PROJET CASSIOPÉE
2021/2022

Rapport de synthèse

Introduce IoT device management into the platform

Camille FAUCHIER
Maxence DURIEU
Rali BENJELLOUN

Tuteurs : Tiphaine HENRY, Walid GAALOUL

Table des matières

1. Introduction

Le développement de l'internet des objets (IoT) a permis d'améliorer la qualité de production en offrant un suivi complet de toute la chaîne de production. Cependant, assurer la sécurité et la confidentialité dans le contexte de l'IoT est une tâche complexe en raison de la faible capacité de ces dispositifs connectés. En général, la gestion des dispositifs IoT repose sur une entité centralisée qui valide les droits de communication et de connexion. Par conséquent, cette entité centralisée peut être considérée comme un point de défaillance. En effet, la faible capacité de calcul des dispositifs IoT peut entraîner des failles de sécurité et de confidentialité dans les systèmes IoT. Ainsi, certains acteurs peuvent violer la vie privée des propriétaires de données en compromettant les dispositifs IoT existants pour obtenir des informations illégales.
L'utilisation de la technologie blockchain pourrait pallier ce problème. La blockchain représente une base de données décentralisée qui stocke les informations en toute sécurité dans des blocs de données immuables. En ce qui concerne la gestion de la chaîne d'approvisionnement, ces caractéristiques offrent des possibilités d'accroître la transparence, la visibilité, l'automatisation et l'efficacité de la chaîne d'approvisionnement.

2. Présentation du projet et des objectifs

2.1. Présentation du sujet

Notre projet consiste à configurer des objets IoT dans un processus métier (BP) en utilisant la blockchain afin de renforcer le contrôle du propriétaire des données sur ses propres objets IoT et améliorer les chaînes d'approvisionnement. Pour ce faire, il a fallu intégrer les objets IoT dans le processus métier (BP) par l'intermédiaire de Smart Contract.

Définissons tout d'abord les différents concepts :

- Les objets IoT sont des objets connectés, capables de communiquer entre eux. Selon Cisco, le nombre d'appareils connectés devrait atteindre 500 milliards d'ici 2030. Une telle augmentation améliorera sans aucun doute la qualité de vie des gens en leur fournissant de meilleures facilités pour diverses applications quotidiennes. Cependant, ces capteurs, ces montres connectés, etc. doivent être respectueuses de notre vie privée et être sécurisées. La sécurité et la fiabilité de ces objets sont donc des points de vigilance de cette technologie.

- Un Smart contrat est l'équivalent informatique dans une blockchain d'un contrat traditionnel. Il exécute automatiquement des conditions définies au préalable et inscrites dans une blockchain. La blockchain peut être assimilée à un grand registre distribué dans le réseau. Des mécanismes de consensus et de cryptographie garantissent l'intégrité des informations inscrites sans intermédiaire ou organisme central. Sur une blockchain, nous ne pouvons pas effacer des informations, seulement en rajouter.

- Le BP ou processus métier est une démarche qui sert à analyser et fluidifier les processus mis en place par des entreprises pour réaliser leurs activités et optimiser leur performance. Un processus consiste en une série de tâches et d'actions, certaines nécessitent des intermédiaires, des collaborateurs, le but final étant d'obtenir un résultat déterminé (par exemple : suivi des demandes d'achats). Les entreprises sont demandeuses d'intégrer des objets IoT dans le Business model, cependant elles restent hésitantes vis à vis de problématiques telles que la sécurisation des données.

2.2. Présentation des objectifs

Effectuer cette intégration via un Smart Contract permettrait de pallier aux problèmes évoqués ci-dessus car dans la blockchain, la gestion est entièrement décentralisée et transparente assurant un stockage fiable des données ainsi que des scripts fiables. L'autonomisation des processus permet ainsi un gain en efficacité ainsi qu'en termes de collaboration. Cet aspect permet de faciliter la résolution de conflits et l'audit. De ce fait, notre objectif principal est d'exécuter les processus métier avec un smart contract.

Pour cela, notre but est de permettre la configuration d'objets IoT dans un graphe DCR (Dynamic Control Response) déployé dans un smart contract.

Voici quelques points intermédiaires afin d'atteindre notre objectif :

- Créer des opérateurs d'allocation et paramétrage d'objets IoT propres au langage DCR ;
- Définir des droits d'accès aux données: lecture, écriture (créer, modifier, supprimer) ;
- Faire à terme ce paramétrage depuis le smart contract ;
- Enregistrer les objets IoT dans la plateforme en tant que NFT ;
- Créer des activités publiques gérées par des objets IoT ;
- Intégrer ces activités dans la plateforme.

3. Modélisation

Nous avons pendant cette étape chercher à imiter le rôle d'objets IoT en tant que sensors et actuators en utilisant plusieurs interfaces. Les objectifs de la phase de modélisation sont donc les suivants :

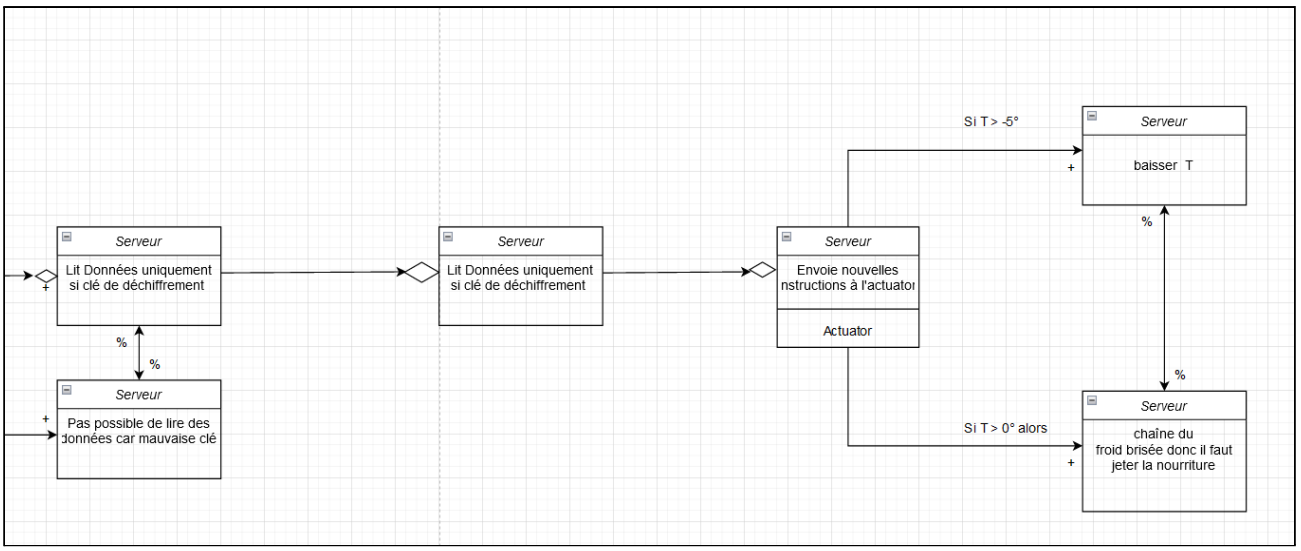
- Comprendre le fonctionnement et la logique des diagrammes DCR appliqué à notre cas d'étude ;
- Modéliser le fonctionnement d'un objet IoT avec un diagramme UML ;
- Permettre la configuration d'objets IoT dans un graphe DCR déployé dans un smart contract ;

3.1. Les graphes DCR

Les graphes DCR représentent les relations entre le workflow (flux evenementiel), les acteurs, les activités et les contraintes. Le modèle de base de DCR Graphs se compose d'un ensemble d'événements et de cinq relations binaires :

Relation Milestone	A →<> B	L'exécution de A doit être terminée pour que B puisse être exécutée.
Relation Response	A •→ B	Lorsque l'exécution de A est terminée, l'exécution de B commence
Relation Include	A →+ B	Si A est exécutée, alors B peut être exécutée
Relation de condition	A →• B	L'exécution de A doit être commencée avant que B puisse être exécutée
Relation d'exclusion	A →% B	Si A est executée alors B ne peut pas être executée

Lorsque que l'événement représente une interaction entre deux acteurs, c'est une **tâche chorégraphique**. Pour représenter une contrainte, nous insérons la condition au niveau de la relation. Pour mieux comprendre voici l'exemple d'une livraison de produits dont la température doit être contrôlée et une partie du diagramme DCR associé :

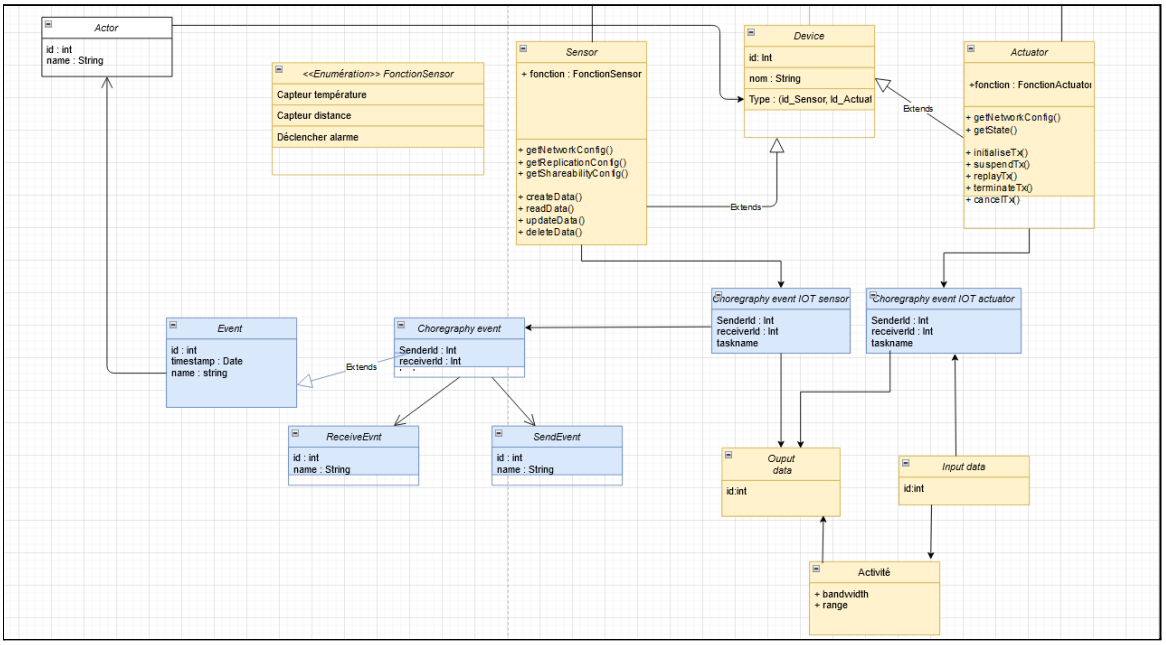


Le capteur température relève des données dans son environnement et les envoie au serveur. On doit alors se poser la question des droits de lecture : qui a accès à la température relevée ? L'entreprise vendant le produit ? Le transporteur ? Le client ? Ces droits sont définis dans le smart contract. Dans le diagramme DCR, nous modélisons ces relations, traduites par la suite dans le smart contract.

3.2. Diagramme de classe UML

Le diagramme de classe UML est un schéma utilisé pour présenter les classes et les interfaces des systèmes ainsi que leurs relations. Une classe décrit les responsabilités, le comportement et le type d'un ensemble d'objets.

Le langage UML nous a ici permis de décrire la structure de notre système en modélisant un type d'objet IoT, son fonctionnement et l'utilisation de la blockchain vis à vis de celui-ci. C'est un travail nécessaire pour pouvoir ensuite implémenter les objets IoT à l'application.



3.2. Les matrices de contrôle d’accès

Comme évoqué précédemment, l’une des problématiques qui subsistent quant à l’utilisation des objets IoT est son accès. Qui peut lire les informations relevées par des capteurs ? Qui peut créer, modifier ou supprimer un objet dans une chaîne d’approvisionnement ? La matrice contrôle d’accès sert à établir des règles d’accès sur les différents objets et datas en fonction du workflow et des acteurs.. Une fois la matrice contrôle d’accès établie (voir ci-dessous), les règles d’accès doivent être implémentées dans le smart contract. Voici un exemple de matrice d’accès :

	Measure T (c)		Compute L	Activity3 (c)	activity4
Donnee 1 (temp)	Sensor T	serveur	Serveur	Role 2, role3	Rôle 2
Read		x	x		
Write	x				
Create	x				
Delete					
Privacy requ.		F	F		

Par exemple, pour le premier cas, on a une tâche chorégraphique entre un capteur de température et un serveur : le capteur peut créer et écrire la mesure T sur le serveur et le serveur peut lire cette donnée.

4. Développement

Les objectifs de cette partie sont les suivants :

- Intégrer les objets IoT dans la plateforme (enregistrement d'un objet IoT comme NFT, et association de l'objet IoT à un champ de données, configuration) ;
- Créer des activités publiques gérées par des objets IoT ;
- Intégrer ces activités dans la plateforme.

4.1. Quelques technologies utilisées dans notre projet

IPFS (Interplanitary File System) *:* IPFS est un système distribué permettant de stocker et d'accéder à des fichiers, des sites Web, des applications et des données.

Ethereum *:* Ethereum est un protocole Blockchain hébergé sur de nombreux ordinateurs à travers le monde, la rendant ainsi totalement décentralisée. Chaque ordinateur dispose d'une copie de la Blockchain et il doit obligatoirement y avoir un accord majoritaire avant que le moindre changement ne puisse être implémenté sur le réseau. C'est dans ce réseau que sont implémentés les scripts de code que l'on appelle les Smart Contracts.

Python et le Framework Flask *:* Le Framework Flask fonctionne sous le langage open source Python. Il permet de très facilement déployer une API (Interface de programmation d'application) c'est-à-dire une interface logicielle qui permet de « connecter » un logiciel ou un service à un autre logiciel ou service afin d'échanger des données et des fonctionnalités

Remix : Remix est une interface permettant de développer, compiler, déployer et tester des smart-contracts Ethereum. Autrement dit c'est un outil de déploiement sur une blockchain utilisant différents moyens.

Solidity : Solidity est un langage de programmation, de type orienté objet. Il permet le développement de smart-contrat sur la blockchain Ethereum.

4.2. Application pré-existante

Il existe déjà une plateforme de gestion de processus métiers dont les tâches publiques sont exécutées sur la blockchain Ethereum. Cette plateforme contient déjà notamment le smart contract qui gère les rôles des acteurs.

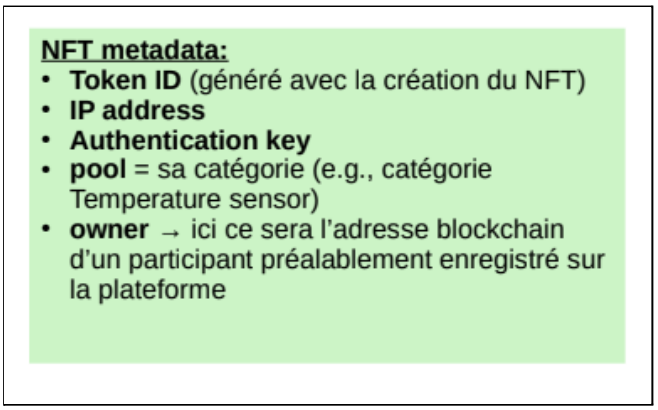
4.3. Implémentation des objets IoT

Dans le contexte de la chaîne d'approvisionnement, il est nécessaire de relier les objets IoT à des identifiants uniques avec des “profils numériques" sur la blockchain. C’est là que la notion de *Jeton non fongible* (NFT) intervient. Chaque unité d'un NFT est unique par rapport à une autre, ce qui permet de suivre leur propriété. Cela fait de l'adoption des NFT un concept approprié pour cartographier les objets IoT tout au long des chaînes d'approvisionnement sur la blockchain.

Pour cela nous utilisons la norme ERC-721, une API pour les NFT des smart-contracts. Si un smart-contract implémente les propriétés requises, il peut être nommé NFT ERC-721 et, une fois déployé, sera responsable d'effectuer un suivi des jetons créés sur Ethereum. Parmi ces propriétés figurent :

- Chaque jeton ERC-721 a un **nom**. Ce champ permet d'indiquer aux contrats et aux applications externes le nom du token.
- Le smart contract a un champ défini appelé **Propriétaire**, ce qui permet de garantir la non-fongibilité du token et de l'identifier cryptographiquement.
- Le smart contract a un champ appelé **Approbation**, par lequel l'autorisation est accordée à une autre entité de transférer le jeton au nom du propriétaire.
- Les jetons ERC-721 ont un champ appelé **métadonnées du jeton.** C'est précisément ce champ qui permet son statut non fongible et abrite toutes ces propriétés qui distinguent un token de toutes les autres.

Les métadonnées de l’objets IoT sont les suivantes :



Pour implémenter ces contrats NFT, nous avons fork un projet GitHub d’OpenZeppelin. C'est-à-dire que nous avons récupéré un projet NFT déjà existant pour avoir le modèle de notre contrat ERC 721.

Ce contrat permet pour le moment de :

- créer un nouvel objet IoT ;
- supprimer un objet IoT.

4.4. Implémentation des pools d’objets IoT

Chaque objet IoT appartient à une catégorie (pool) d’objet. Il faut donc également développer un smart contract qui implémente ces pools. Pour le moment, nous avons repris et adapté le contrat

qui gérait les rôles.

5. Conclusion

Nous avons pu modéliser le fonctionnement et la configuration d'objets IoT au sein d'une chaîne d'approvisionnement avec des diagrammes DCR et UML. Cette étape nous a permis de comprendre les différentes relations entre les objets, les acteurs et les activités afin de pouvoir les déployer dans un smart contract. Nous avons pu par la suite procéder au développement et donc d'intégrer les objets IoT au sein de la plateforme. Pour ce faire, il a fallu enregistrer les objets IoT en tant que NFT et les configurer. Pour approfondir notre sujet, il aurait fallu implémenter les activités des objets IoT au sein de la plateforme.
Ce projet fut très instructif car il nous a permis de découvrir de nouvelles technologies telles que le langage DCR, les rouages de la blockchain ou encore l'implémentation de smart contracts. Il nous a également permis de nous rendre compte des différents problèmes liés à l'utilisation d'objets IoT et des solutions envisagées.

6. Remerciements

Nous souhaitons remercier chaleureusement Joséphine Kohlenberg, le projet Cassiopée, nos tuteurs encadrants (Tiphaine Henry, Walid Gaaloul) pour leur disponibilité et leur accompagnement ainsi que les étudiants organisateurs du Cassiopée pour avoir permis de relever ce challenge académique alliant technique et gestion de projet.