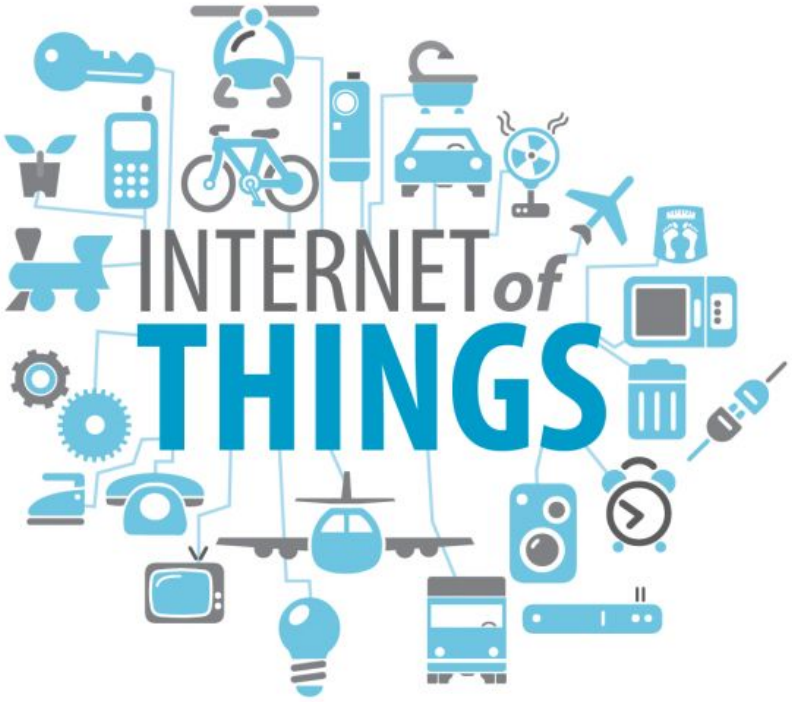
INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES TOULOUSE

Intergiciel pour l'Internet des Objets

Rapport de TP – 5 ISS INSA Toulouse

Noms	:	**ALVAREZ**	**DIOP**
Prénoms	:	**Josué**	**Mame Aminata**
Groupe	:	A1	
Date	:	Janvier 2017	



1. Savoir positionner les standards principaux de l'Internet des Objets

Le principe de oneM2M est d'offrir une couche de service permettant de rendre interopérable plusieurs applications, capteurs, actionneurs et de les connecter entre eux. Cette couche de service doit fournir une interface standardisée permettant aux développeurs de s'abstraire des protocoles de communications des objets utilisés, et du format des données échangées par ces capteurs et actionneurs.

Alors que certains standards M2M définissent des manières de communiquer entre objets connectés (MQTT, Google Thread), ou de gérer un parc d'appareils (OMA DM), OM2M propose une approche plus fédératrice qui permet l'intégration d'objets de nature quelconque au sein d'une couche de service de haut niveau.

2. Déployer une architecture conforme à un standard et mettre en place un système de réseau de capteurs aux services

2.1. Déployer et configurer une architecture IoT en utilisant OM2M

2.1.1. Architecture générale

Pour déployer une architecture IoT avec OM2M, il faut tout d'abord créer un nœud d'infrastructure (**IN**), qui sera au sommet de la hiérarchie. A ce nœud d'infrastructure peuvent s'enregistrer des middle-nodes (**MN**) regroupant des nœuds d'application (AN pour Application Node). Les **MN** peuvent être indifféremment installés sur la même machine physique / virtuelle que le nœud infrastructure, ou sur une autre machine. Ces nœuds **MN** sont souvent installés sur des machines appelées Gateways, qui peuvent fonctionner avec des ressources limitées, et qui communiquent directement avec plusieurs capteurs / actionneurs (via des IPE ou non). La figure 1 illustre ce principe de hiérarchie IN/MN/AN.

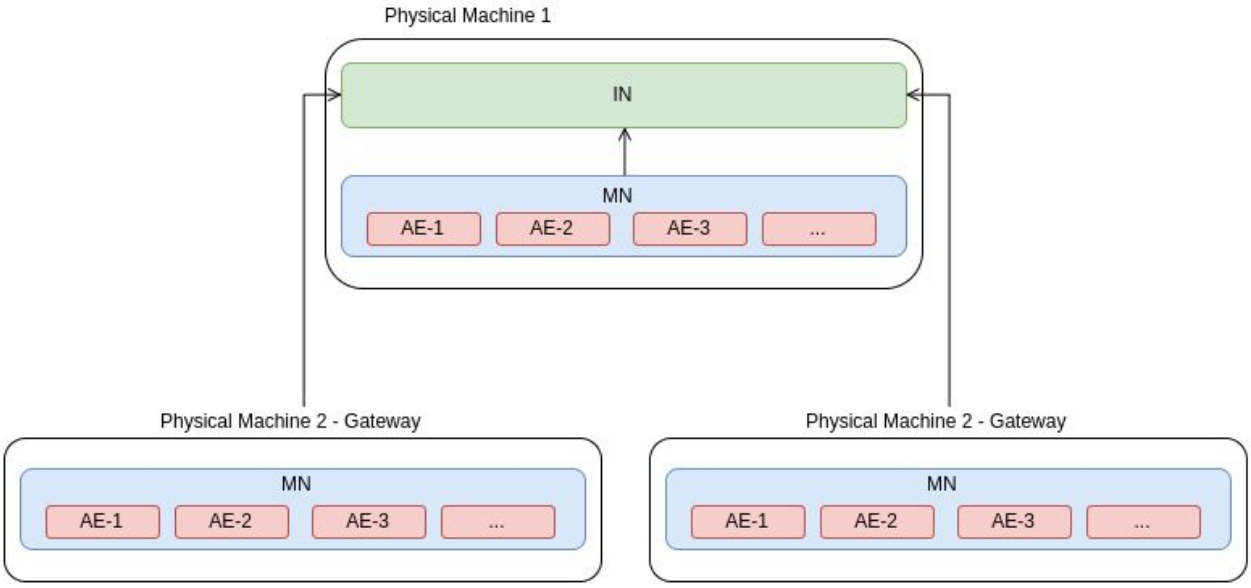


Figure 1. Illustration de l'architecture OM2M avec un IN et plusieurs MN.

Il découle de cette hiérarchie une structure arborescente où chaque nœud contient plusieurs autres nœuds de nature donnée. La première étape du déploiement consiste donc naturellement à déployer un nœud IN et plusieurs nœuds MN qui doivent s'enregistrer sur le nœud IN.

2.1.2. Interactions avec les capteurs et actionneurs

C'est par le biais des ressources Application Entity (AE) que les capteurs et actionneurs interagissent avec la plateforme. Ces derniers peuvent envoyer et recevoir des données via une API REST, fournie par les noeuds MN avec lesquels ils communiquent.

2.2. Interagir avec les objets en utilisant une architecture REST

2.2.1. Ressources OM2M

Les **ressources** OM2M sont des entités virtuelles qui vivent au sein des noeuds IN et MN de la plateforme. Ces ressources sont organisées de manière arborescente. Les noeuds IN et MN ont respectivement une ressource **IN-CSE** ou **MN-CSE** (Service Common Entity) à la racine de cette arborescence.

Pour les noeuds IN, les noeuds MN enregistrés sont représentés par des ressources MN-CSE. Ces ressources MN-CSE ont aussi un “pointeur” vers le noeud IN-CSE parent.

Les deux principaux types de ressources présents dans un MN-CSE sont les **ACP** (Access Control Policy) et les **AE** (Application Entity).

2.2.1.a. Contrôle d’accès via les ACP.

Une des propriétés communes des nœuds **MN-CSE** et **AE** sont les ACPI qui contrôlent les droits d’accès aux ressources liées à un noeud. Les ACPI (Access Control Policy Identifier) sont des liens vers des nœuds ACP qui contiennent les droits d’accès au nœud (*privileges*) et à l’ACP lui-même (*self-privileges*).

Lorsque l’on veut créer une plateforme multi-utilisateurs (administrateurs, capteurs, monitors, etc.), il faut être capable de limiter l’accès de chaque utilisateur à certaines ressources et de limiter la gestion des ACP aux administrateurs uniquement.

Pour cela on peut créer des nœuds ACP qui vont contenir des couples utilisateurs/droits d’accès (création, lecture, modification, suppression, notification, découverte) et les associer à d’autres nœuds.

2.2.2.b Structure d’une application

Une fois les nœuds IN/MN déployés, il est nécessaire de déployer les applications qui vont gérer les interactions avec les objets. Ces applications suivent une structure particulière, comme le montre la figure 2.

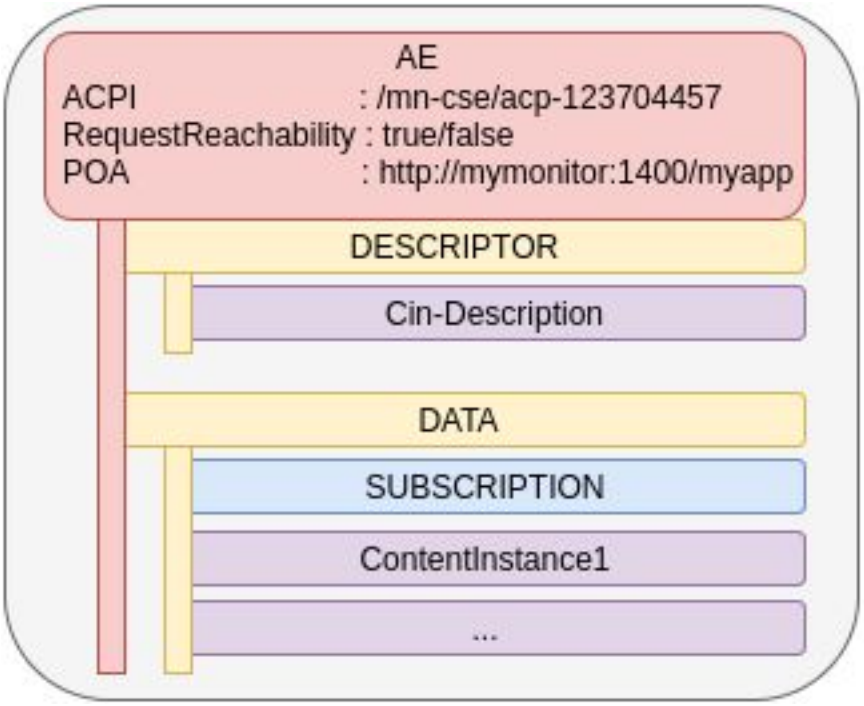


Figure 2. Structure d'une AE.

Le nœud **AE** possède deux nœuds de type « **container** » (en jaune), possédant chacun des nœuds de type « **content instance** » (en violet).

Le nœud **DESCRIPTOR** contient une content instance qui contient des métadonnées concernant les données de l'application (lieu, unité des données, type de capteur, etc.) ainsi que des potentielles opérations pouvant être transmises au capteur/actionneur (configuration du capteur, changement d'état de l'actionneur, demande de transmission d'information depuis le capteur etc.).

Le nœud **DATA** quand à lui contient principalement deux types de nœuds :

- **Content-instance** : les données remontées depuis le capteurs, possiblement agrémentées de métadonnées.

- **Subscription** : ce nœud permet d'envoyer une notification à une ou plusieurs URIs spécifiées dans le nœud subscription (« notificationURIs ») à chaque fois qu'une nouvelle donnée est ajoutée dans le nœud DATA où est présente la souscription. Il est possible de choisir de notifier directement vers l'URL du serveur de monitoring, mais cela n'est pas recommandé. Il est préférable de créer une AE qui contiendra un POA vers le serveur de monitoring, et de mettre l'URL de l'AE dans les champ « notificationURI » du nœud subscription. La figure 3 présente de manière schématique le procédé de souscription.

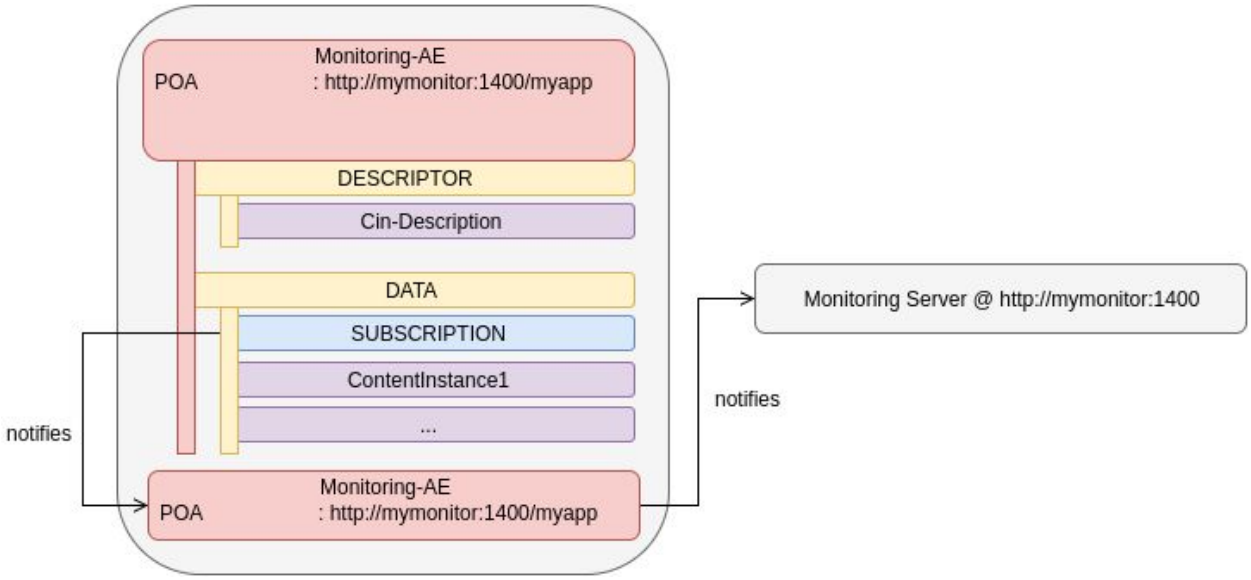


Figure 3. Représentation schématique du procédé de souscription

2.2.1. Interactions avec les ressources.

Il est possible de créer, supprimer, modifier toutes les ressources présentées ci-dessus. Pour cela, la plateforme expose une API REST qui permet de manipuler ces ressources. Cette API s'utilise par le biais d'un client HTTP, qui peut être une application graphique, comme Postman (utilisé pour du debugging) ou un programme qui automatise la manipulation des ressources.

Les souscriptions, quand à elles, nécessitent l'utilisation d'un serveur HTTP pour recevoir les notifications, et effectuer les traitements nécessaires.

2.3. Intégrer une nouvelle / autre technologie dans une architecture IoT

Afin d'intégrer une nouvelle technologie dans une architecture IoT, il est nécessaire de développer un Interworking Proxy Entity (IPE). Cet IPE se charge de faire le lien entre un capteur/actionneur et la plateforme OM2M.
Le principe est simple : le capteur/actionneur communique avec son propre protocole avec l'IPE, et l'IPE communique avec OM2M de façon standard. La communication peut prendre 2 directions.

2.3.1. Communication de OM2M vers le capteur/actionneur

Dans ce cas, les actions à effectuer sont décrites dans un noeud de type Content Instance présent dans le noeud container "DESCRIPTOR", sous la forme de requêtes HTTP, qui seront envoyés à destination d'une AE, qui les redirigera (grâce à un poa) vers l'IPE sur lequel fonctionne un serveur HTTP. Ce serveur réceptionne les requêtes envoyées par la plateforme OM2M, et envoie les ordres correspondant aux requêtes vers le capteur/actionneur, en utilisant le protocole spécifique à ce dernier.

Attribute	Value
location	Home
type	AMB_LAMP
unit	
GET	/mn-cse/mn-name/LAMP_2?op=get
ON	/mn-cse/mn-name/LAMP_2?on=true&bri=254&sat=254&id=LAMP_2
OFF	/mn-cse/mn-name/LAMP_2?on=false&id=LAMP_2
RED	/mn-cse/mn-name/LAMP_2?on=true&bri=254&sat=254&hue=0&id=LAMP_2
GREEN	/mn-cse/mn-name/LAMP_2?on=true&bri=254&sat=254&hue=36210&id=LAMP_2

Figure 4. Exemple d'un content instance de DESCRIPTOR ayant plusieurs actions associées.

2.3.1. Communication du capteur/actionneur vers OM2M

Dans ce cas, le capteur envoie une notification à l'IPE, sur lequel est installé un programme en écoute, utilisant le protocole spécifique du capteur. L'IPE envoie ensuite un noeud de type Content Instance dans le container "DATA" de l'application correspondant au capteur, comprenant les données brutes envoyées par le capteur, ainsi que de potentielles métadonnées qui permettent d'enrichir ces données brutes et de leur donner un sens.

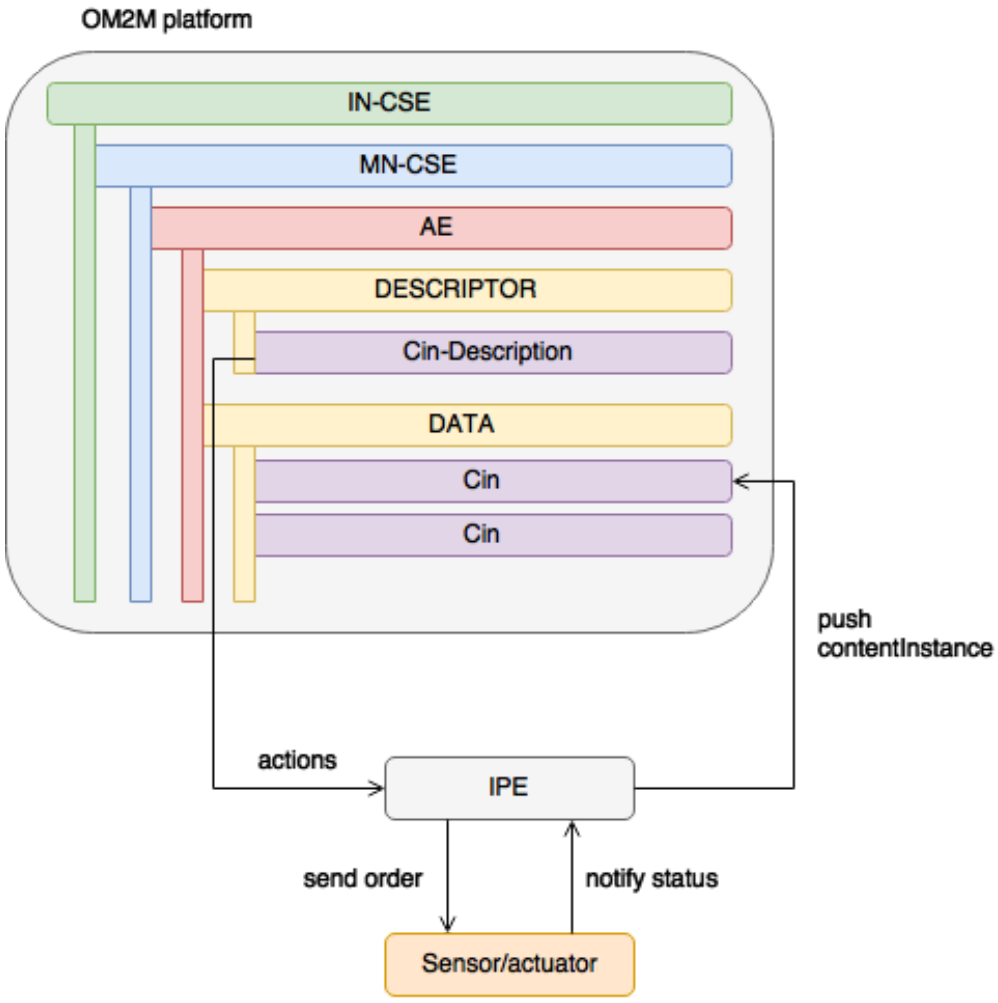


Figure 5. Illustration du procédé de communication entre OM2M et un capteur/actionneur par le biais d'un IPE.