la quantité de données générées augmente également le coût du stockage et traitement des données au niveau de cloud. Par ailleurs, l'hétérogénéité des données entraîne un manque d'interopérabilité entre les applications IdO. Pour répondre à ces besoins et contraintes, nous proposons, dans ce mémoire, Une approche d'annotation sémantique et légère pour minimiser la taille de données dans un environnement IoT. Nous utilisons la notion des règles pour filtrer les données afin de minimiser la taille des données à annoter et aussi minimiser les données transférées vers le cloud, et aussi nous classons les données en des événements. De plus, nous proposons une ontologie de domaine et ontologie des règles pour représenter les connaissances au niveau sémantique. Notre approche a été l'implémente dans un cas d'étude (Feux de Foret ) ce que prouver l'utilité de notre proposition.



**Mots-clés:** Internet des objets, Ontologie des règles, Ontologie de domaine, Événement, Annotation sémantiques.



## Abstract

Internet of Things (IoT) applications rely to the network of heterogeneous devices that includes sensors and gateways. These devices have the ability to detect the environment and collect data continuously, which can be transferred via gateways to the cloud. Data generated by IoT systems is often massive and heterogeneous. As a consequenc , gateways are the most concerned because of their resource limitations. Thus, the amount of data generated also increases the cost associated with storing and processing data at the cloud level. In addition, data heterogeneity leads to a lack of interoperability between IoT applications. To answer to these needs and constraints, we propose in this thesis, a semantic and lightweight annotation approach to minimize data size in an IoT environment. We use the notion of rules to filter the data to minimize the size of the data to annotate and minimize the data transferred to the cloud side, and we classify the data into events. In addition, we propose an ontology domain and ontology rules to represent knowledge at the semantic level. Our approach has been the implementation in a case study (Forest Fire) what to prove the utility of our proposal.



**Keywords**: Internet of Things, ontology of rules, domain ontology, Event, annotation semantics.



## ملـخص

تعتمد تطبيقات إنترنت الأشياء (IoT) على شبكة من الأجهزة غير المتجانسة التي تشمل أجهزة الاستشعار والبوابات. هذه الأجهزة لديها القدرة على الكشف المستمر عن البيئة وجمع البيانات، والتي يمكن بعد ذلك نقلها عبر البوابات إلى السحابة الالكترونية. غالبًا ما تكون البيانات الناتجة عن أنظمة إنترنت الأشياء ضخمة ومتغايرة، لذا فإن البوابات هي الأكثر اهتمامًا بسبب محدودية مواردها. وبالتالي، فإن كمية البيانات المولدة تزيد أيضًا من التكلفة المرتبطة بتخزين البيانات ومعالجتها على مستوى السحابة الالكترونية. بالإضافة إلى ذلك، يؤدي عدم تجانس البيانات إلى عدم قابلية التشغيل البيني بين تطبيقات إنترنت الأشياء. للإجابة على هذه الاحتياجات والقيود نقترح في هذه المذكرة، نهجًا توضيحيًا وثريًا لبيانات إنترنت الأشياء. نستخدم مفهوم القواعد لتصفية البيانات لتقليل حجم البيانات للتعليق، وكذلك تقليل البيانات المنقولة إلى جانب السحابة الالكترونية، ونقوم أيضًا بتصنيف البيانات إلى أحداث. بالإضافة إلى ذلك، قواعد الأنطولوجيا لتمثيل المعرفة على المستوى الدلالي.



**الكلمات المفتاحية:** إنترنت الأشياء، قواعد الأنطولوجيا، علم الأنطولوجيا، الحدث، الشرح الدلالي.

# Table des matières

# Table des figures

# Liste des tableaux

# LISTE DES ABRÉVIATIONS

| | |
|---|---|
| AWS | Amazon Web Services |
| FWI | Forest Fire Weather Index |
| GPIO | General-purpose input/output |
| HTTP | Hypertext Transfer Protocol |
| IAM | Identity and Access Management |
| IoT | Internet of Things |
| M2M | Machine to Mchine |
| MQTT | Message Queuing Telemetry Transport |
| OWL | Web Ontology Language |
| RDF | Resource Description Framework |
| RDFs | Resource Description Framework schema |
| RFID | Radio-frequency Identification |
| SW | Semantic Web |
| TLS | Transport Layer Security |
| WoT | Web of Things |

# INTRODUCTION GÉNÉRALE

## Contexte

Au cours des dernières années, l'IoT a évolué à une vitesse exceptionnelle, connectant un nombre important d'objets hétérogènes (capteurs, actionneurs, smartphones, applications, etc.).

Généralement, les données générées par les systèmes IoT sont massives et hétérogènes. Par conséquent, la tâche des applications IoT pour une interprétation et utilisation efficace de ces données devient de plus en plus complexe. Pour tirer parti de ces données, il est essentiel de les transmettre à des connaissances de haut niveau, ce qui favorise le développement d'applications IoT interopérables et intelligentes[Eva15].

En outre, dans ce mémoire, les technologies Web sémantiques seront utilisées pour résoudre les problèmes liés aux services IoT et à la découverte de données.

## Problématique

Le nombre croissant d'objets connectés et la diversité leur utilisation ont créé des défis liés à l'interopérabilité, à savoir :

- **Hétérogénéité des données :**

  L'hétérogénéité des données(différents sources et formats) entraîne un manque d'interopérabilité entre les applications IoT.

- **Passerelle à des ressources limitées :**

  Les passerelles sont les plus concernées à cause de leur limite de ressources. Ainsi, la quantité de données générées augmente également le coût du stockage et traitement des données au niveau de Cloud. Aussi la limitation de la bande passante des réseaux de communication des passerelles (Plusieurs client on besoin des informations).

## Objectives et contributions

L'objective essentielle de ce mémoire étant de proposer une approche d'annotation légère qui permet de minimiser la taille de données annotées. Par cette annotation, notre approche permet à :

- Supporter le traitement de données hétérogènes en utilisant les techniques du Web sémantique.
- Supporter la réutilisation et le partage des connaissances entre les différentes applications IoT.
- Réduire le coût de traitement et storage des données au niveau du cloud.

## Organisation du mémoire

Aprés l'introduction générale, ce mémoire contient quatre chapitres et une conclusion générale.

Le chapitre 1 est consacré aux concepts de base d'Internet des Objets, Web of Things (WoT), cloud des objets, et les domaines d'application d'IoT.

Dans le chapitre 2, nous présentons les concepts de base du Web sémantique qui seront utilisées tout au long de ce mémoire et les sujets relatifs impliquant les métadonnées, les ontologies et les langages pour représenter les ontologies, les annotations sémantiques, ainsi que les domaines d'utilisation d'ontologies.

Dans le chapitre 3, nous présentons l'architecture de notre approche de l'annotation sémantique et légère pour les données IoT, en décrivant, en détail, ses différents constituants fonctionnels. Un cas d'étude est également présenté pour apporter plus d'illustration à notre approche.

Dans le chapitre 4, nous décrivons une implémentation de notre approche dédiée au cas d'étude proposé.

Dans la conclusion générale, nous discutons les résultats de notre approche et proposons des perspectives pour des futurs travaux de recherche.

# CHAPITRE I

# INTERNET DES OBJETS (IDO)

# I.1 Introduction

L'Internet des objets est une nouvelle vague de l'Internet, est en réalité une partie naissante de l'Internet du futur, appelé IoT, qui vise à interconnecter les gens, les données et tous les objets, de telle sorte qu'il y ait une fusion entre le monde réel (physique) et le monde numérique (virtuel), les objets du monde physique vont être incorporés dans le monde virtuel de l'Internet [SAH16].

Ce chapitre est consacré à donner un survol des généralités sur l'Internet des objets, web des objets, cloud des objets en finissant par la présentation des avantages et domaines d'application de l'Internet des objets.

# I.2 Définition de l'IoT

En anglais l'Internet of Things (IoT) est « un réseau qui relie et combine les objets avec l'Internet, en suivant les protocoles qui assurent leurs communication et échange d'informations à travers une variété de dispositifs[HZ13].»

L'IoT peut être définition comme étant « un réseau de réseaux qui permet, via des systèmes d'identification électroniques normalisés et unifiés, et des dispositifs mobiles sans fil, d'identifier directement et sans ambiguïté des entités numériques et des objets physiques et ainsi, de pouvoir récupérer, stocker, transférer et traiter les données sans discontinuité entre les mondes physiques et virtuels[Pie08].»

## I.2.1 Objet connecté

Un objet connecté est un objet électronique relié à Internet et capable de communiquer des informations, apportant ainsi un service ou une valeur ajoutée. Le premier objet connecte était la lampe DAL (figure I.1), lancé en 2003 par Rafi Haladjan. Sensible au toucher et au bruit, cette lampe communiquait des informations sur la météo, la bourse, la pollution, des alertes Google et même des messages grâce à neuf LED de couleur.Les fonctions proposées aujourd'hui vont beaucoup plus loin que la simple annonce de la météo[ACH17].

FIGURE I.1 – La lampe DAL (premier objet connecté).

Il y a deux types d'objets[Bil15] :

**- Les objets passifs :**ils utilisent généralement un tag (pus Radio Frequency Identification (RFID),code barre 2D).Ils embaraquent une faible capacité de stockage (de l'ordre du kilo octet) leur permettant d'assurer un rôle d'identification. ils peuvent parfois,dans le cas d'une pus RFID,d'embarquer un capteur(température, humidité).

**- Les objets actifs :** ils peuvent être équipés de plusieurs capteurs, d'une plus grande capacité de traitement ou encore être en mesure de communiquer sur un réseau.

### I.2.2 Capteurs

Les capteurs sont de petits appareils disposant de capacités de mesures, voire d'actions, sur leur environnement La température, le taux d'humidité, la luminosité ambiante, la détection de présences ou de mouvements via un accéléromètre, présence de gaz, de polluants ou encore la géolocalisation font partie des informations les plus couramment collectées sur ce type de matériel[Ele15].

Selon les capteurs, ou grâce à l'ajout de cartes additionnelles, la pression atmosphérique, le niveau de radiation ou la pression acoustique (événements sonores) peuvent aussi être quantifiés. Les spécificités innovantes de ces capteurs résident dans leur taille et leur coût réduit, tout en étant dotés des capacités de traitements de l'information, et des possibilités de transmission sans fil[Che13].

### I.2.3 Architecture de l'IoT

L'IoT en tant que domaine de recherche est encore en cours de maturation ; par conséquent, une architecture IoT unique unifiée et communément acceptée n'a pas encore été projetée. En outre, plusieurs facteurs influencent le processus de conception de l'architecture IoT, tels que l'évolutivité, l'interopérabilité, la fiabilité du stockage des données et la qualité de service. En conséquence, plusieurs propositions dans la littérature ont présenté différentes vues architecturales de l'IoT. Alors que certains d'entre eux relaient sur l'architecture (figure I.2) de base à trois couches (Perception, réseau, et couches d'application)[ALO16].

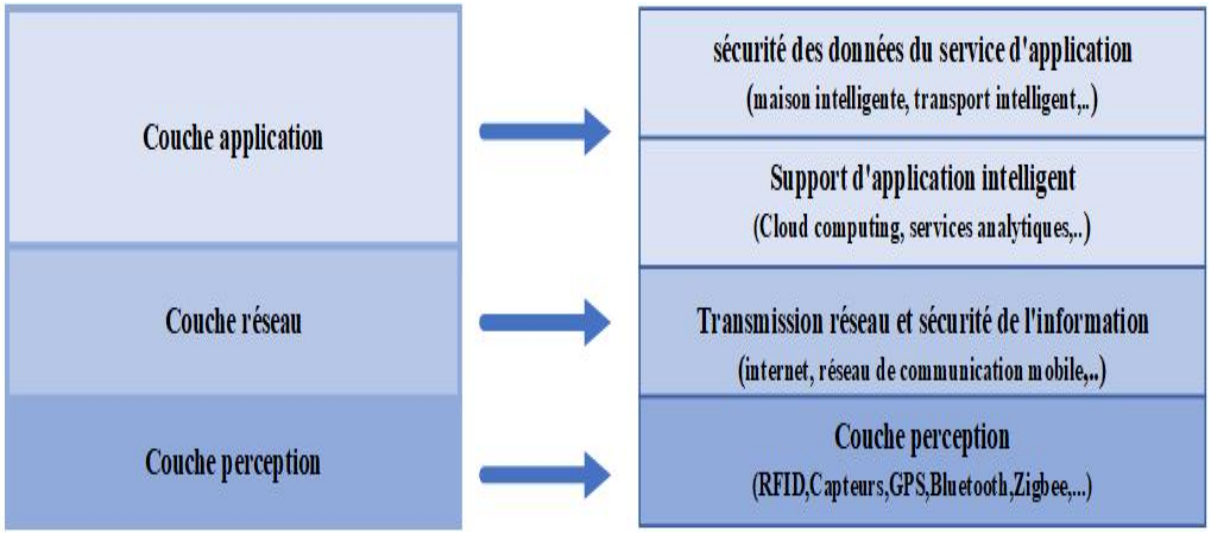


FIGURE I.2 – Architecture de l'Internet des objets

Nous en discutons brièvement dans les sous-sections suivantes :[ALO16]

**- Couche de perception**

La première couche, également connue sous le nom de couche de dispositif, est destinée à représenter les capteurs physiques de l'IoT qui visent à collecter et traiter des informations. Cette couche comprend des capteurs et des actionneurs pour effectuer différentes fonctions telles que l'interrogation de l'emplacement, la température, le poids, le mouvement, la vibration, l'accélération, l'humidité, etc.

**- Couche réseau**

Il est responsable d'adresser chaque objet en utilisant une adresse unique et de transmettre de manière sécurisée les informations de la couche de perception à la couche application et vice versa. Le support de transmission et les protocoles de communication tels que WiFi, Bluetooth et ZigBee font partie de cette couche.

**- Couche d'application**

La couche application fournit les services demandés par les clients. Par exemple, la couche application peut fournir des mesures de température et d'humidité de l'air au client qui demande ces données. L'importance de cette couche pour l'IoT est qu'elle a la capacité de fournir des services intelligents de haute qualité pour répondre aux besoins des clients. La couche application couvre de nombreux marchés verticaux tels que la maison intelligente, la construction intelligente, le transport, l'automatisation industrielle et les soins de santé intelligents.

Bien que l'Internet des objets soit une notion relativement nouvelle, les technologies qui la rendent possible existaient depuis quelques années déjà. On parle alors des réseaux de capteurs sans fil et de la technologie d'identification par radio fréquence. .Dans cette section nous présentons les technologies basiques dans le contexte de l'Internet des objets.

- **Réseau de capteurs sans fil (RCSF) :** c'est un ensemble de nœuds qui communiquent sans fil et qui sont organisés en un réseau coopératif. Chaque nœud possède une capacité de traitement et peut contenir différents types de mémoires, un émetteur-récepteur RF et une source d'alimentation, comme il peut aussi tenir

compte des divers capteurs et des actionneurs[Sta08]. Comme son nom l'indique, le WSN constitue alors un réseau de capteurs sans fil qui peut être une technologie nécessaire au fonctionnement de l'IoT.

- **Identification par radio fréquence( RFID ) :** le terme RFID englobe toutes les technologies qui utilisent les ondes radio pour identifier automatiquement des objets ou des personnes. C'est une technologie qui permet de mémoriser et de récupérer des informations à distance grâce à une étiquette qui émet des ondes radio [ND06]. Il s'agit d'une méthode utilisée pour transférer les données des étiquettes à des objets, ou pour identifier les objets à distance. L'étiquette contient des informations stockées électroniquement pouvant être lues à distance[HZ13].

### I.2.4 Hétérogénéité de l'Internet des objets

Les objets ne sont pas égaux, car ils ont des caractéristiques variées. Certains objets sont mobiles, et ont une position qui évolue au cours du temps. D'autres peuvent être des objets transportables, tels qu'un téléphone mobile, un livre ou des vêtements, ou d'objets mobiles autonomes dotés de capacités motrices, tels qu'une voiture [Bil15].

Bnjamin Billet [Bil15] résume l'impact de l'hétérogénéité sur l'IoT, en distinguant deux sortes de l'hétérogénéité :

#### I.2.4.1 Hétérogénéité fonctionnelle :

Les objets possèdent des capacités spécifiques (statique ou mobile, alimenté en continu ou par une batterie, ressources matérielles, capteurs et actionneurs disponibles, etc.) et à chacune d'elle correspond des contraintes particulières (connectivité intermittente, durée de vie, tâches réalisables, etc.). Différentes approches et techniques doivent être considérées pour gérer les objets en fonction de leurs contraintes et de leurs différences propres.

### I.2.4.1 Hétérogénéité technique

Les technologies matérielles et logicielles utilisées pour construire les objets sont multiples et compromettent l'idéal de collaboration autonome entre objets. De plus, le développement d'applications est complexité, nécessitant des développeurs des connaissances spécifiques sur le fonctionnement de chaque objet.

## I.3 Web of Things (WoT)

L'Internet of Things (IoT) est considéré parmi les domaines de recherche les plus actifs. Avec l'exploration des différents points de vue et l'analyse des différentes méthodes, on constate le développement rapide et la propagation phénoménale de l'IoT, en citant comme exemple la renaissance du Web of Things (WoT), qui permet principalement de réduire les barrières entre le monde physique et le monde virtuel[DC11].

Le WoT propose de faire connecter les dispositifs physiques par les protocoles et les normes du Web et de les utiliser comme des ressources dans le développement d'applications Web[**nailton.and.al15**].

Le WoT peut se définir comme étant « un concept informatique qui décrit un avenir où les objets du quotidien sont entièrement intégrés avec le Web. Les systèmes informatiques qui permettent la communication avec le Web en sont une condition préalable. Ces dispositifs intelligents seraient alors en mesure de communiquer les uns avec les autres en utilisant les standards du Web existants [Tec18]».

## I.4 Cloud des objets

Le cloud computing est un modèle permettant un accès réseau omniprésent, pratique et à la demande à un pool partagé de ressources informatiques configurables (par exemple, réseaux, serveurs, stockage, applications et services) pouvant être rapidement provisionnées et libérées avec un minimum d'effort de gestion ou interaction avec le fournisseur de services.

La technologie Cloud permet une collecte, un stockage, une préparation et une analyse efficaces des données pour toute organisation. Avec l'adoption rapide des applications

SaaS (Software-asa-Service) et l'accumulation de données IoT à des taux sans précédent, la plupart de ces données ont été déplacées vers le cloud. L'IoT promet de futurs dispositifs intelligents capables de communiquer entre eux, avec des personnes et des applications[Gra11].

## I.4.1 Protocoles de communication

### I.4.1.1 MQTT

MQTT (Message Queuing Telemetry Transport)[SAH16] est un autre exemple de protocole applicatif de messagerie sur le web, dont son efficacité est de plus en plus approuvée dans de célèbres applications comme la messagerie sur le réseau social Facebook. Maintenant, son application pour les applications fondées sur les communications M2M dans l'IoT est largement investiguée. Le principe du fonctionnement du protocole MQTT est généralement concentré autour du modèle publish/subscribe. Le protocole fonctionne au-dessus du protocole TCP, avec un mécanisme de communication suffisamment simple pour mieux répondre aux fortes contraintes des réseaux de capteurs connectés à Internet.

Du point de vue architectural, les nœuds capteurs sont des publieurs qui se connectent tous à une entité centrale dite broker. Chaque message est publié au niveau du broker dans une rubrique qui convient au type de la donnée contenue dans le message. Les abonnés peuvent souscrire à plusieurs rubriques. A chaque fois qu'un nouveau message est publié dans l'une des rubriques d'intérêts, il sera immédiatement diffusé aux abonnés intéressés. Dans la figure ci-dessous, on illustre un exemple de l'architecture de communication M2M dans MQTT.

Le modèle est composé de deux abonnés qui élaborent des connexions TCP avec le broker. L'abonné numéro 1 adhère à la rubrique température ensuite, l'abonné 2 publie un nouveau message dans cette rubrique. Le broker se met, immédiatement, à transmettre une copie du message au client 1.

L'efficacité du protocole MQTT a été approuvée pour certaines applications comme : les applications médicales et industrielles. Il s'est avéré même qu'il représente une bonne solution pour les applications mobiles dans des environnements contraints car il présente

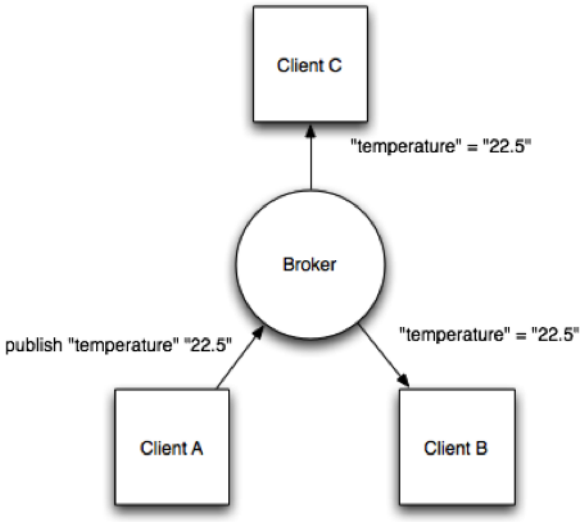


FIGURE I.3 – Exemple de communication M2M suivant le protocole MQTT

une faible empreinte mémoire, une faible consommation d'énergie avec une meilleure distribution de l'information aux destinataires.

### I.4.2 Plateforme IoT

Le choix d'une plateforme IoT nécessite de comprendre le principe de base de cette infrastructure particulière. Elle fait le lien entre, le composant, l'objet, la gateway, les données sur le cloud, les applications logiciels, etc. Elle permet de gérer avec granularité ces différents aspects. Elle prend le rôle d'agrégateur de données, d'outils Big Data donc et d'analyse.

Les fournisseurs proposent ainsi un ensemble d'outils décisifs dans différents secteurs. Certains se spécialisent dans l'installation d'infrastructures au sein des usines, d'autres s'adressent aussi aux "makers", concepteurs qui voudraient monter une preuve de concept rapidement.

Alors qu'il y a des centaines d'entreprises qui s'aventurent dans le développement de la plateforme IoT, d'entreprises comme Amazon, Microsoft et Google sont très en avance sur les autres concurrents [Obj18].

#### I.4.2.1 Amazon Web Services (AWS)

AWS IoT permet une communication sécurisée bidirectionnelle entre les appareils connectés à Internet (par exemple, capteurs, actionneurs, micro-contrôleurs intégrés ou appareils intelligents) et le cloud AWS. Cela permet de collecter les données de télémétrie de plusieurs appareils, et de stocker et d'analyser ces données. D'un autre coté, créer des applications qui permettent aux utilisateurs de contrôler ces appareils à partir de leurs téléphones ou tablettes [aws18].

**A. Composants AWS IoT**

AWS IoT se compose des éléments suivants [aws18] :

1. **Passerelle pour les appareils**

   Permet aux périphériques de communiquer de manière efficace et en toute sécurité avec l'AWS IoT. Agent de messages offre un mécanisme sécurisé pour que les appareils et les applications AWS IoT publient et reçoivent des messages les uns des autres. Il y a possibilité d'utiliser le protocole MQTT directement ou MQTT sur WebSocket pour effectuer des publications et s'abonner.

2. **Moteur de règles**

   Offre des fonctions de traitement des messages et d'intégration avec d'autres services AWS.Il est possible d'utiliser un langage SQL pour sélectionner des données à partir des charges utiles de messages, puis traiter et envoyer les données à d'autres services, comme Amazon S3, Amazon DynamoDB et AWS Lambda.

3. **Service de sécurité et d'identité**

   Partage la responsabilité de la sécurité dans le cloud AWS. Les appareils doivent protéger leurs informations d'identification pour envoyer des données en toute sécurité à l'agent de messages.

4. **Registre**

   Organise les ressources associées à chaque appareil dans le cloud AWS. Il enregistre des appareils et associe jusqu'à trois attributs personnalisés à chacun d'entre eux. Il

associe également des certificats et des ID de client MQTT à chaque appareil pour améliorer la capacité à gérer et dépanner les appareils.

5. **Service d'authentification personnalisé**

   définit des mécanismes d'autorisation personnalisée qui permet de gérer la stratégie d'authentification et d'autorisation à l'aide d'un service d'authentification personnalisé et d'une fonction Lambda. Les mécanismes d'autorisation personnalisée permettent à AWS IoT d'authentifier les appareils et d'autoriser des opérations à l'aide de stratégies d'autorisation et d'authentification par jeton de porteur.

   Les mécanismes d'autorisation personnalisée peuvent mettre en œuvre différentes stratégies d'authentification (par exemple : la vérification JWT, l'appel sortant de fournisseur OAuth, etc.) et doit renvoyer des documents de stratégies qui sont utilisés par la passerelle de l'appareil pour autoriser les opérations MQTT.

### B. Fonctionnement d'AWS IoT

AWS IoT permet à des appareils connectés à Internet de se connecter au cloud AWS et aux applications du cloud d'interagir avec les appareils connectés à Internet. Les applications IoT courantes recueillent et traitent des données de télémétrie de dispositifs ou permettant aux utilisateurs de contrôler un appareil à distance.

Les appareils indiquent leur état en publiant des messages, au format JSON, dans des rubriques MQTT. Chaque rubrique MQTT possède un nom hiérarchique qui identifie l'appareil dont l'état est mis à jour. Quand un message est publié dans une rubrique MQTT, il est envoyé au courtier de messages MQTT d'AWS IoT, qui est responsable de l'envoi de tous les messages publiés dans une rubrique MQTT à tous les clients abonnés à cette rubrique.

La communication entre un appareil et AWS IoT est protégée à l'aide de certificats X.509. AWS IoT peut générer un certificat où l'utilisateur peut utiliser le sien. Dans les deux cas, le certificat doit être enregistré et activé avec AWS IoT, puis copié dans son appareil. Quand l'appareil communique avec AWS IoT, il présente le certificat à AWS IoT sous la forme d'informations d'identification.

Il est recommandé que tous les appareils connectés à AWS IoT possèdent une entrée dans le registre d'objets. Le registre stocke des informations concernant un appareil, ainsi que les certificats utilisés par l'appareil pour sécuriser les communications avec l'AWS IoT.

Pour créer des règles qui définissent une ou plusieurs actions à exécuter en fonction des données d'un message, par exemple, insérer, mettre à jour ou interroger une table DynamoDB. Les règles utilisent des expressions pour filtrer les messages. Lorsqu'une règle correspond à un message, le moteur de règles appelle l'action en utilisant les propriétés sélectionnées.[aws18]

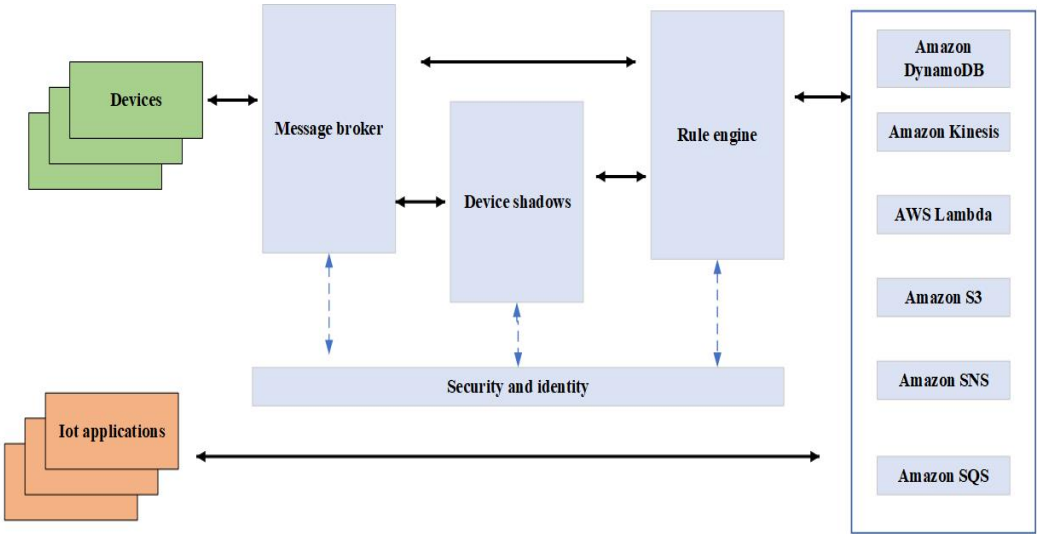


FIGURE I.4 – Fonctionnement d'AWS IoT

### I.4.2.2. Microsoft Azure IoT

Microsoft est très intéressé par l'élaboration de produits pour l'Internet des objets. Pour l'initiative la Plateforme IoT compatible avec les services de cloud Microsoft Azure, la suite Azure IoT est sur l'offre.

Pour traiter l'énorme quantité d'informations générées par les capteurs. Azure IoT Suite est livré avec Azure Stream Analytics pour traiter des quantités massives d'informations en temps réel [wik18].

### I.4.2.3. Google Cloud IoT

Google Cloud IoT est un ensemble de services entièrement gérés et intégrés qui permet de connecter, gérer et absorber facilement et en toute sécurité les données IoT à partir de dispositifs dispersés à l'échelle mondiale, de traiter et d'analyser / visualiser ces données en temps réel et de mettre en œuvre des changements et prendre des mesures au besoin.

Google peut faire bouger les choses. Avec sa Plateforme de bout en bout, Google Cloud figure parmi les meilleures plateformes IoT qui existe actuellement. Avec la capacité de gérer la grande quantité de données à l'aide de Cloud IoT Core, Google se démarque des autres. Des analyses avancées obtenues grâce à Google Big Query et Cloud

Data Studio[wik18].

Certaines des fonctionnalités de la Plateforme Google Cloud sont les suivantes :

- Accélération de l'entreprise.
- Accélération des appareils.
- Réduction des coûts avec le service Cloud.
- Écosystème partenaire.

### I.4.3 Comparaison entre platformes IoT

Ce tableau représente trois plateformes de cloud dans le but de faire une comparaison du coté de service iot core dans les contraintes suivantes [Vun18][goo18] :

| Sujet | AWS IoT | Azure IoT | Google cloud IoT |
|---|---|---|---|
| **Protocoles** | HTTPs, MQTT | HTTPs, MQTT, AMQP | MQTT, Android Things |
| **SDK** | C, Node.JS, Java, Python, IOS | NET, C, Java, Node.JS, Python | Java, Python, Go, Ruby |
| **Sécurité** | TLS (authentification mutuelle) | TLS (authentification du serveur uniquement) | IAM (gestion des identités et des accès) |
| **Communication** | Moteur de règles | Deux points d'extrémité différents | Communication bidirectionnelle |
| **Tarification** | Gratuit pour 1an puis 5$ par million de messages | Essais gratuits - cout par service | -Aucun coût initial requis.<br>-En moyenne 60% de moins<br>avec 0 $ payé d'avance. |

Tableau I.1: Comparaison entre platformes IoT

## I.5 Domaines d'application

Nous constatons que le concept de l'IoT est en pleine explosion vu que nous avons de plus en plus besoin dans la vie quotidienne d'objets intelligents capables de rendre l'atteinte de nos objectifs plus facile. Ainsi, les domaines d'applications de l'IoT peuvent être variés.

Plusieurs domaines d'application sont touchés par l'IoT [ACH17] :

**- Transport et logistique**

Voitures, trains, bus et vélos se voient de plus en plus dotés de capteurs, actuateurs et d'une logique de traitement des informations. Les routes aussi peuvent être munies de capteurs et tags (étiquettes) qui envoient des informations sur la circulation a ux stations de contrôles mais aussi directement aux voyageurs pour mieux gérer le trafic, améliorer la sécurité routière et guider les touristes.

**- Soins et santé**

Les objets connectés permettent de suivre et identifier en temps réel et à la demande , les outils, équipement et médicaments. Pour avoir des informations instantanément sur un patient peut souvent être déterminant.

**- Environnements intelligents**

Capteurs et actuateurs distribués dans plusieurs maisons et bureaux peuvent augmenter le confort dans ces environnements : le chauffage peut s'adapter à la météo, l'éclairage suivant l'horaire et la position du soleil ; des incidents domestiques peuvent être évités avec des alarmes et beaucoup d'énergie pourrait être économisée.

Les environnements intelligents peuvent aussi améliorer l'automatisation en milieu industriel avec un déploiement massif de tags RFID associés aux différentes étapes de la production. La ville intelligente est un exemple d'environnement intelligent. Le quartier d'affaires de Songdo en Corée du Sud est la première ville intelligente opérationnelle.

**- Signalement**

Dans ce domaine on trouve des systèmes permettant à un opérateur ou à un utilisateur de signaler un dysfonctionnement.

Dans une copropriété ou immeuble, un boîtier permettant de signaler à la régie gestionnaire un dysfonctionnement de l'ascenseur, un problème de propreté ou la nécessité de tondre la pelouse, utilisera les technologies de l'IoT pour sa communication.

# I.6 Conclusion

Tout au long de ce chapitre, nous avons présenté des définitions et notions de l'IoT et les fameux cloud IoT comme AWS Amazone, Microsoft Azure et Google IoT. De plus nous avons montré quelques domaines d'application des technologies IoT. Vu que les objets IoT et leurs services et applications doivent exploiter et communiquer les différents données capturées, l'annotation sémantique des données est nécessaire. Le chapitre qui suit présente ce concept important.

# CHAPITRE II

# ANNOTATION SÉMANTIQUE

# II.1 Introduction

L'objectif de l'annotation est de représenter la sémantique de l'élément annoté. Cette sémantique est essentiel pour l'utilisation automatique (par la machine) de cet élément. Les annotations utilisées pour annoter un document peuvent être en langage libre ou formel c'est le cas d'utilisation de l'ontologie.

Dans ce chapitre, nous introduisons d'abord le Web sémantique, par la suite une section va être consacrée à la notion de l'ontologie. Puis, nous allons présenter l'annotation sémantique et ses domaines d'utilisation. Finalement, et parce que notre s'inscrit dans le cadre de l'annotation sémantique dans le domaine de l'IoT, nous allons présenter les travaux voisins.

# II.2 Web sémantique

Les technologies du Web sémantique (SW) ont été largement utilisées pour interpréter et intégrer des données provenant d'une diversité de ressources sur le Web. Récemment, ils ont été étendus au domaine IoT pour améliorer la qualité des données et promouvoir l'interopérabilité. Ceci est réalisé en modélisant des données IoT basées sur des vocabulaires partagés qui peuvent être interprétés par différents agents logiciels. Ce processus est appelé annotation sémantique, ce qui implique l'utilisation de plusieurs normes SW telles que : OWL, RDF et RDF pour créer des modèles conceptuels (c.-à-d. Ontologie) pour décrire les concepts de domaine d'application et les relations qui existent entre eux [Ad17].

## II.2.1 Langages du web sémantique

Afin de représenter l'information sur le Web sémantique et simultanément la rendre à la fois syntaxiquement et sémantiquement interopérables à travers les applications, il est nécessaire d'utiliser des langages spécifiques. Il est important pour les développeurs du web sémantique de mettre d'accord sur la syntaxe des données et la sémantique avant de les coder dans leurs applications, il y a plusieurs langages, certains d'entre eux sont de niveau supérieur, d'autres sont de niveau inférieur.

D'une façon ou d'une autre, la plupart d'entre eux sont basés sur RDF (Resource Description Framework), et les RDF schémas [Bra15].

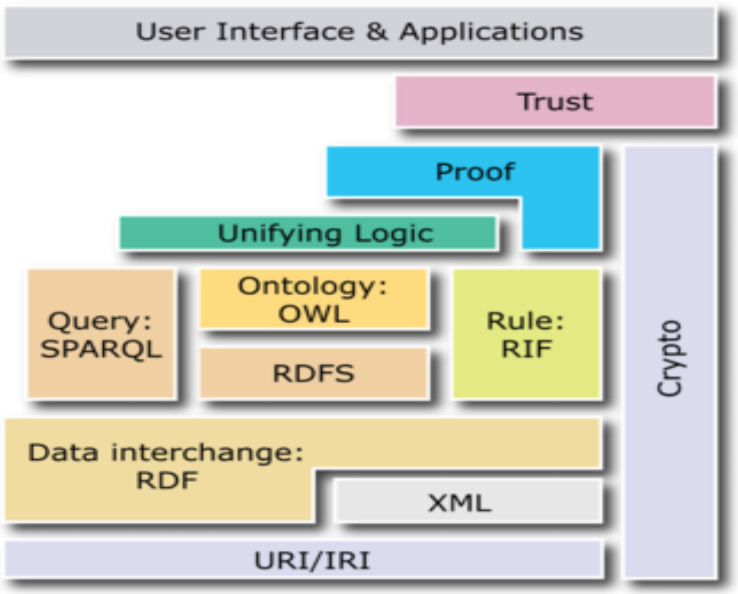


FIGURE II.1 – L'architecture en couches du Web sémantique

#### II.2.1.1 RDF

RDF est un langage d'assertion et d'annotation. Les assertions affirment l'existence de relations entre les objets. Elles sont donc adaptées à l'expression des annotations que l'on veut associer aux ressources du Web. RDF est un langage formel qui permet d'affirmer des relations entre des "ressources"[Ahm09]. Le modèle RDF définit trois types d'objets :

**- Ressources :** les ressources sont tous les objets décrits par RDF. Généralement, ces ressources peuvent être aussi bien des pages Web que tout objet ou personne du monde réel. Les ressources sont alors identifiées par leur URI (Uniform Resource Identifier).

**- Propriétés :** une propriété est un attribut, un aspect, une caractéristique qui s'applique à une ressource. Il peut également s'agir d'une mise en relation avec une autre ressource.

**- Valeurs :** les valeurs en question sont les valeurs particulières que prennent les propriétés.

Ces trois types d'objets peuvent être mis en relation par des assertions, c'est à dire des triplets (ressource, propriété, valeur), ou encore (sujet, prédicat, objet).

***Exemple :***

La page d'adresse «www.adresse de la page.com» a été écrite par «www.auteur de la page.com» qui a pour mail «mail de l'auteur»

- Sujet (ressource) : www.adresse de la page.com
- Prédicat (propriété) : créateur
- Objet (ressource) : www.auteur de la page.com

Table I.1 représente la syntaxe RDF/XML de l'exemple précédent :

```
<rdf :RDF
xmlns :rdf="http ://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns :s=http ://description.org/schema/
xmlns :v=http ://description.org/identity/>
<rdf :Description about="http ://www.adresse de la page ">
<s :createur rdf :resource = http ://www.auteur de la page />
</rdf :Description>
<rdf :Description about="http ://www.auteur de la page">
<v :Email>mail de l'auteur</v :Email>
</rdf :Description>
</rdf :RDF>
```

Tableau II.1: Exemple d'une syntaxe RDF

### II.2.1.2 Resource Description Framework shema (RDFs)

Comme son nom l'indique, RDFS a pour but de définir des schémas de méta-données. Il définit le sens, les caractéristiques et les relations d'un ensemble de propriétés. La définition peut inclure des contraintes pour les valeurs potentielles et l'héritage des propriétés d'autres schémas. Il est, en effet, une extension sémantique de RDF afin de fournir un mécanisme pour décrire les groupes associés de ressources et les relations entre les ressources.

L'intérêt de RDFS est qu'il facilite l'inférence sur les données et renforce la recherche sur ces données [Rad89].

# II.3 Ontologies

## II.3.1 Définitions

Ontologies telles que définies par "Sont des spécifications formelles et explicites d'une conceptualisation sémantique partagée qui sont compréhensible par la machine et des modèles abstraits de la connaissance consensuelle". Le concept d'ontologie a été transmis du domaine de la philosophie au domaine de l'informatique pour faciliter le processus de représentation de connaissances en décrivant certains domaines comme un ensemble de concepts et les relations entre eux. L'ontologie comprend quatre composantes principales : les classes, les relations, les attributs et les individus. Les classes sont le concept principal à décrire, qui peut être composé d'une ou de plusieurs sous-classes, utilisées pour définir des concepts plus spécifiques. Les individus sont des instances de classes ou leurs propriétés. Enfin, les relations en ontologie permettent de connecter tous les éléments précités [ALO16].

## II.3.2 Constituants d'une ontologie :

Une ontologie définit les termes et les concepts utilisés pour décrire et représenter un domaine de connaissance, ainsi que les relations entre eux. Par conséquent, une ontologie contient :

- **Concepts :** appelés aussi termes ou classes d'ontologie. Selon *Lortal Gaëlle*dans [Gaë02], un concept représente un type d'objet dans l'univers. Selon *Uschold et al.* dans [Usc96], un concept peut représenter un objet, une idée, ou bien une notion abstraite.
- **Relations :** Les relations représentent un type d'interaction entre les concepts d'un domaine. Nous distinguons deux types de relation [Gaë02] :
  - — Les relations inter-concepts : l'abstraction, la subsomption, l'équivalence, la disjonction et bien d'autres relations définies par le concepteur de l'ontologie.
  - — Les relations inter-relations : la subsomption, l'inverse, l'exclusivité, et l'incompatibilité.

- **Instances (Objets) :** des concepts correspondent aux individus concrets dans le domaine . *Fürst Frédéric* dans [Lji04] considère l'ensemble d'objets comme l'extension du concept, qui regroupe les objets manipulés à travers le concept.

### II.3.3 Langage de spécification d'ontologie

Plusieurs langages de spécification d'ontologies (ou langages d'ontologies) ont été développés pendant les dernières années, et ils deviendront sûrement des langages d'ontologie dans le contexte du Web sémantique. Certains d'entre eux sont basés sur la syntaxe de XML, tels que XOL (Ontology Exchange Language), SHOE (Simple HTML Ontology Extension -qui a été précédemment basé sur le HTML), OML (Ontology Markup Language), RDF (Resource Description Framework), RDF Schéma. Les 2 derniers sont des langages créés par des groupes de travail du World Wide Web Consortium (W3C). En conclusion, quatre langages additionnels sont établis sur RDF(S) pour améliorer ses caractéristiques : OIL (Ontology Inference Layer), DAML+OIL, OWL (Web Ontology Language) et le langage KAON. Notons que ces langages acceptent les conversations entre eux, soit directement ou bien par l'utilisation d'un langage pivot tel que le OWL [Ahm09].

#### II.3.3.1 OWL :

OWL "Web Ontology Language". Il est défini par le W3C, Le langage OWL est basé sur la recherche effectuée dans le domaine de la logique de description. OWL permet de décrire des ontologies, c'est-à-dire qu'il permet de définir des terminologies pour décrire des domaines concrets.

Une terminologie se constitue de concepts et de propriétés (aussi appelés rôles en logiques de description). Un domaine se compose d'instance de concepts. OWL est divisé en trois sous-langages, OWL Lite, OWL DL, et OWL Full. Ces langages visent différents groupes d'utilisateurs [Ahm09][SIH09].

- **OWL Lite :**répond à des besoins de hiérarchie de classification et de fonctionnalités de contraintes simples de cardinalité 0 ou 1 [SIH09]. Une cardinalité 0 ou 1 correspond à des relations fonctionnelles, par exemple, une personne a une

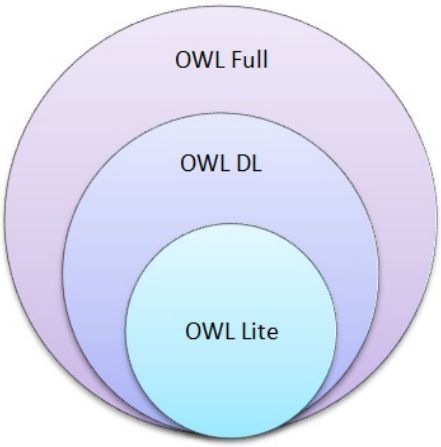


FIGURE II.2 – Dépendance hiérarchique entre les sous langages OWL

adresse. Toutefois, cette personne peut avoir un ou plusieurs prénoms, OWL Lite ne suffit donc pas pour cette situation [Ahm09].

- **OWL DL :** concerne les utilisateurs qui souhaitent une expressivité maximum couplée à la complétude du calcul (cela signifie que toutes les inférences seront assurées d'être prise en compte) et la décidabilité du système de raisonnement (c.à.d. que tous les calculs seront terminés dans un intervalle de temps fini). Il est nommé DL car il correspond à la logique de description [SIH09].

- **OWL Full :** est destiné aux utilisateurs qui veulent une expressivité maximale. Il se caractérise par une compatibilité totale avec RDF et RDF Schéma[SIH09], mais l'inconvénient d'avoir un haut niveau de capacité de description, quitte à ne pas pouvoir garantir la complétude et la décidabilité des calculs liés à l'ontologie .

## II.4 Annotation sémantique

### II.4.1 Définition

Selon *Bringay et al.* [**s.brin.et.al.03**] une annotation est une note particulière attachée à une cible. Ce dernier peut être un document, une collection de documents, une partie de

document ou bien une autre annotation.

*DESMONTILS et al.*[E D02] défini l'annotation comme une information graphique ou textuelle attachée à un document et le plus souvent placée dans ce document.

Il existe plusieurs rôles des annotations parmi eux on cite les suivants :
- Aide à la lecture et à la compréhension du document.
- Mémoire externe pour retrouver des informations ou indiquer un emplacement.
- Opérationnalisation de l'information.
- Pour la création d'indexe dans le processus de la recherche sémantique.

## II.4.2 Types annotation

### II.4.2.1 Annotation manuelle

Dans le cas de l'annotation manuelle, l'annotateur sélectionne la forme d'annotation, sélectionne ou pose l'annotation et enfin décide de la créer, le processus d'annotation sur papier est manuel. Il s'agit d'un processus fluide et sans effort. C'est le cas aussi des outils d'annotations informatiques qui essaient de reproduire simplement le processus sur papier vers l'outil informatique.

Mais, le processus d'annotation manuel devient pesant dans les trois cas suivants :
- l'annotateur doit créer un nombre important d'annotations cognitives.
- l'annotateur doit créer un nombre important d'annotations computationnelles.
- l'annotation créée est sémantique. Dans ce cas, l'annotateur doit spécifier pour chaque annotation le concept qu'elle représente dans une représentation formelle des connaissances (une ontologie ou un réseau sémantique de concepts). [Ilh12]

### II.4.2.2 Annotation automatique

L'annotation automatique signifie que la machine sélectionne elle-même ou pose l'annotation, crée les annotations, les enregistre et les affiche dans le cas ou elles sont cognitives.

Elle vise à alléger la charge de l'utilisateur tandis que ce type baisse la précision de la tâche et augmente sa productivité [Azo06].

#### II.4.2.3 Annotation semi-automatique

Afin de faire un compromis entre les deux tâches précédentes, leur combinaison est devenue nécessaire ; c'est ce qui est connu sous le nom « l'annotation semi-automatique », elle nécessite l'intervention de l'utilisateur pour valider les décisions du système [Hac06].

L'annotateur commence à annoter manuellement ; pendant ce temps la machine analyse textuellement ses annotations, et en génère des règles d'annotation (modèles d'annotation). Ensuite l'outil utilise ces modèles pour déduire des passages potentiellement annotables et y crée des annotations candidates. L'annotateur humain peut ensuite valider ou non les annotations proposées par l'outil [Azo06].

## II.5 Domaines d'utilisation

Les ontologies sont employées comme une forme de représentation de la connaissance d'un mondeou d'une partie du monde.

Dans plusieurs disciplines sont touchés par l'ontologie. Par exemple :

- Le web sémantique.
- L'intelligence Artificielle.
- Le génie logiciel.
- L'informatique biomédical.

## II.6 Travaux connexes

Il y a certaines tentatives d'utiliser des techniques web sémantiques pour traiter le problème de l'hétérogénéité de l'IoT pour l'annotation de données. Dans cette section, on se limite par les travaux qui intègrent les technologies SW à la périphérie des réseaux IoT où ils utilisent la notion de règle sémantique

### II.6.1 Travail de A.Gyrard

#### A. Description

Gyrard dans [A13] propose une architecture basé sur l'annotation pour mesurer des données hétérogènes capturées (passerelles de capteurs) avec la sémantique. Le but de ce travail est l'intégration de différentes ontologies et protocoles basés sur le raisonnement

sémantique pour cartographier des ontologies hétérogènes.

Les différentes étapes de ce processus, sont les suivantes :

1. Les passerelles d'agrégation convertissent les données détectées en mesures sémantiques à l'aide de technologies Web sémantiques (RDF, RDFS, OWL et ontologies de domaine).
2. Les applications basées sur la sémantique lient ces données à Linked Open Data pour préformer le raisonnement.
3. Une ontologie de mesure de capteur (SenMESO) a été conçue pour convertir automatiquement des mesures de capteurs hétérogènes en données sémantiques.

**B. Critique**

- L'absence de mécanisme de préparation et de filtrage des données, ce qui implique que la passerelle doit traiter et annoter toutes les données, quelle que soit leur importance dans le contexte d'une application.
- Techniques sémantiques lourdes (heavyweight) Augmente la consommation de ressources et le trafic réseau entre le cloud et la passerelle.

### II.6.2 Travail de I. Khan et al.

**A. Description**

Khan et al dans [**khan.&al**] proposent une architecture d'annotation ciblant principalement les réseaux de capteurs sans fil virtualisés. Cette approche basée sur deux niveaux : l'annotation des données et le stockage de l'ontologie, dans le but d'annotation des données. Les différentes étapes de ce processus, sont les suivantes :

1. Le premier reçoit une application de haut niveau de demande pour annoter des données de capteur.
2. chaque capteur impliqué dans le processus ayant son propre agent d'annotation.
3. L'agent d'annotation télécharge l'ontologie requise à partir de la superposition de l'ontologie.

**B. Critique**

- le modèle de description n'est pas suffisant pour décrire tous les aspects des données, qui sont nécessaires pour raisonner sur les données comme leur filtrage.
- l'augmentation de données générés qui qui entraîne la surcharge de trafic de réseau.

### II.6.3 Travail de M. Al-Osta et al.

**A. Description**

Al-Osta et al dans [ALO16] proposent une approche d'annotation sémantique qui peut être implémentée sur des passerelles IoT connectées à un nombre limité de capteurs. dans le but d'annotation des données. Les différentes étapes de ce processus, sont les suivantes :

1. *Etape de préparation* l'objectif principal de cette étape de préparation est d'analyser et de formuler les données brutes envoyées par les nœuds de capteurs et convertisse aux format XML.
2. *Etape d'annotation de données* Ce module reçoit ensuite les données du module de préparation au format XML, il met en œuvre le processus d'annotation pour marquer ces données par des concepts issus du mappage d'une ontologie de domaine d'application et d'ontologies de références. ils utilisents deux ontologies établies à savoir : ssn et DUL comme ontologies de référence et l'ontologie du domaine Sensor (Sdo).
3. Après l'annotation, le fichier convertisse au format RDF.

**B. Critique**

- Plusieurs présentations, XML dans la première étape et au final présenté au format RDF. Par conséquent, l'‘augmentation d'utilisation des ressources de passerelle.

## II.7 Conclusion

Dans ce chapitre de l'annotation sémantique, nous avons présenté, en premier, son cadre global à savoir le Web sémantique, puis son outil clé à savoir les ontologies. Après la présentation de ses différents domaines d'utilisation, nous nous concentrons sur l'annotation sémantique au domaine de l'IoT en présentant les différents travaux connexes à

notre travail. Le chapitre suivant présente notre approche de l'annotation sémantique et légère des données IoT qui se base sur l'utilisation de notre modèle de règles de filtrages des données.

# CHAPITRE III

# CONTRIBUTION

# III.1 Introduction

Dans ce chapitre, nous allons présenter notre proposition de l'approche d'annotation sémantique et légère pour les données IoT. Plus précisément, les systèmes des surveillances basés sur la technologie des IoT. Nous nous rappelons que l'annotation a pour objectif de permettre aux applications d'interroger et exploiter les données d'une manière automatique. Notre approche vise à répondre aux limites des autres approches que nous avons présentées précédemment. La limite majeure est que ces approches ne conviennent pas avec les dispositifs limités en terme des ressources, c'est souvent le cas des gateways (passerelles).

Dans notre proposition nous allons utiliser la notion de règle pour filtrer les données afin de minimiser la taille des données à annoter et aussi minimiser les données transférées au côté cloud. Dans la description de cette approche, nous allons commencer par la présentation de l'architecture générale puis nous allons en détailler, et pour bien illustrer notre approche nous montrons, comme cas d'étude, les feux de forêt.

# III.2 Architecture globale de l'approche proposée

L'architecture d'un système IoT en général est composée de trois catégories de composants, à savoir les nœuds de capteurs, les passerelles et les plates-formes de cloud.

Notre approche proposée est basée sur l'utilisation des ontologies pour l'annotation des données collectées et les règles pour le traitement des données. L'étape de traitement minimise la taille des données à envoyer au niveau du cloud en utilisant la notion des évènements. Par d'autres termes, nous classifions les données provenant des capteurs en événements (ex. alerte, avertissement et normal). Nous considérons seulement s'il y a de changement au niveau des événements déclenchés par les nouvelles données capturées ou bien d'autres paramètres que nous allons présenter dans la section détaillant notre architecture (Section 3). D'une manière générale, cette proposition va minimiser le coût (la taille) de traitement et d'envoi des données tout en évitant d'annoter et envoyer des données inutiles.

Notre approche est également organisée de trois niveaux, à savoir : les nœuds de capteurs, passerelle, et le niveau Cloud.

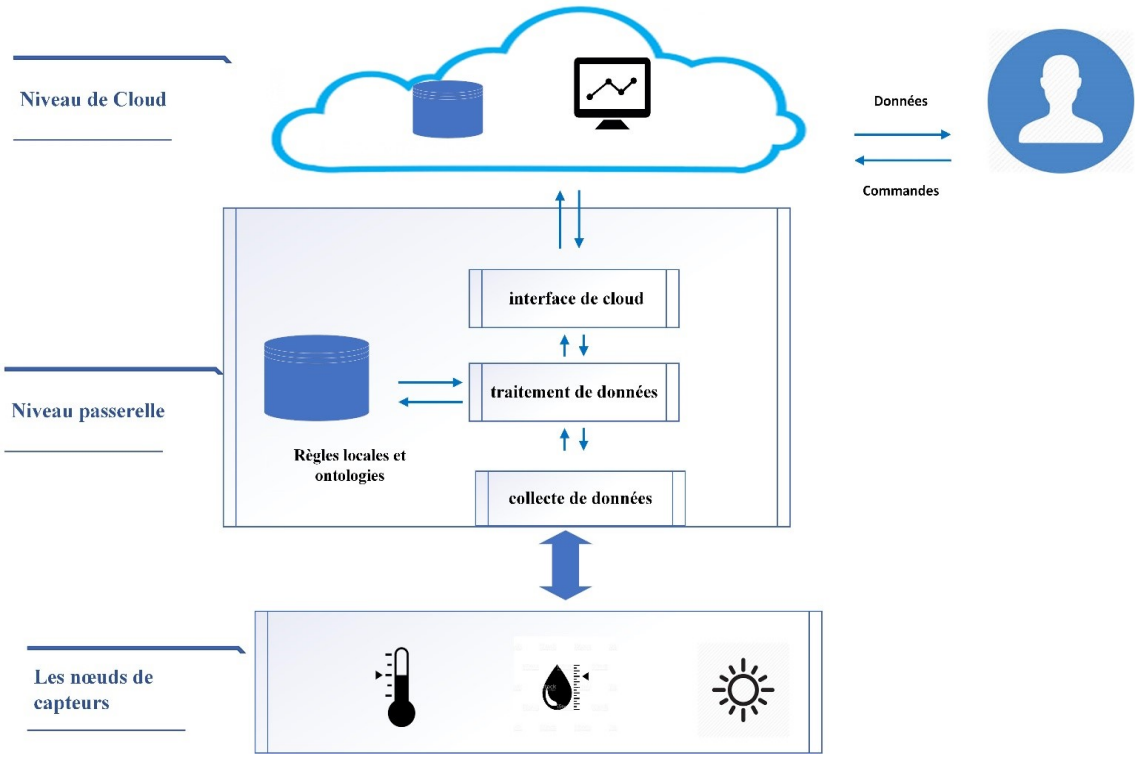


FIGURE III.1 – Architecture globale de l'approche proposée

1. **Niveau nœuds de capteurs :**

   c'est le niveau le plus bas et consistent en un ensemble de capteurs à ressources limitées. La tâche principale des nœuds de capteurs est uniquement de collecter des données et de les envoyer aux passerelles.

2. **Niveau passerelle :**

   notre proposition de base est dans ce niveau qui est composé en trois modules : module collecte les données, module de traitements et module interface au cloud. Les dispositifs dans la catégorie passerelle ont plus de ressources de calcul par rapport aux nœuds de capteurs. Une passerelle agit comme un collecteur des données

capturées et établit un pont de connexion entre les nœuds de capteurs et les services du cloud IoT.

Nous appliquons les règles des évènements pour minimiser les données à envoyer en filtrant celles qui ne considérés pas comme significatif, c.à.d. elles n'entrainent pas un changement aux types des événements. En outre, nous prenons en compte les limitations de ressources des dispositifs de passerelle, car le filtrage des règles et des concepts pour chaque passerelle permet d'instancier uniquement ceux qui sont pertinents pour les données collectées par les capteurs connectés à la passerelle.

3. **Niveau de Cloud :**

   À ce niveau, on trouve les connaissances partagées par les différents Gateways, eEt l'ontologie de domaine et l'ontologie des règles partagées par dans le Cloud.

   Parmi les modules plus importants utilisés à ce niveau sont :

   - **Module de stockage :** qui permet le sauvegarde de toutes les données collectées par les différents gateway, dans une base de données (souvant NoSQL) .
   - **Module de visualisation :** qui permet de présenter les différentes données collectées sous forme graphique comme les courbes.
   - **Module analytique :** qui permet d'analyser les données, on trouve aussi les algorithmes de Big data.

## III.3 Architecture détaillée de l'approche proposée

### 1. Module collecte de données :

Ce module est composé de deux sous-modules : lecture et agrégation. L'objectif du premier est la lecture des données de capteurs entrants (humidité, température... etc.) ça dépend le nombre de capteurs et les envoyer au sous-module Agrégation.

Après lecture, le sous-module d'agrégation collecte les données lues et les envoyer vers le module de traitement.

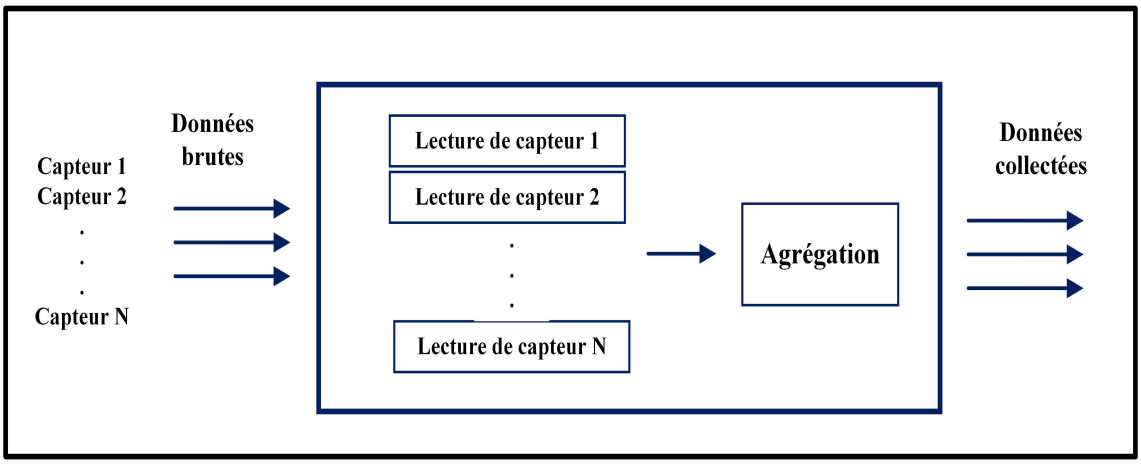


FIGURE III.2 – Architecture de module collecte de données

## 2. Module traitement de données :

Ce module joue le rôle essentiel de notre proposition, l'objectif principal de ce module de traitement est le filtrage et l'annotation des données collectées envoyées par le module collecte de données, en filtrant l'évènement redondant pour minimiser les ressources informatiques requises pour l'annotation

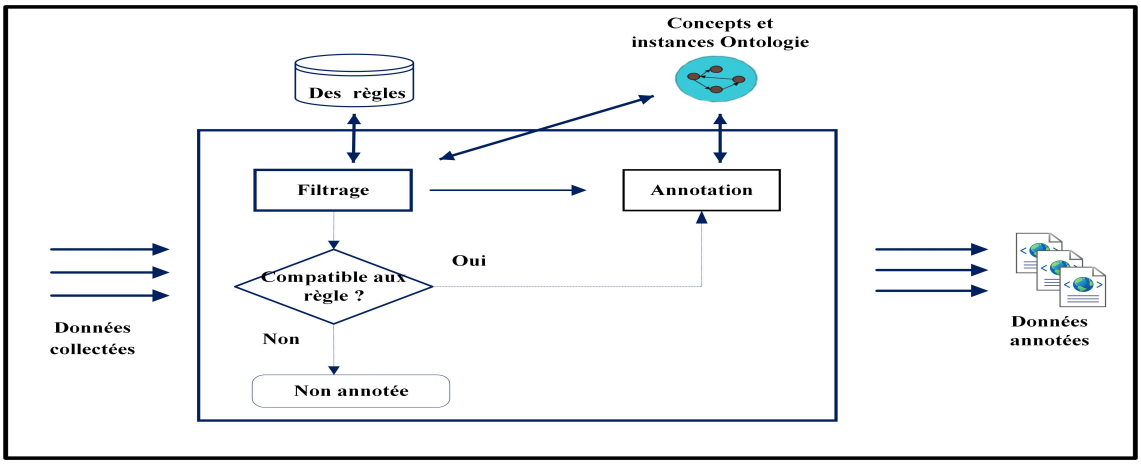


FIGURE III.3 – Architecture de module traitement de données

- **Sous module de filtrage**

Dans ce sous module, les données entrantes de module collecte de données sont filtrées à travers l'utilisation des règles afin de connaître si les nouvelles données entrainent un important nouvel évènement et il est donc nécessaire de les envoyer.

**Des règles :**En se basant sur le principe de partage des connaissances entre les différents gateways, nous proposons une ontologie des règles. Par la suite, on choisit les règles spécifiques et pertinentes au gateway.
Les conditions des règles sont des vérifications sur les données collectées et leurs conclusions sont la génération des évènements.
*Par exemple :*
Si( Température.Vlaleur>= 10)&(Température.Valeur< 30)Alors lancer_événement (normal).

Le moteur de règle inclus dans le sous-module de filtrage permet de déduire l'événement correspond aux nouvelles données. Par la suite, le sous-module de filtrage compare le nouvel événement généré à l'ancien événement et s'il y a un changement, il déduit que les données sont significatives et il doit les envoyer après leur annotation.
Par d'autre terme :
Si ( évènementActuel ! = évènementPrécédent) Alors envoyer_l'évènement.

En notant qu'il est en raison de la fiabilité, le système peut envoyer les nouvelles données même s'il n'y a pas de changement au niveau des événements si cela le cas après un nombre donné. Par exemple, on peut envoyer les données après 10 génération de l'événement 'Normal', cela permet d'envoyer un signe de vie 'Heartbeat'. C-à-d, le système est toujours fonctionnel et cela permet à éviter d'interpréter son arrêt (faillite) comme il n'y a pas d'un nouvel événement.
Si((évènementActuel = évènementPrécédent )& (Nombre.évènementsDupliqués>=10))Alors envoyer_l'évènement.

La figure III.4 présente graphe RDFS d'ontologie des règles.

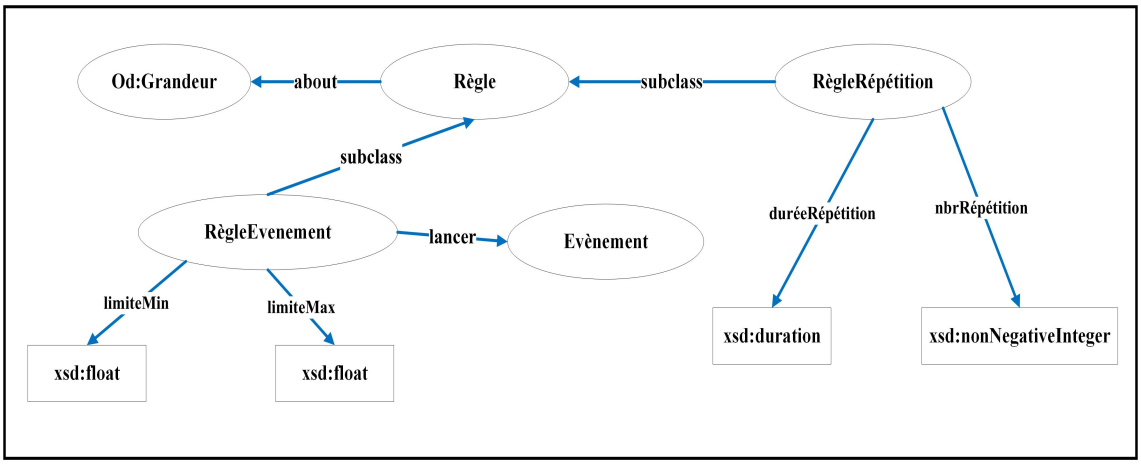


FIGURE III.4 – Graphe d'ontologie de règle

- On a créé classe Règle, classe Evènement, classe Règle Evènement et classe Règle Répétition.
  - **Classe Règle :**

    Présente la classe générique de toute les règles. Elle contient une propriété <about de> dont son domaine est dans la classe Grandeur (classe d'une ontologie de domaine).
  - **Classe Règle Evènement :**

    C'est une sous-classe de la classe Règle. Elle représente les règles qui peuvent déclencher des événements. Elle contient des propriétés : 'limiteMin', 'limiteMax' qui présentent un intervalle min et max pour identifierle type d'évènement et 'lancer' pour lancer un évènement.
  - **Classe Règle Répétition :**

    C'est une sous classe de la classe Règle. Cette classe apporte par son utilisation, une contrôle et configuration au système, par exemple, on peut considérer un évènement comme significatif même s'il est répété, si le nombre de répétition dépasse la valeur 'nbrRepetion'ou dépasse une 'dureeRepetition'.

On utilise le langage OWL en raison de sa richesse pour représenter les ontologies.

```
<?xml version="1.0"?>
<rdf:RDF xmlns:owl ="http://www.w3.org/2002/07/owl#"
    xmlns:rdf ="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:xsd ="http://www.w3.org/2001/XMLSchema#"
    xmlns:od="http://localhost/ontoDom.owl#">

    <owl:Ontology rdf:about="Ontologie des règles"/>

<!----------Déclaration des Class Règle------------------------->
    <owl:Class rdf:ID="Regle"/>
    <owl:ObjectProperty rdf:ID="about">
        <rdfs:domain rdf:resource="#Regle"/>
        <rdfs:range rdf:resource="od:Grandeur"/>
    </owl:ObjectProperty>

<!----------Déclaration des Class Regle------------------------->
    <owl:Class rdf:ID="Evenement"/>
<!----------class RegleEvenement------------------------->
    <owl:Class rdf:ID="RegleEevenement">
        <rdfs:subClassOf rdf:resource="#Regle"/>
    </owl:Class>

    <owl:DatatypeProperty rdf:ID="limiteMin">
        <rdfs:domain rdf:resource="#RegleEevenement"/>
        <rdfs:range  rdf:resource="xsd:float"/>
    </owl:DatatypeProperty>

    <owl:DatatypeProperty rdf:ID="limiteMax">
        <rdfs:domain rdf:resource="#RegleEevenement"/>
        <rdfs:range  rdf:resource="xsd:float"/>
    </owl:DatatypeProperty>

    <owl:DatatypeProperty rdf:ID="lancer">
        <rdfs:domain rdf:resource="#RegleEevenement"/>
        <rdfs:range  rdf:resource="#Eevenement"/>
    </owl:DatatypeProperty>

<!----------class RegleRepetition------------------------->

    <owl:Class rdf:ID="RegleRepitition">
        <rdfs:subClassOf rdf:resource="#Regle"/>
    </owl:Class>

    <owl:ObjectProperty rdf:ID="nbrRepetition">
        <rdfs:domain rdf:resource="#RegleRepitition"/>
        <rdfs:range rdf:resource="xsd:nonNegativeInteger"/>
    </owl:ObjectProperty>

    <owl:ObjectProperty rdf:ID="dureeRepetition">
        <rdfs:domain rdf:resource="#RegleRepitition"/>
        <rdfs:range rdf:resource="xsd:duration"/>
    </owl:ObjectProperty>

</rdf:RDF>

```

- **Sous module annotation**

Après avoir filtré les données, nous devons les annoter. Ce sous-module reçoit des données filtrées comme illustré dans la figure III.3, par la suite, il exécute le processus d'annotation pour marquer ces données en utilisant l'ontologie du domaine (concepts filtrés et instances). Alors que c'est au choix du développeur de construire (ou filtrer) une ontologie pour représenter les principaux concepts et les instances nécessaires au niveau de passerelle. Une fois les données sont annotées, on les envoie au module interface au cloud.

**Concepts et instances d'ontologies** :Ce répertoire contient seulement les concepts et les instances d'ontologie de domaine adéquats au gateway et les types des capteurs utilisés. Par exemple, si on utilise deux capteurs de température et humidité alors on peut utiliser uniquement les concepts de « Température » et « Humidité » leur instances (leur valeurs capturés), dans l'objectif de minimiser autant que possible les ressources au niveau de passerelle.

La figure III.5 présente graphe RDFS d'ontologie de domaine proposé.

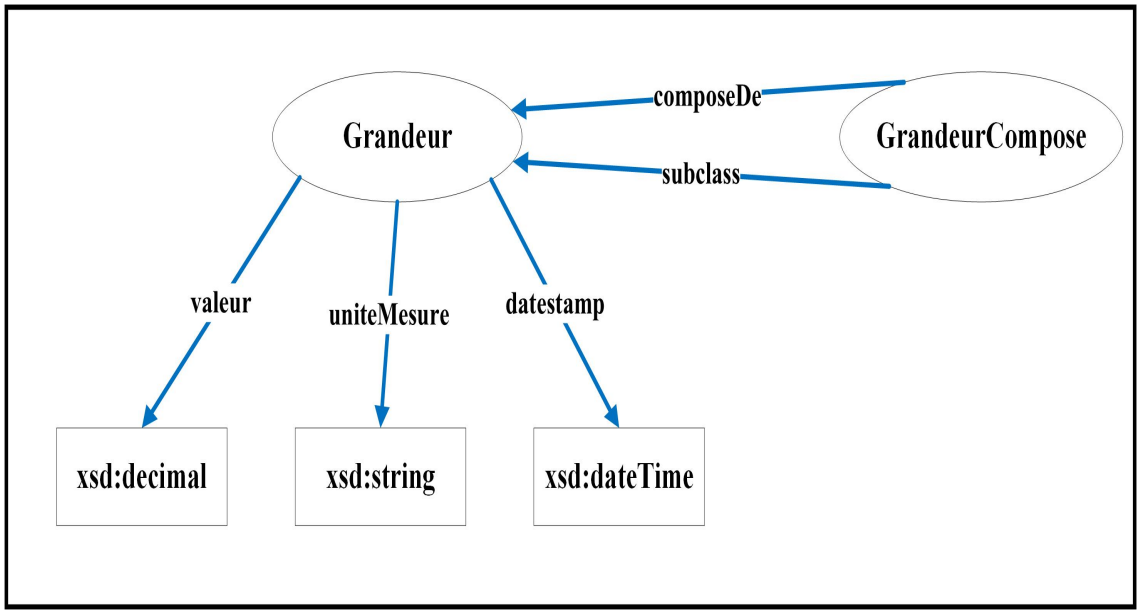


FIGURE III.5 – Graphe d'ontologie de domaine

- On a créé deux classe Grandeur et classe Grandeur composé.
  - **Classe Grandeur :**
    - ✓ C'est la classe qui représente tous les concepts de domaine (entrant de capteur) par exemple (Température).
    - ✓ Contient tous les propriétés de cette classe :valeur, uniteMesure, datestamp, station.
  - **Classe Grandeur composé :**
    - ✓ C'est une sous-classe de la classe Grandeur.
    - ✓ Elle contient :Les propriétés : composeDe qui réprésente ensemble de grandeur au moins 2 grandeur.

On utilise le langage OWL pour sa représentation.

```
<?xml version="1.0"?>
<rdf:RDF
    xmlns:owl ="http://www.w3.org/2002/07/owl#"
    xmlns:rdf ="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:xsd ="http://www.w3.org/2001/XMLSchema#">

    <owl:Ontology rdf:about="ontologie de domaine Generale"/>

    <owl:Class rdf:ID="Grandeur"/>
    <owl:DatatypeProperty rdf:ID="valeur">
        <rdfs:domain rdf:resource="#Grandeur"/>
        <rdfs:range  rdf:resource="xsd:decimal"/>
    </owl:DatatypeProperty>

    <owl:DatatypeProperty rdf:ID="uniteMesure">
        <rdfs:domain rdf:resource="#Grandeur"/>
        <rdfs:range rdf:resource="xsd:String"/>
    </owl:DatatypeProperty>

    <owl:DatatypeProperty rdf:ID="datestamp">
        <rdfs:domain rdf:resource="#Grandeur"/>
        <rdfs:range rdf:resource="xsd:dateTime"/>
    </owl:DatatypeProperty>

    <owl:Class rdf:ID="GrandeurCompose">
        <rdfs:label xml:lang="fr">classe Grandeur Composé</rdfs:label>
        <rdfs:subClassOf rdf:resource="#Grandeur"/>
    </owl:Class>

    <owl:DatatypeProperty rdf:ID="composeDe">
        <rdfs:label xml:lang="fr">Composé de</rdfs:label>
        <rdfs:domain rdf:resource="#GrandeurCompose"/>
        <rdfs:range  rdf:resource="#Grandeur"/>
    </owl:DatatypeProperty>

    <owl:Restriction>
        <owl:onProperty rdf:resource="#composeDe"/>
        <owl:minCardinality rdf:datatype="xsd:nonNegativeInteger">2
        </owl:minCardinality>
    </owl:Restriction>

</rdf:RDF>
```

## 3. Module interface cloud

Ce module prend en charge la communication avec le cloud et fourni une indépendance fonctionnelle entre le niveau physique et le niveau de services cloud.

Il est en outre composé du deux sous-module : transfert de données annotées et mettre à jour les règles et les concepts qui correspondent à la fonctionnalité de base de la passerelle.

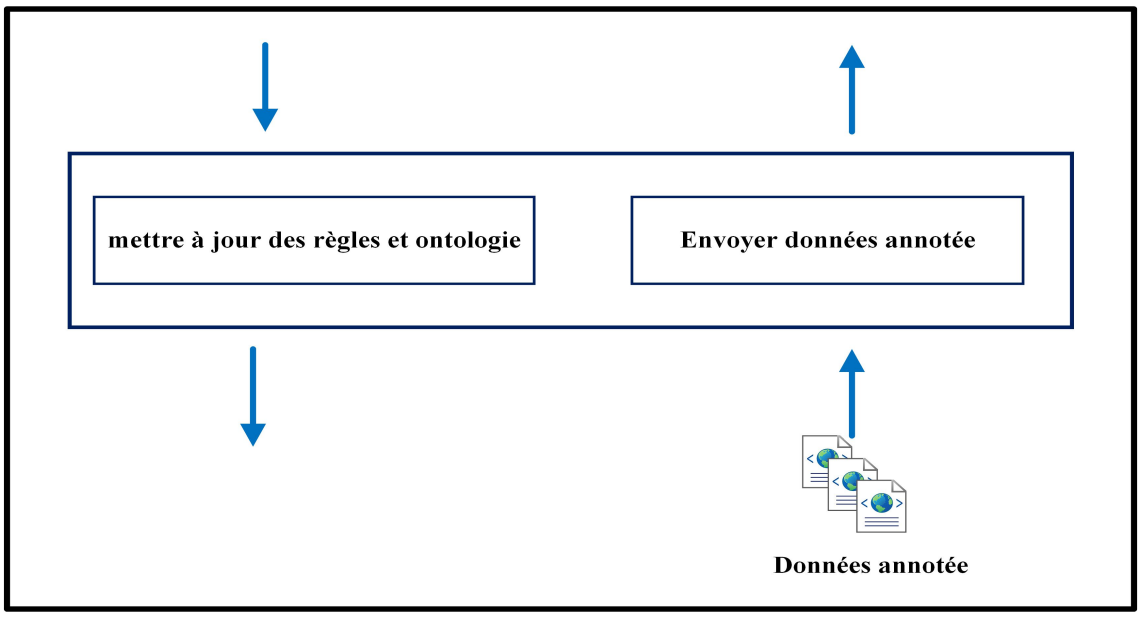


FIGURE III.6 – Architecture de module interface cloud

- **Sous module transfert de données**

L'objectif de ce module est d'envoyer les données annotées de niveau passerelle au niveau de cloud comme illustré dans la figure III.4 pour les publier et les utiliser par les utilisateurs du système (ou bien leurs applications).

- **Sous module mettre à jour les règles et les concepts**

Ce module gère les mises à jour des règles, les concepts d'ontologie du domaine et leurs instances, qui conviennent la fonctionnalité de passerelle et les capteurs connectés à la passerelle. Nous supposons que l'administrateur du système a l'ontologie complète du domaine (par exemple, au niveau du cloud) et que via ce module il peut sélectionner les

concepts concernés. Par la suite, on peut déduire d'une manière automatique les règles spécifiques au cloud à savoir celles qui contiennent au niveau de leurs préconditions l'un des concepts sélectionnés. De plus, les instances des concepts sont leurs valeurs capturées par les capteurs de passerelle.

Par exemple, si on a une passerelle connectée aux deux capteurs température et humidité, nous devons alors avoir au moins de concepts à savoir « Température » et « Humidité » et toutes les règles qui manipules ces deux concepts. Les valeurs (données) obtenues par les capteurs représentent leurs instances.

## III.4 Étude de cas

Au cours des trente dernières années l'Algérie est très touchée par les feux de forêts et donc grosses perte, parmi les causes important de feux de forêt sont des causes météorologiques, pour cela nous prenons cette phénomène comme un cas d'étude pour notre approche proposée.

L'Algérie est l'un des pays où le problème des feux de forêts, assez peu connu par la communauté scientifique, se pose avec acuité par son impact dévastateur, si en valeur absolue les superficies brûlées restent relativement modestes au regard d'autres pays du pourtour méditerranéen, la rareté des forêts et les menaces de désertification font que ces incendies ont un impact particulièrement désastreux. L'Algérie ne possède en effet que 4,1 millions d'hectares de forêts, soit un taux de boisement de 1,76 %. Or la fréquence rapprochée des incendies qui se suivent avec un intervalle de retour de moins de 10 ans à un impact catastrophique sur le plan écologique.

Un cumul de 42555 feux dans Algérie, ayant parcouru 910 640 hectares durant la période 1985-2010 l'évolution annuelle révèle la variabilité des feux de forêts et surtout l'existence d'années catastrophiques en 1993, 1994, 2000, 2007 .Tandis qu'une grande majorité des feux (soit 82,1 %) est maîtrisée avant que la surface parcourue ne dépasse 10 ha, 3,2 % des feux touchent des superficies de plus de 100 ha, dont 0,6 % sont supérieurs à 500 ha. Cette dernière catégorie représente 272 feux (dont 109 pour la seule année 1994). [**Ouahiba.et.al.**]

## III.5 Application de l'approche proposée a l'étude de cas

Dans le domaine de feux des forets, on utilise l'Indice Foret-Météo (IFM), ou Forest Fire Weather Index (FWI) , est un indice des conditions météorologiques propices aux incendies de forets. II fut publié en 1970 après plusieurs années de travaux par des agents de recherche du Service canadien des forets.[**Ouahiba.et.al.**]

FWI est une fonction calculée à base des variables suivantes : **Température (°C), Humidité relative (%), Vent (km/h), Pluie (mm)**.Le modèle de calcule de FWI dans le site Natural Resources Canada [1].

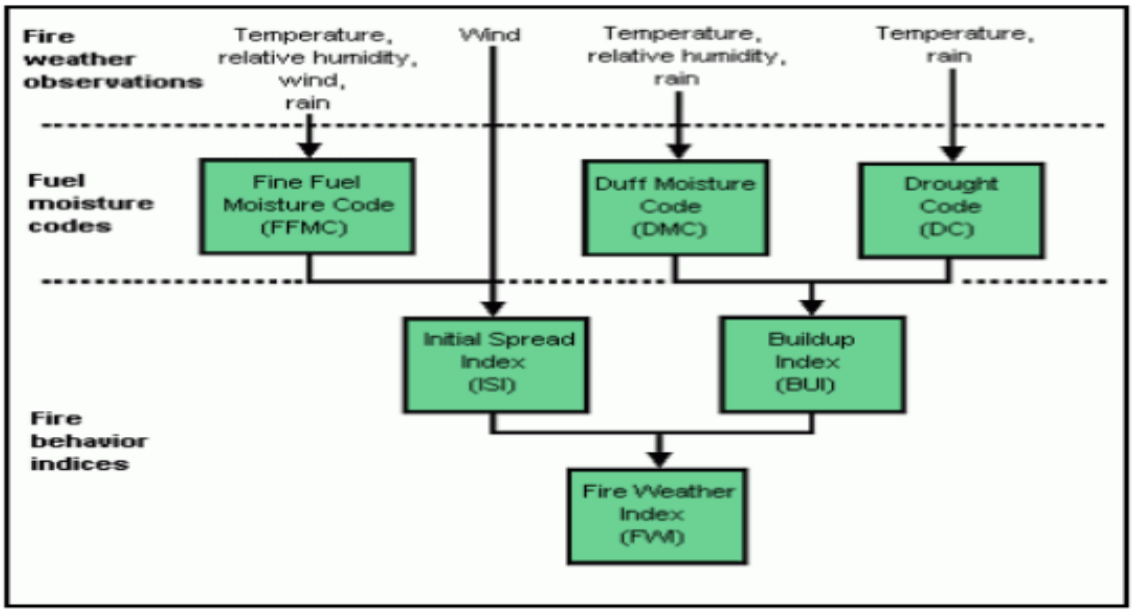


FIGURE III.7 – Indice FWI

Pour appliquer notre approche proposée il faut ajouter des concepts concernés de domaine feux de forêt dans ontologie de domaine, Pour cela nous construisons une ontologie nommé extension d'ontologie de domaine.

1. http ://cwfis.cfs.nrcan.gc.ca/background/summary/fwi

la figure III.8 présente graphe RDFS d'extension d'ontologie de domaine.

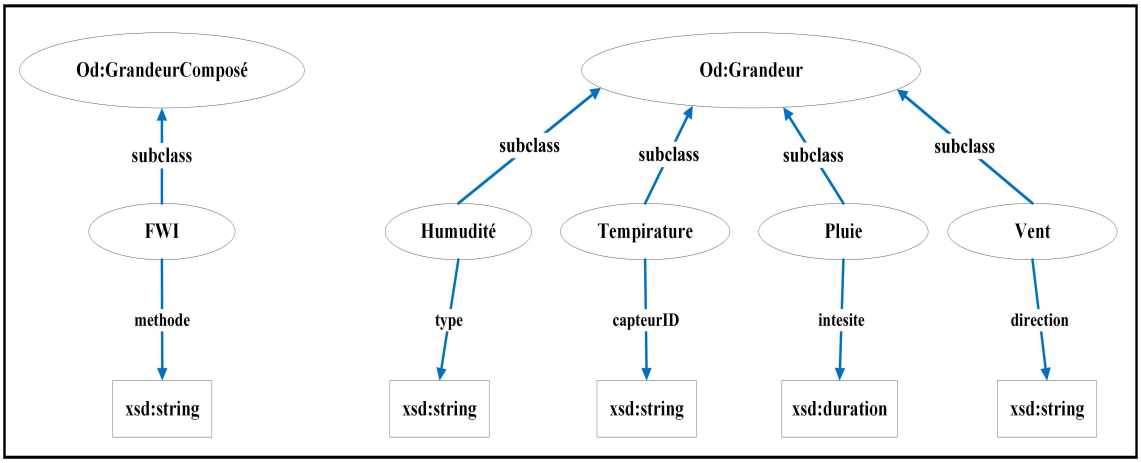


FIGURE III.8 – Graphe extension d'ontologie de domaine

- On crée les classes à partir des concepts utilisés de notre domaine.
  - ✓ Les classes Température, Humidité, Pluie, Vent sont tous des sous-classe de classe Grandeur.
  - ✓ Et aussi La classe FWI c'est une sous-classe de classe Grandeur Composé.
  - — **Classe Température :** elle contient la propriété 'capteurID' pour le type de capteur de température.
  - — **Classe Humidité :** elle contient la propriété 'type' représente le type d'humidité(vapeur ou liquide).
  - — **Classe Pluie :** elle contient la propriété 'intensité' représente hauteur d'eau précipité par unité de temps.
  - — **Classe Vent :** elle contient la propriété 'direction' pour avoir la direction de vent Nord par exemple.
  - — **Classe FWI :** elle contient la propriété 'méthode' pour chaque type de FWI il y'a une fonction spéciale.

On utilise le langage RDF pour sa présentation.

```
1   <?xml version="1.0"?>
2   <rdf:RDF
3       xmlns:owl ="http://www.w3.org/2002/07/owl#"
4       xmlns:rdf ="http://www.w3.org/1999/02/22-rdf-syntax-ns"
5       xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
6       xmlns:xsd ="http://www. w3.org/2001/XMLSchema#"
7       xmlns:od ="http://localhost/ontoDom.owl#">
8
9       <owl:Ontology rdf:about="ontologie de domaine extension"/>
10      <!----------class Temperature------------------------->
11      <owl:Class rdf:ID="Temperature">
12          <rdfs:subClassOf rdf:resource="#Grandeur"/>
13      </owl:Class>
14
15      <owl:DatatypeProperty rdf:ID="capteurID">
16          <rdfs:domain rdf:resource="#Temperature"/>
17          <rdfs:range  rdf:resource="xsd:string"/>
18      </owl:DatatypeProperty>
19      <!----------class Humudité------------------------->
20      <owl:Class rdf:ID="Humudite">
21          <rdfs:subClassOf rdf:resource="#Grandeur"/>
22      </owl:Class>
23
24      <owl:DatatypeProperty rdf:ID="type">
25          <rdfs:domain rdf:resource="#Humudite"/>
26          <rdfs:range  rdf:resource="xsd:string"/>
27      </owl:DatatypeProperty>
28      <!----------class Pluie------------------------->
29      <owl:Class rdf:ID="Pluie">
30          <rdfs:subClassOf rdf:resource="#Grandeur"/>
31      </owl:Class>
32
33      <owl:DatatypeProperty rdf:ID="intensite">
34          <rdfs:domain rdf:resource="#Pluie"/>
35          <rdfs:range  rdf:resource="xsd:duration"/>
36      </owl:DatatypeProperty>
37      <!----------class Vent------------------------->
38      <owl:Class rdf:ID="Vent">
39          <rdfs:subClassOf rdf:resource="#Grandeur"/>
40      </owl:Class>
41
42      <owl:DatatypeProperty rdf:ID="direction">
43          <rdfs:domain rdf:resource="#Vent"/>
44          <rdfs:range  rdf:resource="xsd:string"/>
45      </owl:DatatypeProperty>
46      <!----------class FWI------------------------->
47      <owl:Class rdf:ID="FWI">
48          <rdfs:label xml:lang="en"> Fire Weather Index </rdfs:label>
49          <rdfs:subClassOf rdf:resource="#GrandeurCompose"/>
50
51      </owl:Class>
52
53      <owl:DatatypeProperty rdf:ID="methode">
54          <rdfs:domain rdf:resource="#FWI"/>
55          <rdfs:range  rdf:resource="xsd:string"/>
56      </owl:DatatypeProperty>
57  </rdf:RDF>
```

Dans ce qui suit, nous allons présenter d'une manière générale [2] comment notre architecture va servir la mise en œuvre d'un système de détection des feux des forets.

◊ **Module collecte de données :**

Les valeurs de Température, Humidité, Vent et Pluie se capture à travers des capteurs au niveau de nœuds de capteur. Plus précisément, ces valeurs sont lues à travers le sous-module de lecture... et agrégées dans le sous-module d'agrégation et après l'agrégation. Par la suite, on calcule la valeur de FWI et les envoyer au module de traitement.

◊ **Module de traitement :**

Dans sous-module de filtrage, on applique les règles sur les données en fonction de la valeur FWI, elles appliquent sa condition pour gérer les évènements comme suite :

- Si(fwi >= 0) & (fwi < 3) **Alors lancer_évènement(Faible danger)**.
- Si(fwi >= 3) & (fwi <13) **Alors lancer_évènement(limité)**.
- Si(fwi >= 13) & (fwi <23) **Alors lancer_évènement(marquée)**.
- Si(fwi >= 23) & (fwi < 28) **Alors lancer_évènement(fort)**.
- Si(fwi >=28) **Alors lancer_évènement(très fort)**.

Le fichier suivant représente un extrait de la description des instances des règles utilisées. On utilise le langage RDF et RDFS pour sa représentation . Pour cela, on a créé :

- Des instances de la classe Evènements par exemple Faible danger.
- Des instances de la classe Règle Eevènement par exemple RE1.
- Des instances de la classe Regle Repitition par exemple regleEnvoiData.
- Des instances de la propriété de la classe Règle Evènement limitMin,limitMax.
- Des instances de la propriété de la classe Règle Répitition nbrRepetition,dureeRepetition.

2. Les détails techniques seront représentés dans le chapitre suivant.

```
1   <?xml version="1.0"?>
2   <rdf:RDF
3       xmlns:rdf="http://www. w3. org/1999/02/22-rdf-syntax-ns#"
4       xmlns:or="http://localhost/ontoRegle.owl"
5       xmlns:odi="http://localhost/ontoDomExtension.owl">
6
7       <or:evennement rdf:ID="FaibleDanger"/>
8
9       <or:RegleEevenement rdf:ID="RE1">
10          <or:limiteMin>0</or:limiteMin>
11          <or:limiteMax>3</or:limiteMax>
12          <or:lancer rdf:resource="#FaibleDanger"/>
13          <or:about rdf:resource="ode:FWI"/>
14      </or:RegleEevenement>
15
16      <or:RegleRepitition rdf:ID="regleEnvoiData"/>
17          <or:nbrRepetition>
18              <rdf:Bag >
19                  <rdf:li rdf:resource="0"/>
20                  <rdf:li rdf:resource="10"/>
21              </rdf:Bag >
22          </or:nbrRepetition>
23
24      <or:dureeRepetition>T10M</or:dureeRepetition>
25          <or:about >
26              <rdf:Bag >
27                  <rdf:li rdf:resource="ode:Temperature"/>
28                  <rdf:li rdf:resource="ode:Humidite"/>
29                  <rdf:li rdf:resource="ode:Pluie"/>
30                  <rdf:li rdf:resource="ode:Vent"/>
31                  <rdf:li rdf:resource="ode:FWI"/>
32              </rdf:Bag >
33          </or:about >
34      </or:RegleRepitition>
35
36  </rdf:RDF>
```

Après d'appliquer les règles sur les données brutes pour extraire les données filtrées, il vient le rôle du sous-module de l'annotation qui annote les données filtrées. Le fichier suivant représente la description des instances de l'ontologie **d'extension de l'ontologie du domaine**. On utilise le langage **RDF** pour toutes les instances.

- On crée les instances de concept de domaine et implémente les valeurs de chacun.

*Par exemple :*

pour instancier la classe Température, nous donnons 14.6 pour la propriété 'valeur', 2018-0-19T09 :00 :00 pour 'timestamp', °C pour 'unitéDeMesure', DHT22 pour 'capteurID'.

```
<?xml version="1.0"?>
<rdf:RDF
    xmlns:owl ="http://www.w3.org/2002/07/owl#"
    xmlns:rdf ="http://www.w3.org/1999/02/22-rdf-syntax-ns"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:xsd ="http://www. w3.org/2001/XMLSchema#"
    xmlns:ode ="http://localhost/ontoDomExtension.owl#">

    <ode:Temperature rdf:ID="T0">
        <ode:valeur>14.6</ode:valeur>
        <ode:timestamp>2018-05-19T09:00:00</ode:timestamp>
        <ode:uniteMesure>°C</ode:uniteMesure>
        <ode:capteurID>DHT22</ode:capteurID>
        <ode:station>oued</ode:station>
    </ode:Temperature>

    <ode:FWI rdf:ID="fwi0">

        <ode:composeDe rdf:resource="T0"/>
        <ode:composeDe rdf:resource="H0"/>
        <ode:composeDe rdf:resource="P0"/>
        <ode:composeDe rdf:resource="V0"/>

        <ode:valeur>4</ode:valeur>
        <ode:timestamp>2018-05-19T09:00:00</ode:timestamp>
        <ode:station>guemar </ode:station>
        <ode:methode>canadien</ode:methode>
    </ode:FWI>

</rdf:RDF>
```

◊ **Module interface de cloud :**

Ce sous-module transfert de données au fichier annoté et il l'envoie au niveau de cloud. De plus, S'il y'a des nouvelles règles proposées par l'administrateur ou l'utilisateur ou bien des nouveaux concepts de domaines alors le système les prend en considération par le sous-module de mettre à jour les règles et ontologie.

**- Niveau cloud :**

À ce niveau nous utilisons module de stockage et module de visualisation.

- **Module de stockage :** pour enregistrer les données annotées entrants de sous module interface cloud aux dans la base de données.
- **Module de visualisation :** pour présenter les données enregistrées de base de données.

## III.6 Conclusion

L'ajout de la sémantique aux données capturées soutient la tâche des applications IoT de haut niveau pour les interpréter et en extraire des connaissances. Dans ce chapitre, nous avons présenté notre approche d'annotation sémantique et légère des données au niveau de la passerelle IoT.

Pour répondre aux conditions de ressources limitées des périphériques IoT (souvent la passerelle), nous avons développé un mécanisme de filtrage de données, qui agit comme un concentrateur pour réduire la taille des données à traiter lors de l'étape d'annotation.

Dans le prochain chapitre, nous allons montrer l'implémentation de notre approche et son application sur le cas d'étude des feux de forêts.

# CHAPITRE IV

# IMPLÉMENTATION

# IV.1 Introduction

Le présent chapitre traite de l'implémention de notre approche théorique afin de rendre opérationnelles, nos propositions énoncées au chapitre précédent.

Dans ce chapitre, nous allons tout d'abord aborder les outils de système utilisés, montage et configurations du système et enfin nous allons présenter pseudo code et résultat d'exécution.

# IV.2 Outils de système utilisés

## IV.2.1 Outils matériels

Dans cette partie, nous présentons les outils matériels nécessaires utilisés pour implémenter notre approche.

Pour la passerelle nous utilisons Raspberry PI3 et capteur DHT22 pour capturer les données.

### IV.2.1.1 Raspberry PI

Raspberry Pi est une série de petite ordinatrice monocarte développée au RoyaumeUni par la Fondation Raspberry Pi pour promouvoir l'enseignement de l'informatique de base dans les écoles et dans les pays en développement[Ras18].La figure IV.1 représente l'architecture de raspberry PI 3.

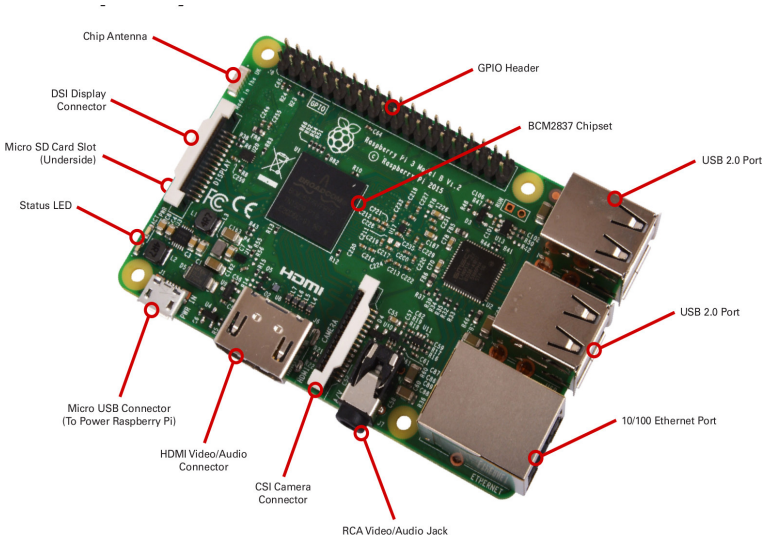


FIGURE IV.1 – Architecture de Raspberry Pi 3

Elle est caractérisé par :

- Carte mère Raspberry Pi 3 Type B.
- Processeur intégré Quad-core ARM Cortex-A53 1.2 GHz (Broadcom BCM2837).
- RAM : 1 GB.
- GPU Dual Core VideoCore IV Multimedia Co-Processor.
- Lecteur de cartes Micro SD.
- Ports : HDMI, 4x USB, RJ45, jack 3.5 mm, connecteurs pour APN et écran tactile.
- Wi-Fi b/g/n et Bluetooth 4.1.
- Support des distributions dédiées basées sur Linux et Windows 10.[LDL18]

### IV.2.1.2 Capteur numérique DHT22

Le AM2302/DHT22 est un capteur de température et d'humidité numérique hautepré-cision 16 bit. Il est calibré en usine avec sauvegarde des registres de calibration dans la ROM interne[Ele15].La figure IV.2 est une photo de capteur DHT22.

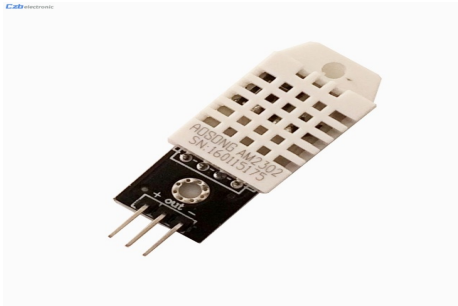


FIGURE IV.2 – Capteur DHT22

Il est caractérisé par :

- Les capteurs de température et d'humidité DHT à faible coût.
- Alimentation 3 à 5 V et E / S.
- Bon pour des lectures d'humidité de 0-100% avec une précision de 2-5%.
- Bon pour des lectures de température de -40 à $80^{\circ}$ C Précision de $\pm 0.5^{\circ}$C.
- Pas plus de 0,5 Hz de fréquence d'échantillonnage (une fois toutes les 2 secondes).
- Taille du corps 15.1mm x 25mm x 7.7mm.
- 3 broches avec espacement de 0,1.[**aosong**]

## IV.2.2 Outils logiciels

Dans cette partie, nous présentons les outils logiciels nécessaires pour implémenter notre approche.

### IV.2.2.1 Au niveau du Passerelle

- **Python :** Nous utilisons le langage de programmation python version3, supporte la programmation objet, multi paradigme et multi plateformes. Il favorise la programmation impérative structurée, fonctionnelle et orientée objet. Il est doté d'un typage dynamique fort, d'une gestion automatique de la mémoire par ramasse miettes et d'un système de gestion d'exceptions. Il est conçu pour optimiser la productivité des programmeurs en offrant des outils de haut niveau et une syntaxe simple à utiliser[Pyt18]

  Pour utiliser python dans l'implémention de notre approche, on a besoin des bibliothèques nécessaires RDFLib et Adafruit Python DHT.

- **RDFLib :** Bibliothèque Python pour exploiter les fichiers RDF, un parseur simple mais puissant Pour représenter des informations sous forme de graphiques .[Gro18]
- **Adafruit Python DHT :** Bibliothèque Python conçue spécifiquement pour fonctionner avec les capteurs de la série DHT d'Adafruit, on l'utilise pour lire l'humidité et la température depuis le capteur DHT22.[Lad18]

#### IV.2.2.2 Au niveau du Cloud

Nous choisissons les services offerts par la plateforme **Amazon Web Service (AWS) IoT** . Notre choix est motivé par sa supériorité envisagée aux autres plateformes des Cloud selon l'étude comparative entre les plateformes (sous-sectionI.4.3 de chapitre1).

- **X.509 :** sont des certificats numériques qui font appel à la norme d'infrastructure de clé publique X.509 pour associer une clé publique à une identité contenue dans un certificat.
- **AWS Rules :** Règles offrent à notre dispositifs la possibilité d'interagir avec les services AWS.Les règles sont analysées et les actions exécutées en fonction du flux de rubrique MQTT.
- **AWS DynamoDB :** service de base de données non relationnelle (NoSQL) entièrement géré par **AWS Rules**, et fournit des performances rapides et prévisibles avec les systèmes IoT.[aws18]
- **AWS Register**.
- **Protocole MQTT**.

#### IV.2.2.3 Eagle.io

Nous utilisons eagle.io pour représenter graphiquement les données enregistrées dans la base de données.

Les tableaux de bord (Dashboard) permettent aux utilisateurs de créer des vues personnalisées de leurs données dans des mises en page visuellement attrayantes en utilisant des jauges animées, des graphiques, des listes, des cartes et d'autres contrôles graphiques.[eag17]

### IV.2.3 Configuration Raspberry

Pour configurer raspberry, nous avons suivi les étapes suivantes :

- Téléchargement de l'image Raspbian sur le site 'https :/ /www.raspberrypi.org /downloads /raspbian/'.
- extraction de l'image raspbian.img sur une carte SD (plus 8 Gb) et la monter dans raspberry.

Après le démarrage du système, nous installons des bibliothèques python '**RDFLib** et **adafruit python DHT**.

- Installation de la bibliothèque Adafruit Python DHT illustrée dans la figure IV.3.

```
sudo apt-get install python-dev
sudo apt-get install python-pip
sudo apt-get install git
git clone https://github.com/adafruit/Adafruit_Python_DHT.git
cd Adafruit_Python_DHT
sudo python setup.py install
```

FIGURE IV.3 – Installation de la bibliothèque Adafruit

- Installation de la bibliothèque RDFLIb illustrée dans la figure IV.4.

```
~ $ pip install rdflib
Collecting rdflib
  Downloading rdflib-4.2.1.tar.gz (889kB)
    100% |████████████████████████████████| 892kB 227kB/s
Collecting isodate (from rdflib)
  Downloading isodate-0.5.4.tar.gz
Requirement already satisfied (use --upgrade to upgrade): pyparsing in ./anaconda/lib/python3.5/site-packages (from rdflib)
Building wheels for collected packages: rdflib, isodate
  Running setup.py bdist_wheel for rdflib ... done
  Stored in directory: /Users/ raspberry.user /Library/Caches/pip/wheels/5f/cc/22/bba65aa21c61d538dffb4960bd74ece936f3b0ef71ad
e47fb5
  Running setup.py bdist_wheel for isodate ... done
  Stored in directory: /Users/ raspberry.user /Library/Caches/pip/wheels/8a/b6/2c/9346712a1822e562d16c4d6a5ae242ebf45128a8cb2e
7b4abe
Successfully built rdflib isodate
Installing collected packages: isodate, rdflib
Successfully installed isodate-0.5.4 rdflib-4.2.1
```

FIGURE IV.4 – Installation de la bibliothèque RDFLib

### IV.2.4 Montage DHT22

Le montage de DHT22 illustré dans le figure IV.5, le DHT22 nécessite que le pin 1 est connecté à une source 3,3 V, pin 2 aux GPIO de Raspberry et pin 3 à GROUND.[**aosong**]

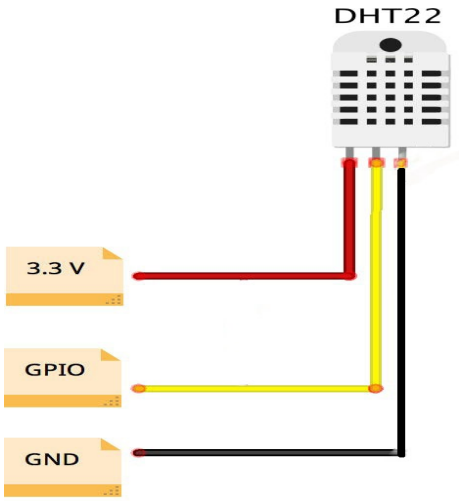


FIGURE IV.5 – Schéma de câblage DHT22

### IV.2.5 Configuration de service AWS IoT

Pour connecter notre Raspberry Pi à la plateforme AWS IoT, il faut suivre les étapes suivantes :

- Nous nous connectons à AWS Management Console et ouvrons la console AWS IoT à partir de l'adresse 'https ://aws.amazon.com/iot'.

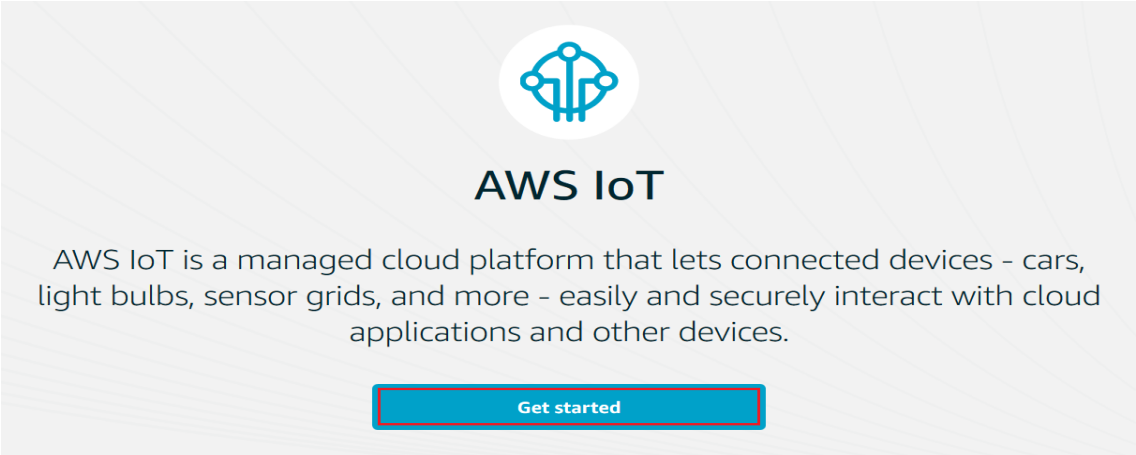


FIGURE IV.6 – Page principale de AWS IoT

- Création d'un objet (dispositif).

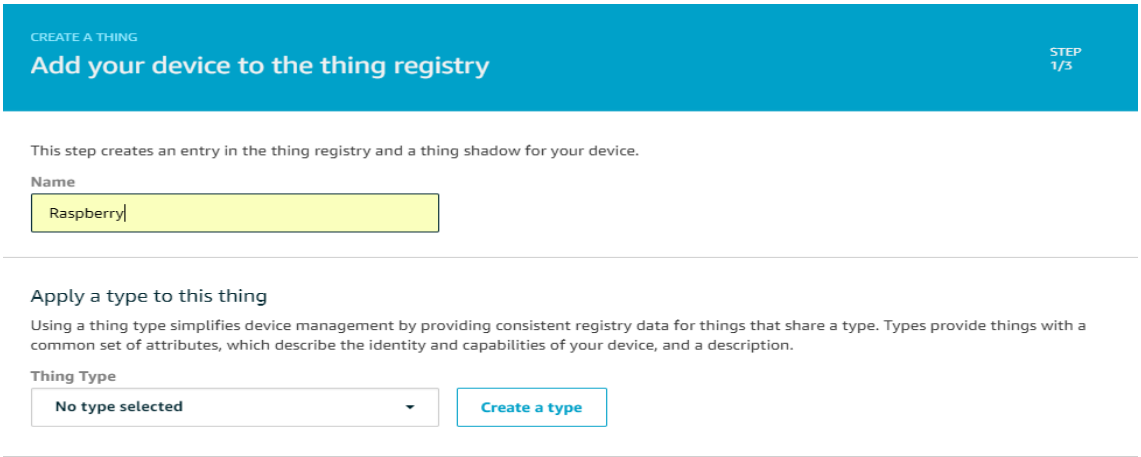


FIGURE IV.7 – Création de l'objet

- Création d'un certificat, Cela permet de générer un certificat X.509 pour notre objet créé.
- Téléchargement de nos clés publiques et privées, certificats et CA racine, puis activer le certificat X.509.

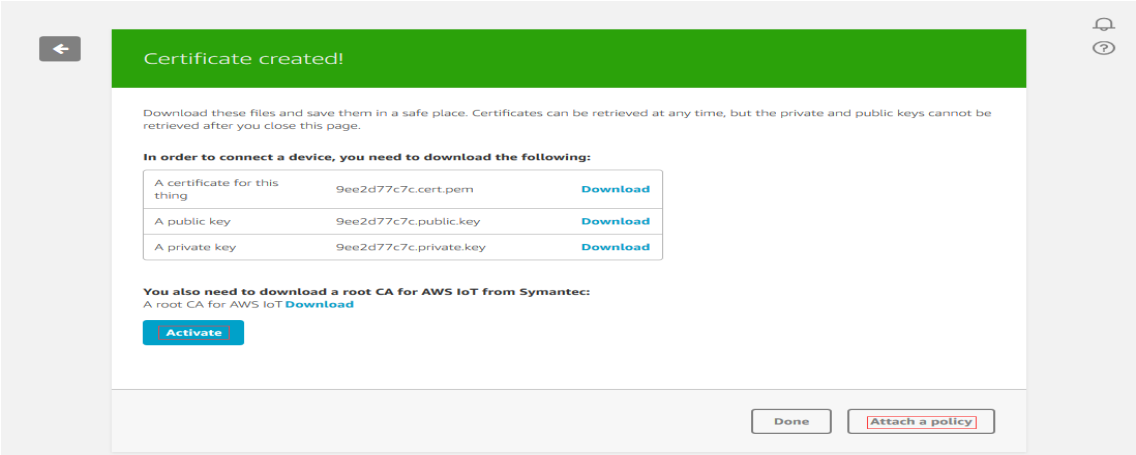


FIGURE IV.8 – Téléchargement les clés de certificat

- Création d'une stratégie (policy). Pour cela il faut entrer un nom pour la stratégie dans le champ **Name** et Dans le champ **Action**, entrer iot ( :∗) et aussi (∗) Dans le champ **Resource ARN**, enfin cocher **Allow**. Cela permet à notre Raspberry Pi de publier des messages dans AWS IoT.

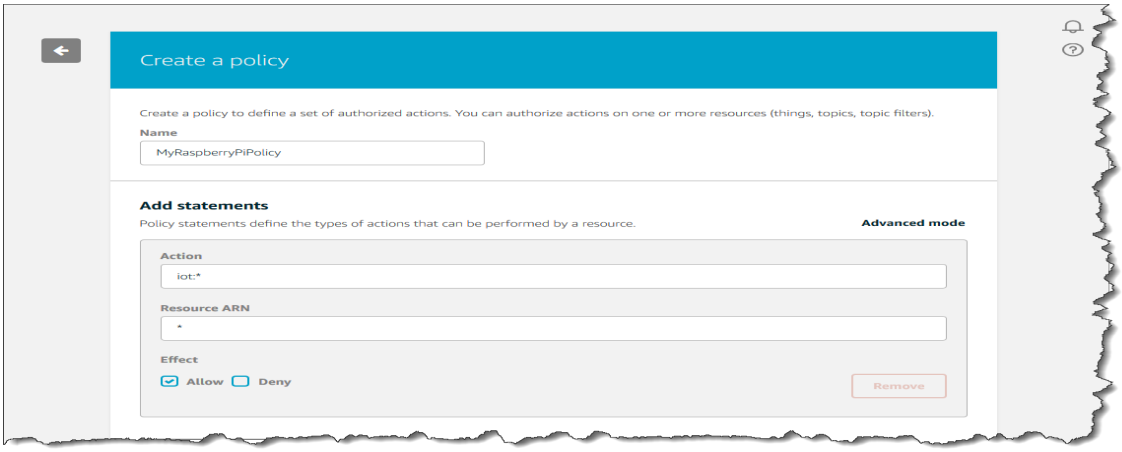


FIGURE IV.9 – Création policy

Nous créons une table dans DynamoDB, pour ce faire, il faut créer une règle DynamoDB. cette dernière permet de prendre des informations à partir d'un message MQTT entrant et de les mettre dans une table DynamoDB.

Les étapes pour la création d'une règle DynamoDB sont :

- Sur la page **Create a rule**, nous entrons le nom de notre règle et sa description.

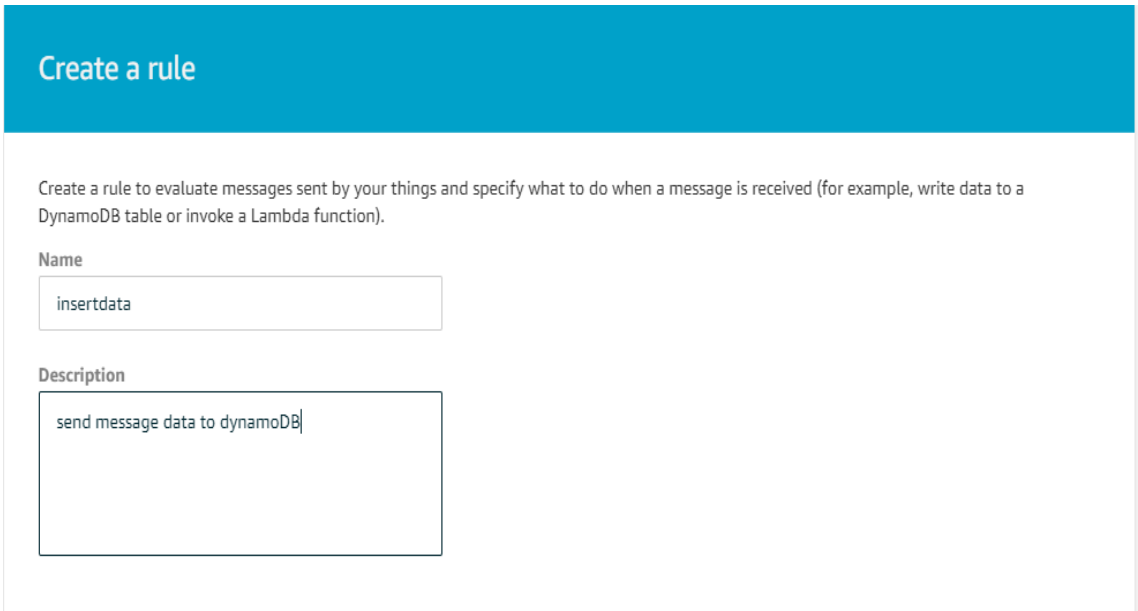


FIGURE IV.10 – Création de la règle

- Dans le champ **Attribute**, nous entrons (∗). Cela spécifie que nous souhaitons envoyer le message MQTT entier.
- Dans le champ **Topic** filter, nous utilisons le topic **Ann/topic** qui nous a déterminé la règle DynamoDB lors de la réception du message.

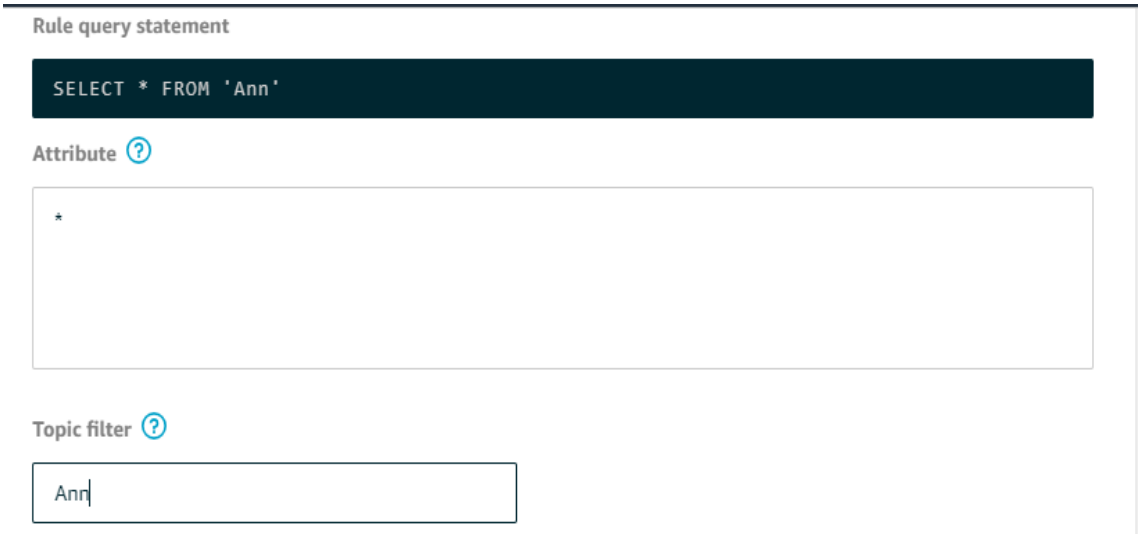


FIGURE IV.11 – Instruction de requête de règle

- Sur la page **Select an action**, nous sélectionnons **Insert a message into a DynamoDB table**, puis nous choisissons **Configure action**.

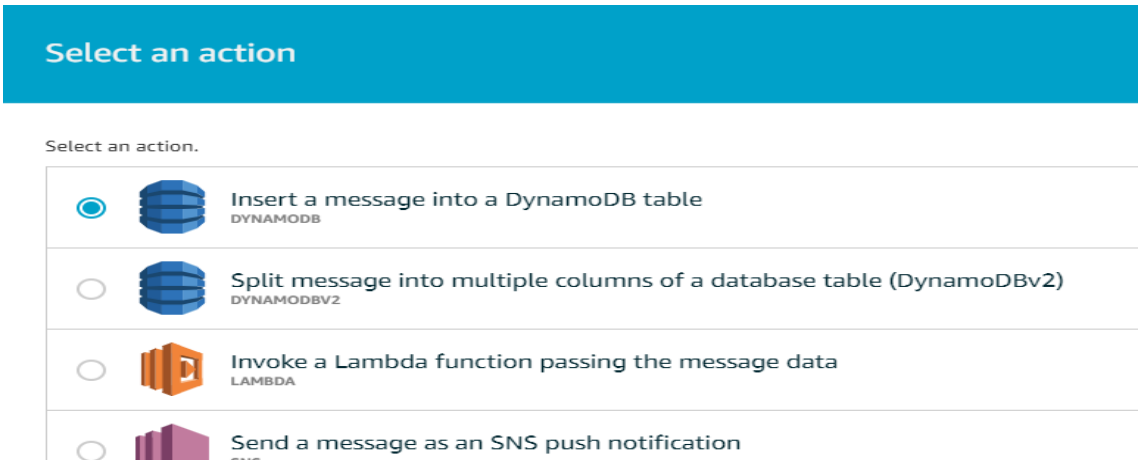


FIGURE IV.12 – Sélection l'action

- Après la sélection de l'action, nous créons la table base de données nommée TimeSeriesTable.

- Sur la page **Configure action**, nous choisissons la table TimeSeriesTable dans la liste déroulante **Table name**. Sous **Hash key value**, nous entrons $timestampp. Cela indique à la règle qu'elle doit prendre la valeur de l'attribut timestampp dans le message MQTT et l'écrire dans la colonne timestampp de la table DynamoDB.

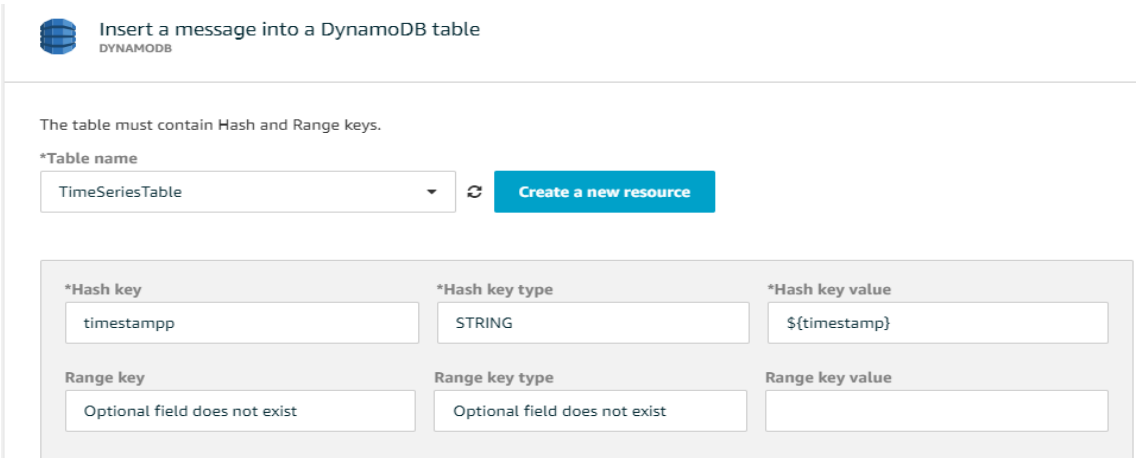


FIGURE IV.13 – Insertion du message dans dynamoDB

- Enfin, pour afficher la table que nous avons créée, nous revenons à la console DynamoDB et nous sélectionnons la table TimeSeriesTable.

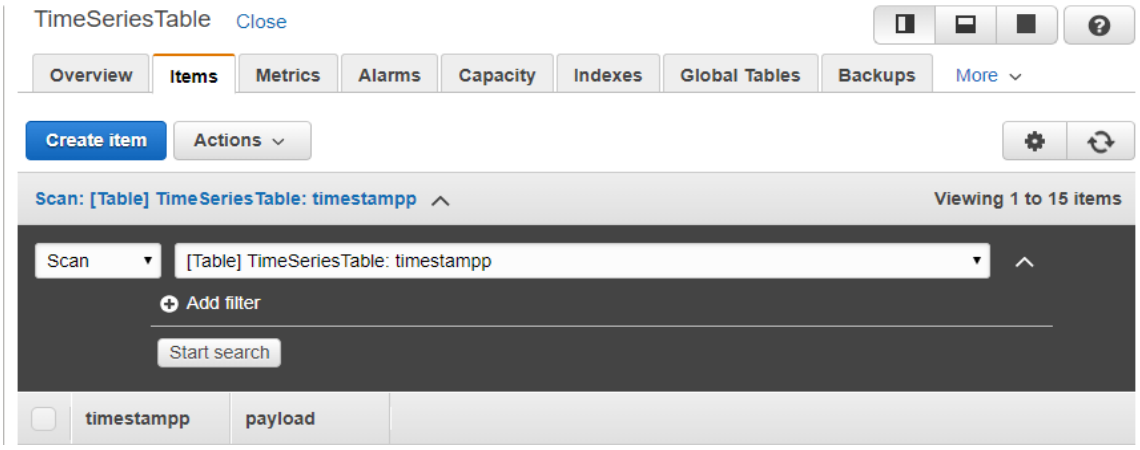


FIGURE IV.14 – La table TimeSeriesTable

### IV.2.6 Création d'un compte eagle.io

Nous nous sommes inscrites dans free trail de eagle qui nous permet d'utiliser le compte 30 jours gratuits.

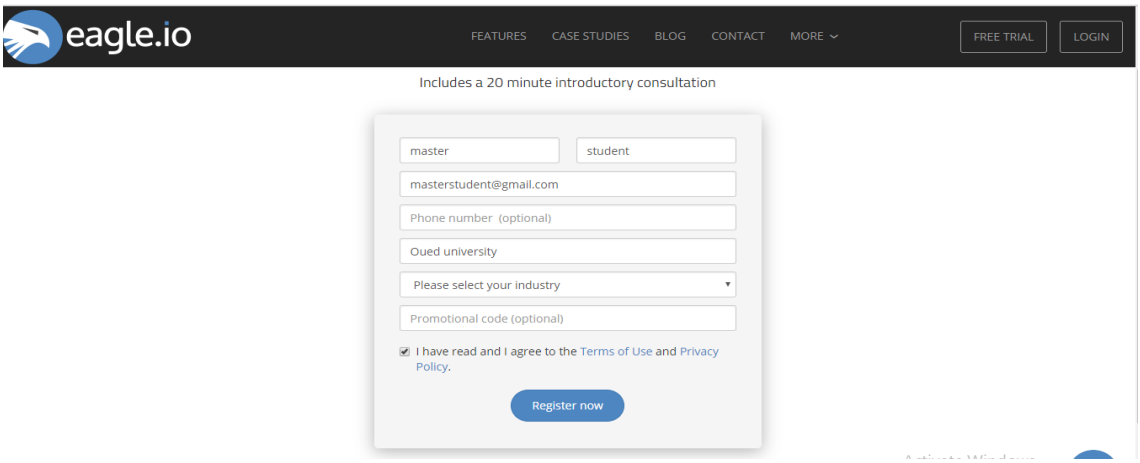


FIGURE IV.15 – Formulaire d'inscription dans eagle.io

## IV.3 Pseudo codes et résultat d'exécution

Dans cette section, nous présentons des pseudo codes et résultats aux quels nous avons abouti.

### IV.3.1 Instances de domaine

Ce fichier représente les instances de domaine au format JSON.

```
{
   "timestamp":"1527381172",
   "H2018-05-27 00:32:52.108345":{
      "rdf:type":{
         "value":"ode:Humudite",
         "type":"uri"
      },
      "od:valeur":{
         "value":"45.79999923706055",
         "type":"decimal"
      },
      "od:datestamp":{
         "value":"2018-05-27T00:32:52",
         "type":"dateTime"
      },
      "od:uniteMesure":{
         "value":"%",
         "type":"leteral"
      },
      "ode:type":{
         "value":"vapeur",
         "type":"literal"
      }
   },
   "P2018-05-27 00:32:52.109923":{
      "rdf:type":{
         "value":"ode:Pluie",
         "type":"uri"
      },
      "od:valeur":{
         "value":"992.59",
         "type":"decimal"
      },
      "od:datestamp":{
         "value":"2018-05-27T00:32:52",
         "type":"dateTime"
      },
      "od:uniteMesure":{
         "value":"mm",
         "type":"leteral"
      },
      "ode:intensite":{
         "value":"P1DT2H",
         "type":"literal"
      }
   },
   "V2018-05-27 00:32:52.400624":{
      "rdf:type":{
         "value":"ode:Vent",
         "type":"uri"
      },
      "od:valeur":{
         "value":"988.51",
         "type":"decimal"
      },
      "od:datestamp":{
         "value":"2018-05-27T00:32:52",
         "type":"dateTime"
      },
      "od:uniteMesure":{
         "value":"kmph",
         "type":"leteral"
      },
      "ode:direction":{
         "value":"nord",
         "type":"literal"
      }
   }
}
```

### IV.3.2 Pseudo codes python

- **Lecture de capteur DHT22**

  Ce code représente une fonction python qui permet de lire la valeur de température et l'humidité de FWI, On utilise la bibliothèque Adafruit DHT22.

```
def sensing(self):
    h,t =Adafruit_DHT.read_retry(22, 2)
    self.setValeur(h)
```

FIGURE IV.16 – Lecture de capteur DHT22

- **Filtrage des règles évènements**

  Ce code représente une fonction python qui permet de filtrer les valeurs de FWI, on utilise une requête SPARQL.

```
def regleEvent (val,classe):

    if("ori:RegleRepitition//"+classe not in FilterEngine.RuleGrandeurList):
        return 0

    query = 'select ?regle  ?evnt '
    query += ' where {'
    query += ' ?regle rdf:type or:RegleEevenement .'
    query += ' ?regle or:about ?grand .'
    query += ' ?regle or:lancer ?evnt .'
    query += ' ?regle or:limiteMin ?lmin .'
    query += ' ?regle or:limiteMax ?lmax .'
    query += ' FILTER( '
    query +=   ' str(?grand) ="'+str(classe)+'"'
    query +=   ' && xsd:integer(?lmin) <= '+str(val)
    query +=   ' && xsd:integer(?lmax) > '+str(val)
    query += ' ) . }'
    rows = FilterEngine.executeQuery(query)

    for row in rows:
        event = FilterEngine.namespace+row.evnt.rsplit('#')[-1] # return 'ori:FaibleDanger'
        return event
    return 0
```

FIGURE IV.17 – Filtrage des règles évènements

- **Main loop**

  Ce code python permet d'exécuter les taches principales du passerelle, telles que, lire les valeurs de grandeurs, filtrage des donnée selon des règles sémantiques et envoi des évènements et annotations au Cloud.

```
loop=0
while True:
    loop += 1
    print('\n~~~~~~~~~~~~~~~~~~~~loop %s\n'%loop)
    sdata=''
    alldata=''
    #reading of grandeurs data
    for grandeur in grandeurs:
        grandeur.sensing()
        #alldata += grandeur.getJson()+'\n'
        if (grandeur.getValeur() is not None):
            if ( FilterEngine.regleRep(grandeur.getNbrRepetition(),grandeur.UriRef)==1):
                sdata += ", "+grandeur.getJson()
            alldata += ", "+grandeur.getJson()

    #sending event to cloud
    if(fwi.getValeur() is not None ):
        e = FilterEngine.regleEvent(fwi.getValeur(),"ode:FWI")
        myAWSIoTMQTTClient.publish('event', '{"event":"'+e+'"}', 1)
        print('Published topic %s: %s\n' % ('event', '{"event":"'+e+'"}'))

    #sending data instance annotion to cloud
    if(sdata is not None):
        timestamp = calendar.timegm(time.gmtime())
        messageJson ='{"timestamp":"'+str(timestamp)+'"'+sdata+'}'
        myAWSIoTMQTTClient.publish(topic, messageJson, 1)
        print('Published topic %s: %s\n' % (topic, messageJson))
    time.sleep(2)
```

FIGURE IV.18 – La fonction main loop

### IV.3.3 Résultat d'exécution

La figure IV.19 représente la table TimeSeriesTable aprés l'annotation de données.

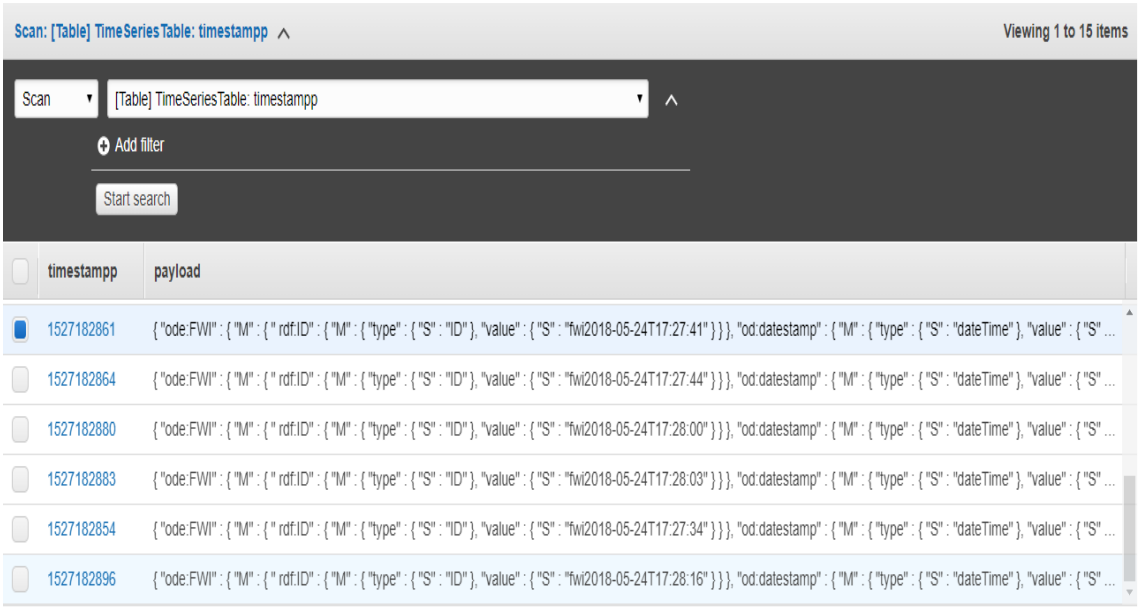


FIGURE IV.19 – Table TimeSeriesTable

La figure IV.20 représente le tableau de bord de données enregistrées dans la table TimeSeriesTable.


FIGURE IV.20 – Table de bord du résultat

1. Valeur de température.
2. Valeur de Vent.
3. Valeur d'humidité.
4. Valeur de pluie.
5. Valeur de FWI.
6. Valeur d'évènement.
7. Notification (Alarm).
8. Table historique.
9. Les courbes standard de chaque valeur.

La figure IV.21 représente les données générées dans une heure avec et sans des règles sémantiques .

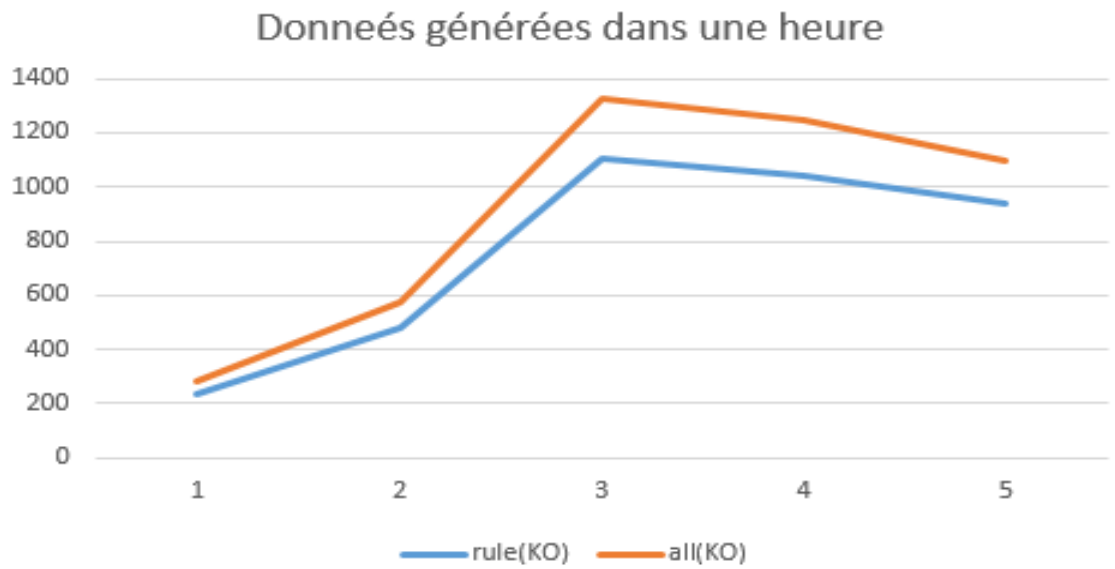


FIGURE IV.21 – Donneés générées dans une heure

La figure IV.22 représente la cumule de données générées par le temps (5h) avec et sans des règles sémantiques .

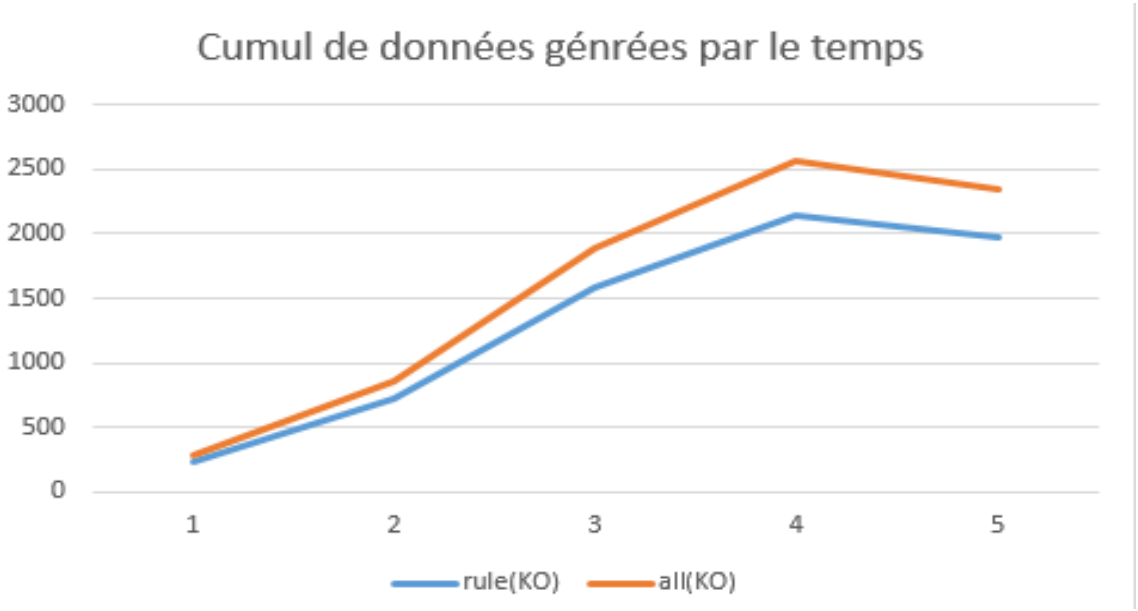


FIGURE IV.22 – Cumul de données génrées par le temps

## IV.4 Conclusion

Dans ce chapitre nous avons présenté les détails d'implémentation de notre approche. L'avantage de notre approche est de minimiser la quantité des données à annoter par une méthode de filtrage des données inutiles à base des règles sémantiques, pour répondre aux conditions de ressources limitées sur les périphériques de passerelle et celles au niveau du cloud tels que, le coût de stockage et traitement des données. Nous avons utilisé le service AWS DynamoDB pour stocker nos données annotées dans une Base noSQL afin de les présenter dans le tableau de bord (Dashboard). Les résultats obtenus dans le domaine étudié montrent l'efficacité de la minimisation de la quantité de données envoyées au Cloud, ce qui, à son tour, améliore la performance globale du système.

# CONCLUSION GÉNÉRALE

La délégation de la tâche de traitement des données sur l'environnement d'un système IoT permet d'effectuer des fonctions intelligentes à proximité des puits des capteurs, telles que le traitement de modèles sémantiques et le raisonnement.

Dans ce mémoire, nous avons présenté notre approche d'annotation sémantique et légère pour les données IoT.

Au niveau de passerelle nous avons proposé un modèle basé sur le web sémantique pour l'annotation des données sur la passerelle IoT. Pour répondre aux conditions de ressources limitées sur les périphériques de passerelle, nous avons développé des règles sémantiques pour le filtrage des données à annoter.

Au niveau de cloud nous avons enregistré les données annotées dans la base de données (DynamoDB) et les représenter à travers la table de bord (Dashboard).

Les résultats obtenus montrent l'efficacité de notre approche dans la réduction de la quantité des données envoyées au cloud, et par conséquent, l'amélioration de la performance globale du système.

## Perspectives

Au terme de ce travail qui peut être vu comme un début nous pouvons proposer un nombre de perspectives. Ces dernières peuvent venir compléter, améliorer notre travail. Parmi ces perspectives, nous pouvons citer :

- Enrichir notre modèle des règles pour prendre en charge d'autres aspects tels que la sécurité, la performance et la maintenance.
- Pour améliorer la qualité du processus d'annotation, nous nous concentrerons sur l'ajout des concepts provenants d'ontologies différentes, en tenant compte de la réduction de la taille de l'ontologie.
- Développer une interface graphique pour permettre aux développeurs de configurer le processus d'annotation en fonction d'un ensemble de paramètres et de règles personnalisés.

# BIBLIOGRAPHIE


[A13] Gyrard A. “An Architecture to Aggregate Heterogeneous and Semantic Sensed Data”. In : *Extended Semantic Web Conference, Springer* 7882 (2013), p. 697–701.

[ACH17] LAOUBI Lyes ACHAT ASALAS. “Conception et réalisation d’une application mobile cross- Platform pour l’Internet of things”. Mémoire de Master en Informatique. Université de Bejaia, juin 2017.

[Ad17] Mahmud Al-Osta AHMED BALI et Gherbi Ab DELOUAHED. “An Ontology-based Approach for IoT Data Pro cessing using Semantic Rules”. Thèse de doct. Ecole de technologie superieure, Montreal, Canada, 2017.

[Ahm09] Bali AHMED. “Evolution des Mappings suite à une Evolution d’Ontologie”. Mémoire de Magister en Informatique. Ecole Doctorale d’informatique de l’Est, Pôle Constantine, fév. 2009.

[ALO16] MAHMUD ABDALLA AL-OSTA. “IoT – MaaS : Modeling as a Service for Internet of Things Applications Development”. Doctoral Oral Exam DGA 1033 Report. UNIVERSITÉ DU QUÉBEC, août 2016.

[aws18] AWS. *AWS IoT*. Avr. 2018. URL : https://docs.aws.amazon.com/fr_fr/iot/latest/%20developerguide/what-is-aws-iot.html.

[Azo06] AZOUAOU.F. "Modèles et outils d'annotations pour une mémoire personnelle de l'enseignant". Thèse de doctorat. Université Joseph Fourier, 2006.

[Bil15] Benjamin BILLET. "Système de gestion de flux pour l'Internet des objets intelligents". Thèse de doct. Université de Versailles-Saint Quentin en Yvelines, juin 2015.

[Bra15] KHALDI BRAHIM. "Spécification d'ontologies dans les stratégies pédagogiques dédiées au e-learning". Mémoire de Magister en Informatique. UNIVERSITE DES SCIENCES ET DE LA TECHNOLOGIE d'ORAN Mohamed Boudiaf, fév. 2015.

[Che13] Sylvain CHERRIER. "Architecture et protocoles applicatifs pour la chorégraphie de services dans l'Internet des objets". Thèse de doct. UNIVERSITÉ PARIS-EST, nov. 2013.

[DC11] S. Guo D. ZENG et Z. CHENG. "The web of things : A survey". In : *Journal of Communications* 6.6 (2011), p. 424–438.

[E D02] C. JACQUIN E. DESMONTILS. "Annotations sur le Web : notes de lecture". In : *IRIN, Université de Nantes ; Journées Scientifiques Web Sémantique* (2002).

[eag17] EAGLE.IO. *eagle.io documentation*. 2017.

[Ele15] Aosong(Guangzhou) ELECTRONICS. *Temperature and humidity module AM2302 Product Manual*. Co.,Ltd., 2015.

[Eva15] D. EVANS. "The internet of things : How the next evolution of the internet is changing everything." Thèse de doct. CISCO, 2015.

[Gaë02] Lortal GAËLLE. “État de l’art Ontologies et Intégration/Fusion d’ontologies”. In : *laboratoire Dialogue et Intermédiations Intelligentes de la Direction des Interactions Humaines* (2002).

[goo18] Team GOOGLE. *Pricing*. Avr. 2018. URL : https://cloud.google.com/pricing/.

[Gra11] Peter Mell Timothy GRANCE. “The NIST Definition of Cloud Computing”. In : *NIST Special Publication 800-145* (2011).

[Gro18] GROMGULL. *RDFLib*. Mai 2018. URL : http://librdf.org/.

[Hac06] Hakim HACID. “Annotation Semi-automatique de Grandes BD”. Thèse de doct. France : Université Lumière Lyon 2, 2006.

[HZ13] M. HAN et H. ZHANG. “Business intelligence architecture based on internet of things”. In : *Journal of Theoretical & Applied Information Technology* 50.1 (2013), p. 90–95.

[Ilh12] Boukhalfa ILHEM. “Annotations collaboratives des documents pédagogiques”. Master II Génie logiciel. Université Ferhat Abbas de Sétif, 2012.

[Lad18] LADYADA. *Adafruit Python DHT*. Mai 2018. URL : https://github.com/adafruit/Adafruit_Python_DHT.

[LDL18] LDLC. *Raspberry Pi 3 Model B*. Mai 2018. URL : https://www.ldlc.com/fiche/PB00205573.html.

[Lji04] Stojanovic LJILJANA. “Methods and Tools for Ontology Evolution”. Ph.D thesis. University of Karlsruhe, 2004.

[ND06] R. Marcel N. DANIEL et K. DANIEL. “Livre blanc Machine To Machine enjeux et perspectives : Orange Business Services”. In : *Syntec informatique* (2006), p. 60.

[Obj18] OBJETCONNECTE.COM. *Comparatif des plateformes IoT*. Avr. 2018. URL : https://www.objetconnecte.com/comparatif-plateforme-iot/.

[Pie08] Françoise Massit-Folea PIERRE-JEAN BENGHOZI Sylvain Bureau. "L'Internet des objets. Quels enjeux pour les Européens ?" Thèse de doct. Ecole polytechnique et TELECOM Paris Tech, 2008.

[Pyt18] PYTHON. *Langage Python*. Mai 2018. URL : https://www.python.org/.

[Rad89] Bicknell E. et Blettner M. RADA R. Mili H. "Developement and Application of a Metric on Semantic Net". In : *IEEE Transactions on Systems, Man and Cybernetics* 19.1 (1989), p. 17–30.

[Ras18] RASPBERRY. *Raspberry PI3*. Mai 2018. URL : https://www.raspberrypi.org.

[SAH16] Somia SAHRAOUI. "Mécanismes de sécurité pour l'intégration des RCSFs à l'IoT (Internet of Things)". Thèse de doct. Université de Batna 2, nov. 2016.

[SIH09] KLAI SIHEM. "Construction d'une ontologie a partir de bases de donnees pour l'aide a la maintenance industrielle application : turbine a vapeur". Mémoire de Magister en Informatique. Université 20 août 1955, Skikda, 2009.

[Sta08] J. A. STANKOVIC. "Wireless sensor networks". In : *IEEE Computer Society* 41.10 (2008), p. 92–95.

[Tec18] TECHOPEDIA. *What is the Web of Things (WoT) ? - Definition from Techopedia*. Avr. 2018. URL : https://www.techopedia.com/definition/26834/web-of-things-wot.

[Usc96] M USCHOLD M. ; Grüninger. "Ontologies Principles Methods and Applications". In : *In Knowledge Engineering Review* 11.2 (1996), p. 93–155.

[Vun18] Radu VUNVULEA. *What Cloud Platform Should You Pick for IoT ? A Head to Head Between AWS and Azure*. Avr. 2018. URL : https://medium.com/the-why-and-how/what-cloud-platform-should-you-pick-for-iot-a-head-to-head-between-aws-and-azure-43cf8918fea6.

[wik18] Internet of things WIKI. *Top 20 IoT Platforms in 2018*. Avr. 2018. URL : https://internetofthingswiki.com/top-20-iot-platforms/634/.