REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR
ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE Dr YAHIA FARES de MEDEA



FACULTE DES SCIENCES

DEPARTEMENT DE MATHEMATIQUES ET INFORMATIQUE

Mémoire de Fin d'Etudes de Master

Filière : Informatique

Spécialité : Ingénierie des Systèmes et Technologie du Web

Thème

Proposition d'une méthodologie de conception et de développement des jumeaux numériques pour les distributeurs automatiques de produits

Proposé et Dirigé Par :

Mr. RAHIM Messaoud

Réalisé Par :

BENKORTBI Asmaa
KADIK Maroua

Membres de jury :

Mr. MAZARI Redha
Mr. RAFAA Taher

Année Universitaire 2019/2020

Dédicace

Je dédie ma contribution dans ce travail

A mes chers parents pour leur soutien indéfectible dans les moments les plus difficiles

A mes deux frères Houssem Eddine et Ahmed Yacine

Aux familles Kadik et Skendri

A mon amies qui a contribué à la réalisation de ce travail : Asmaa

A toutes mes amies proches et lointaines

A mes enseignants qui ont éclairé mon parcours d'apprentissage

Maroua

Dédicace

À mes proches que j'aime et qui m'encouragent

À mon fils Mohamed Amine

À mes parents, mon mari Yacine, mes frères et sœurs

À ma famille Benkortbi

À ma belle famille Ramoul

À tous ceux qui me sont chers

A à ma très chère amie : Maroua

A toute la promotion 2019-2020

Asmaa

Remerciements

*T*ous d'abord nous remercions **ALLAH** de nous avoir guidé et donné le courage et la force d'accomplir ce modeste travail.

*N*ous adressons nos profonds remerciements à notre promoteur et encadreur Mr. **RAHIM Messaoud,** Enseignant chercheur à l'université de Médéa pour sa patience, encouragement et ses précieuses orientations. Son œil critique nous a été très précieux pour structurer ce travail et améliorer sa qualité.

*N*os remerciements vont à Mr **Cheblaoui Mohamed lamine** pour avoir eu la patience de répondre à nos innombrables questions.

*N*otre respect aux membres du jury qui ont bien voulu accepter de juger ce modeste travail, d'apporter leurs réflexions et leurs critiques scientifiques.

A la fin, nous aimerions exprimer notre gratitude a tous les employeurs et administratifs de notre université **Dr. YAHIA FARES.**

A toutes les personnes qui nous ont encouragé et soutenu de prés ou du loin durant la réalisation de ce travail.

Résumé

Un jumeau numérique de système est une représentation virtuelle de système réel. Notre travail est de développer des jumeaux numériques de système de distributeur automatique au but de simuler, prédire et optimiser le produit et le système de production. L'implémentation de ce jumeau numérique se fait via internet of things IOT en basant sur l'ingénierie dirigée par les modèles. nous utilisons le langage SysML pour la modélisation et la plateforme (Eclipse Ditto) pour la création des jumeaux numériques. Nous avons développé une application web permettant de capter et simuler les produits et le système de production.

Mots clés: Jumeau numérique, ingénierie des modèles, système de distribution automatique, diagramme de blocs, SysML, eclipse ditto, internet des objets.

Abstract

A digital system twin is a virtual representation of a real system. Our study is to develop digital twins for the vending system to simulate, predict and optimize the product and the production system. The implementation of this digital twin is done via internet of things IOT based on model engineering. We use the SysML language for modelling and the platform (Eclipse Ditto) for the creation of digital twins. We have developed a web application allowing capturing, simulating, predicting and optimizing the products and the production system.

Key words: Digital twin, model engineering, automatic distribution system, block diagram, SysML, ecipse ditto, internet of thing.

ملخص

توأم النظام الرقمي هو تمثيل افتراضي لنظام حقيقي. تتمثل مهمتنا في تطوير توائم رقمية لنظام البيع لمحاكاة المنتج ونظام الإنتاج والتنبؤ بهما وتحسينهما. يتم تنفيذ هذا التوأم الرقمي عبر إنترنت الأشياء IOT على أساس الهندسة التي يقودها النموذج. نستخدم لغة SysML للنمذجة والمنصة (Eclipse Ditto) لإنشاء توائم رقمية. لقد قمنا بتطوير تطبيق ويب يسمح بالتقاط ومحاكاة والتنبؤ وتحسين المنتجات ونظام الإنتاج.

الكلمات المفتاحية: توأم رقمي ، هندسة نموذجية ، نظام توزيع آلي، مخطط تخطيطي ، سيسامال، ايكليبس ديتو ، إنترنت الأشياء.

Tableau	Titre	Page
Chapitre I : SYSML et DIGITAL TWIN pour l'ingénierie des systèmes		
Figure I.1	Jumeaux numériques d'un équipement/système/installation	04
Figure I.2	Internet des objets	06
Figure I.3	Architecture de l'IOT	07
Figure I.4	Flux rouge nœud simulant des dispositifs IOT et testant la communication flux dans l'infrastructure du système	08
Figure I.5	Flux de protocole MQTT	09
Figure I.6	Relation entre UML et SYSML	12
Figure I.7	Type de diagramme SYSML	13
Figure I.8	Les quatre piliers du SYSML	14
Figure I.9	Exemple d'un bloc	16
Figure I.10	Relation de composition	16
Figure I.11	Relation de l'association	16
Figure I.12	Relation de l'agrégation	17
Figure I.13	Liens d'instanciation et de représentation entre un méta-modèle, un modèle et un original	17
Figure I.14	Architecture MDA	18
Figure I.15	Principe de transformation des modèles	19
Chapitre II : Distributeurs automatiques des produits		
Figure II.1	Présentation des colonnes simples/doubles des distributeurs des produits	23
Figure II.2	Anatomie d'un distributeur automatique des produits	23
Figure II.3	Anatomie interne d'un distributeur automatique des produits	24

Figure II.4	Les composants internes d'un distributeur automatique des produits	25
Chapitre III : Conception		
Figure III.1	Intégration avec le système des distributeurs automatiques des produits	29
Figure III.2	Architecture de l'application Web	29
Figure III.3	Modèle de Bloc des distributeurs automatiques des produits	31
Figure III.4	Méta-modèle des jumeaux numériques	31
Figure III.5	Relation entre Thing et feature	33
Figure III.6	Exemple de distinction entre les blocs	34
Figure III.7	Objet annoté par des commentaires	35
Figure III.8	Modélisation des droits d'accès	35
Figure III.9	Exemple de modélisation des droits d'accès sur un objet	36
Figure III.10	Etapes de transformation	37
Figure III.11	Modèle jumeau numérique d'un distributeur automatique des produits secs	40
Figure III.12	Fichier format JSON d'un objet	41
Figure III.13	Fichier format JSON d'une policy	42
Figure III.14	Diagramme de cas d'utilisation	44
Figure III.15	Diagramme de séquence de gestion d'application	46
Figure III.16	Diagramme de séquence de gestion de distributeur	47
Figure III.17	Diagramme de séquence de gestion de transformation	48
Figure III.18	Diagramme de séquence de suppression de distributeur	49
Figure III.19	Diagramme de séquence de consultation des distributeurs	50
Figure III.20	Diagramme de séquence de la recherche d'un produit	50
Figure III.21	Diagramme de séquence de la création d'un distributeur	51
Figure III.22	Diagramme de séquence de la modification du prix d'un produit	52

Chapitre IV : Réalisation et test

Figure IV.1	Annotation de Thing	55
Figure IV.2	Annotation de features	55
Figure IV.3	Annotation de policy	55
Figure IV.4	Règle de transformation des objets	56
Figure IV.5	Règle de transformation des fonctionnalités	56
Figure IV.6	Règle de transformation des policy	57
Figure IV.7	Règle ACCELEO pour transformation des droits d'accès	58
Figure IV.8	Règle ACCELEO pour transformation des objets	58
Figure IV.9	Création du projet Plugin	59
Figure IV.10	Conversion à fichier jar	60
Figure IV.11	Simulation des fonctionnalités du compartiment es produits	61
Figure IV.12	Simulation de l'ajout des produits auxiliaires	61
Figure IV.13	Simulation de l'ajout des produits livrés	62
Figure IV.14	Simulation de l'achat des produits	62
Figure IV.15	Interface pour commencer la gestion des distributeurs	63
Figure IV.16	Interface pour savoir plus sur le site	63
Figure IV.17	Interface de localisation des distributeurs	64
Figure IV.18	Affichage de distributeur des produits secs	64
Figure IV.19	Affichage de distributeur des produits liquides	65
Figure IV.20	Affichage des prix et quantités des produits	65
Figure IV.21	Cas d'un compartiment vide	65
Figure IV.22	Recherche des produits existants	66
Figure IV.23	Recherche des produits nonexistants	66
Figure IV. 24	Interface de page d'authentification	66
Figure IV. 25	Interface de page de transformation	67

Figure IV. 26	Interface lors de la transformation	67
Figure IV. 27	Interface de la création d'un distributeur	68
Figure IV. 28	Choix de localisation	68
Figure IV. 29	Interface de suppression d'un distributeur 1	69
Figure IV. 30	Interface de suppression d'un distributeur 2	69
Figure IV. 31	Interface de changement de prix 1	70
Figure IV. 32	Interface de changement de prix 2	70

Tableau	**Titre**	**Page**
Chapitre III : Conception		
Tableau III.1	Transformation d'objet	38
Tableau III.2	Transformation de fonctionnalités	39

LISTE DES ACRONYMES

ATL : Atlas Transformation Langage

BDD : Block Definition Diagram

CSS : Cascading Style Sheets

HTML : Hypertext Markup Language

IOT : Internet Des Objets

M2T : Model to Texte

M2M : Model to Model

MDA : MODEL DRIVEN ARCHITECTURE

MDE : MODEL DRIVEN ENGINEERING

PHP : Hypertext Preprocessor .

SYSML : Systems Modeling Language

UML : Unified Modeling Language

XML : EXtensible Markup Language

Sommaire

Introduction Générale 01

Chapitre I : SYSML et DIGITAL TWIN pour l'ingénierie des systèmes

I.1 Introduction 03

I.2 Jumeaux numérique 03

I.2.1 Définition 03

I.2.2 Les Jumeaux numériques et leurs valeurs propres 04

I.2.3 Les éléments technologiques clefs 04

I.2.4 Exemples d'usage 05

I.3 Internet des objets IOT 05

I.3.1 IOT : une présentation 06

I.3.2 Caractéristiques 06

I.3.3 IOT architecture 07

I.4 Node-red 08

I.4.1 Protocole MQTT 09

I.4.2 Flux MQTT très simple 09

I.5 Plateforme DITTO 09

I.5.1 Fonctionnalités 10

I.5.2 Quand utiliser DITTO 10

I.6 Les objectifs de développement de jumeaux numériques 10

I.7 SYSML 11

I.7.1 Relation entre UML et SYSML 12

I.7.2 Résumé du diagramme SYSML 12

I.7.3 Implémentation et exécution 14

I.8	Diagramme de Bloc	15
I.8.1	Un Bloc	15
I.8.2	Relation	16
I.9	Méta modélisation	17
I.10	Principe du modèle driven architecture (MDA)	18
I.11	Transformation des modèles	18
I.12	Modèle driven engineering (MDE)	19
I.13	Conclusion	20

Chapitre II : Distributeur automatique des produits

II.1	Introduction	21
II.2	Histoire de distributeur automatique	21
II.2.1	Naissance de distributeur automatique	21
II.2.2	Définition de distributeur automatique	21
II.3	Avantage et inconvénient des distributeurs automatiques	21
II.4	Présentation des produits secs	22
II.5	Anatomie d'un distributeur automatique des produits	23
II.6	Composants internes d'un distributeur automatique des produist	25
II.7	Les différents types de distributeurs automatiques des produits	26
II.8	Processus de mise en marche initiale	26
II.9	Sécurité	27
II.10	Conclusion	27

Chapitre III : Conception

III.1	Introduction	28
III.2	Idée Générale	28
III.3	Modélisation des systèmes des distributeurs automatiques des	30

	produits	
III.4	Méta modèle des jumeaux numériques	31
III. 4.1	THING	32
III. 4.2	Features	32
III.4.3	Policy	33
III.4.4	Distinction entre les blocs	34
III.4.5	Comment définir les droits d'accès	35
III.5	Application de transformation	36
III.5.1	Transformation modèle vers modèle	37
III.5.2	Transformation modèle vers texte	40
III.6	Application de supervision à travers les jumeaux numériques	42
III.6.1	Objectif de l'application	43
III.6.2	Diagramme de cas d'utilisation	43
III.6.3	Diagramme de séquence	45
III.6.4	Intégration avec le système des distributeurs automatiques des produits	55
III.7	Conclusion	53

Chapitre IV : Réalisation et test

IV.1	Introduction	54
IV.2	Implémentation	54
IV.2.1	Modélisation	54
IV.2.2	Transformation	56
IV.2.2.1	Transformation ATL	56
IV.2.2.2	Transformation ACCELEO	57
IV.2.3	Intégration des transformations	58
IV.2.4	Platforme des jumeaux numériques utilisés	60

IV.3	Etudes des cas	60
IV.3.1	Simulation des systèmes des distributeurs automatiques des produits	61
IV.3.2	Interfaces de l'application Web	62
IV.3.2.1	Interface d'accueil	63
IV.3.2.2	Interface D'authentification	66
IV.3.2.3	Interface de transformation	67
IV.3.2.4	Interface de création des distributeurs	68
IV.3.2.5	Interface de suppression des distributeurs	69
IV.3.2.6	Interface de changement des prix	69
IV.4	Conclusion	70
Conclusion Générale		71
Références bibliographiques		73

INTRODUCTION GENERALE

INTRODUCTION GENERALE

Il est indéniable que l'informatique fait maintenant partie intégrante de notre vie privée et professionnelle. Aujourd'hui elle est partout présente, que ce soit dans les magasins par le paiement électronique et le stockage des informations, ou que ce soit également par le paiement par distributeurs automatiques, ou en passant par les divers services qui demandent paiement électronique comme par exemple les stations-services, les billets électroniques et les réservations qui se font en ligne. Ces exemples ne sont qu'une partie infime de la présence toujours grandissante de l'informatique, du mail à la conquête de l'espace.

Un système est un assemblage, une collection organisée (possédant une structure) d'objets reliés ou branchés (en interrelation) les uns aux autres, de façon à former une entité ou un tout remplissant une ou plusieurs fonctions.

Le jumeau numérique est la copie digitale d'un processus. Cette copie 'à l'identique' va détecter des problèmes, sur un appareil, sans le toucher. C'est la version virtuelle programmée sur ordinateur, de quelque chose qui existe dans le monde réel. Selon le bureau d'études Gartner, cette technologie du "digital twin", figure parmi les dix technologies qui vont bouleverser notre vie.

Nous avons pris comme thème de notre travail la création d'une image numérique des distributeurs automatique des produits. La question qui a accompagné notre recherche pourrait être posée comme suit : Comment gérer plusieurs distributeurs automatiques de produits d'une entreprise?

Pour répondre à cette question, nous avons projeté de mettre en place une application web en mesure de développer un jumeau numérique pour les distributeurs automatiques au but d'aider la gestion automatique d'une entreprise de ses distributeurs automatiques de produits. Tout cela en vue de faciliter de capter et actionner comme nous l'avons signalé auparavant.

Le mémoire est organisé en quatre chapitres comme suite:

Dans le premier chapitre, nous expliquons la notion de jumeau numérique en détail et le système de distribution automatique de produits avec tous ses composent internes et

externes. Quant au deuxième chapitre, il s'agira de présenter le langage de modélisation SysML et l'ingénierie dirigée par les modèles quand a utilisé dans ce projet. Le troisième chapitre sera la présentation de la méthodologie proposé avec tous les étapes de développement de jumeau numérique des distributeurs automatiques de produits. Le quatrième chapitre sera consacré à la réalisation et test de l'application. Le champ de cette application est la « Gestion des Distributeurs Automatique de Produits ».

CHAPITRE I

SYSML et DIGITAL TWIN pour l'ingénierie des systèmes

I.1. INTRODUCTION

Le numérique envahit notre quotidien. Le numérique se retrouve partout: dans l'agriculture, l'industrie, l'enseignement, l'administration, la santé, la défense, les transports, l'écologie, l'habitat et bien d'autres domaines encore. Le jumeau numérique s'agit d'une réplique numérique exacte, d'un équipement, prenant en compte l'intégralité de ses composants et de leurs propriétés. Dans ce chapitre en va présenter la notion de digital twin et le langage de modélisation SysML, et l'ingénierie dirigée par les modèles.

.

I.2. JUMAUX NUMERIQUES

I.2.1. Définition

Le jumeau numérique est un clone virtuel d'un système physique ou d'un processus. Il implique systématiquement l'existence d'un couple « modèle numérique » avec l'objet qu'il copie. Les objets concernés peuvent être un produit, une machine, une ligne de production, un process, une supply chain. Suivant le système concerné et l'utilisation souhaitée il peut être un modèle géométrique, multiphysique, fonctionnel, comportemental. Il doit évoluer dans le temps comme son jumeau réel.

Il permet d'améliorer le pilotage, la sécurisation et l'optimisation de lignes de production et des usines, la continuité numérique au niveau du produit, de sa conception à sa fin de vie, la surveillance et la maintenance prédictive. Il permet de mettre en place de nouveaux modèles économiques dans la supply chain (modèle as a service). Il permet d'obtenir un gain en qualité des produits en améliorant la correction des process. Il permet une traçabilité accrue des objets/process intégrant d'avantage d'informations sur les composants, les fournisseurs et la production. Il est un outil de rupture pour les besoins de formation et de démonstration sur des systèmes complexes et difficilement duplicables ou transportables. [2]

Le jumeau numérique d'un équipement/système/installation industriel est une image numérique de cet équipement/système/installation qui s'appuie :

- Sur une représentation virtuelle qui peut être à base physique, issu d'un traitement de données ou d'un mixte des deux
- Sur une connexion avec l'équipement, le système ou l'installation qu'il représente permettant au modèle de s'actualiser en fonction des données qu'il reçoit. [1]

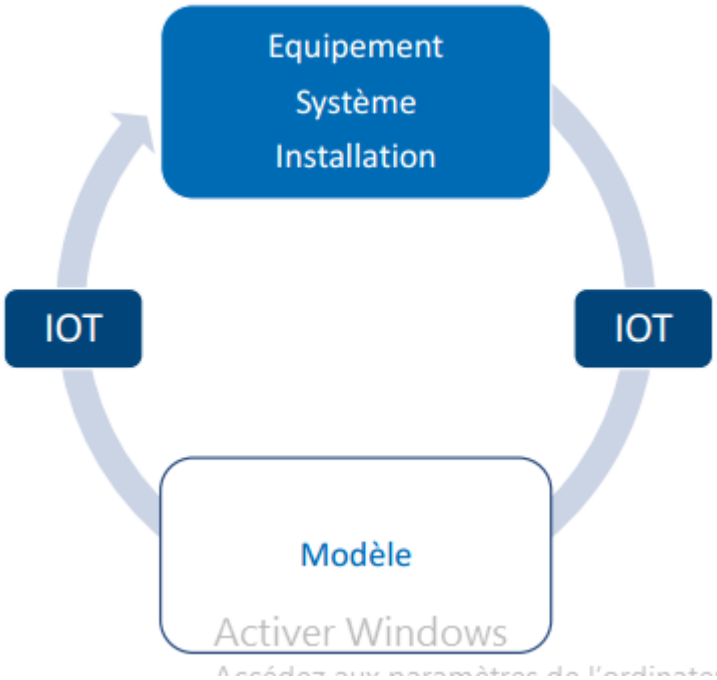


Figure I.1 : Jumeau numérique d'un équipement/système/installation

I.2.2. Jumeaux numériques trouvent leurs valeurs dans des fonctions qui leur sont propre

• Individualiser les décisions à la situation de chaque composant

• Etendre du domaine d'observation (limité dans le monde réel aux points de mesure) à l'ensemble du modèle

• Prédire le comportement futur de l'équipement. [1]

I.2.3. Eléments technologiques clefs

Il doit être le plus fidèle possible à son jumeau réel et être capable d'évoluer en temps réel ou en accéléré. Cela impose :

1. Un échange de données entre les deux jumeaux : une capacité à générer de la donnée pertinente, fiable et au bon niveau de précision via la mise en place de capteurs distribués, une capacité de collecte, d'acheminement et de stockage sécurisée des données
2. Une stratégie de construction du jumeau numérique permettant d'incrémenter et d'enrichir les modèles physiques par les données
3. Une capacité du JN à intervenir sur le jumeau physique pour du (de la) contrôle/correction automatique

4. Un recalage du jumeau numérique à partir des données transmises par le jumeau.

5. Une capacité à calculer un état prédictif dans des temps courts compatibles avec la prise de décision. [2]

I.2.4. Exemples d'usage

- Maintenance prédictive et dimensionnement des postes de travail.
- Amélioration de la qualité : détection de déviations, réduction de la variabilité, analyse approfondie des données historiques après un évènement
- Amélioration du rendement de l'objet ou de l'usine
- Diminution du temps de développement de nouveaux produits/process
- Traçabilité accrue des objets/process (informations sur les composants, les fournisseurs et la production) [2].

I.3. INTERNET DES OBJETS IOT

L'Internet des objets fait référence à un type de réseau pour connecter quoi que ce soit à Internet sur la base des protocoles stipulés grâce à des équipements de détection de l'information pour effectuer des échanges d'informations et des communications afin reconnaissances, intelligentes, positionnement, traçage, surveillance et administration.

Le paradigme de l'Internet des objets (IoT) est basé sur l'intelligence et l'auto-configuration nœuds (objets) interconnectés dans une infrastructure de réseau dynamique et globale. Celles-ci les choses peuvent être des appareils informatiques, des machines mécaniques et numériques, des objets, des animaux ou les personnes qui disposent d'identifiants uniques et la possibilité de transférer des données sur un réseau sans nécessiter d'interaction homme à homme ou homme à ordinateur [4].



Figure I.2 : Internet des objets

I.3.1. IOT : Une présentation

Internet des objets = des objets + de "l'intelligence". Un organe de calcul = un "minuscule" ordinateur pouvant exécuter un "logiciel"= un microcontrôleur+ une connexion réseau vers l'internet [5].

I.3.2. Caractéristiques

Les caractéristiques fondamentales de l'IOT sont les suivantes:

- **Inter connectivité:** en ce qui concerne l'IOT, tout peut être interconnecté avec l'information et la communication mondiales Infrastructure.
- **Services liés aux objets:** l'IoT est capable de fournir services liés aux choses dans les contraintes des choses, telles que protection de la vie privée et cohérence sémantique entre physique les choses et leurs objets virtuels associés. Afin de fournir services liés aux choses dans les contraintes des choses, technologies dans le monde physique et le monde de l'information changement.
- **Hétérogénéité:** les appareils de l'IOT sont hétérogènes comme basé sur différentes plates-formes matérielles et réseaux. Ils peuvent interagir avec d'autres appareils ou plates-formes de services via différents réseaux.

- **Changements dynamiques:** l'état des appareils change dynamiquement, par exemple, dormir et se réveiller, connecté et / ou déconnecté en tant que ainsi que le contexte des appareils, y compris l'emplacement et la vitesse. De plus, le nombre d'appareils peut changer dynamiquement.
- **Échelle énorme:** le nombre d'appareils qui doivent être gérés et qui communiquent entre eux seront au moins un ordre de grandeur plus grand que les appareils connectés au Internet actuel. La gestion des données sera encore plus critique générés et leur interprétation à des fins d'application. Cela concerne la sémantique des données, ainsi que les données efficaces manipulation.
- **Sécurité:** à mesure que nous tirons profit de l'IoT, nous ne devons pas oublier sur la sécurité. En tant que créateurs et destinataires de l'IoT, nous doit concevoir pour la sécurité. Cela comprend la sécurité de notre personnel donné et la sécurité de notre bien-être physique. Sécuriser les points de terminaison, les réseaux et les données se déplaçant à travers tout cela signifie créer un paradigme de sécurité qui évoluera.
- **Connectivité:** la connectivité permet l'accessibilité au réseau et compatibilité. L'accessibilité se connecte à un réseau pendant la compatibilité offre la capacité commune de consommer et produire des données.

I.3.3. IOT Architecture

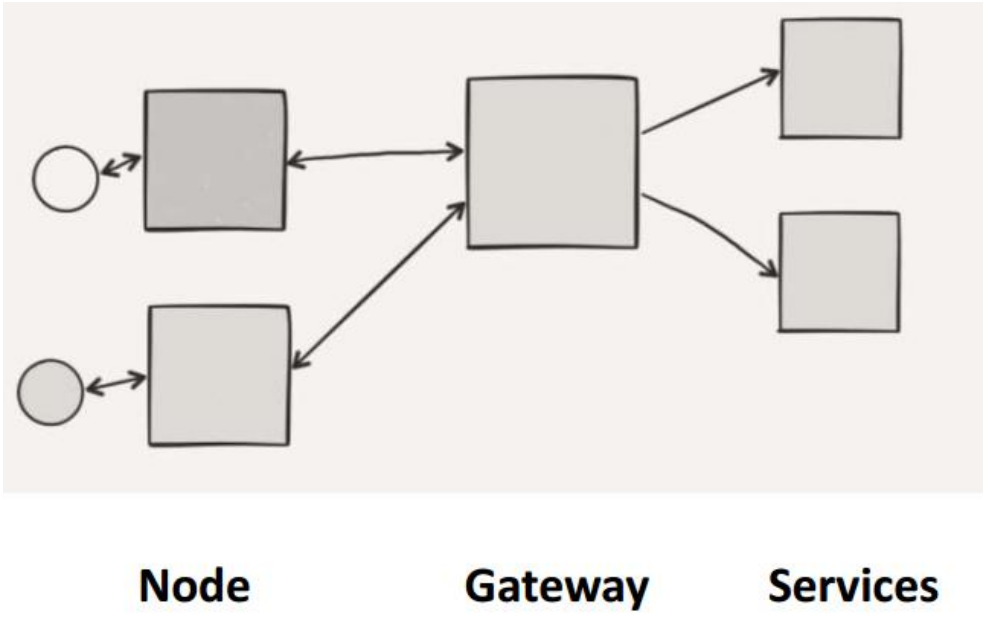


Figure I.3 : Architecture de l'IOT

I.4. NODE-RED

L'outil de simulation Node-RED est construit sur Node.js, tirant pleinement parti de son modèle non bloquant piloté par les événements. Cela le rend idéal pour fonctionner en périphérie du réseau sur du matériel à faible coût tel que le Raspberry Pi ainsi que dans le cloud. Ce flux agit comme un simulateur de périphérique virtuel qui envoie des données Json aux courtiers MQTT (serveurs) sur Internet. Dans mon exemple, je l'envoie aux appareils test.mosquitto.org et AWS IOT. Flux assez simple. Vous pouvez ajouter des appareils supplémentaires si vous le souhaitez. [8]

Pour envoyer le SMS, nous allons utiliser la plateforme Node-Red. C'est une plateforme de programmation visuelle dédiée à la création d'applications de type Internet of Things. Elle a été développée à l'origine par IB et elle est basé sur la notion de programmation en flux [6].

Node-RED fournit un éditeur de flux sous forme d'une application web reposant sur un serveur Node.js. Les flux créés dans Node-RED sont tous stockés sous format JSON [6].

Après l'installation de Node-RED, vous devez le lancer en exécutant la commande node-red à travers l'invite de commandes et en ensuite nous pouvons accéder à la page web d'accueil de Node-RED en utilisant un navigateur web et l'url suivante : http://localhost:1880.

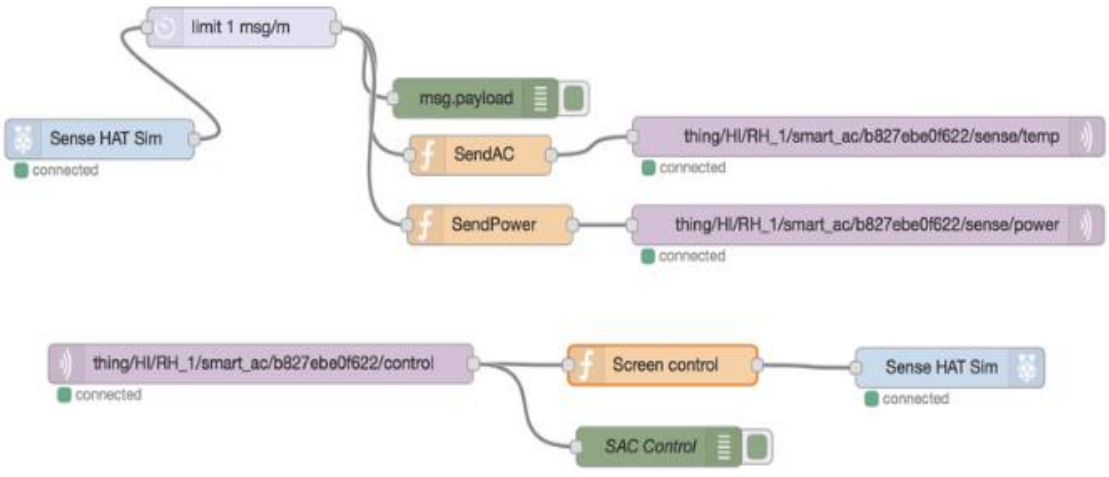


Figure I.4 : Flux rouge nœud simulant des dispositifs IOT et testant la communication flux dans l'infrastructure du système.

I.4.1. POTOCOLE MQTT

MQTT est un protocole Client/Serveur. Les clients sont des dispositifs matériels qui communiquent avec un serveur pour diffuser et/ou recevoir des messages [7].

I.4.2. Flux MQTT très simple

Faire glisser un nœud inject (dans input) et un nœud MQTT (dans output), relier les deux nœuds. [9]

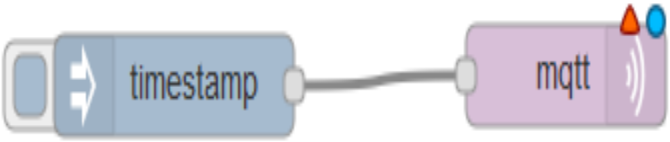


Figure I.5 : Flux de protocole MQTT

I.5. PLATE FORME ECLIPSE DITTO

Eclipse DITTO est une technologie de l'IOT implémentant un modèle logiciel appelé «jumeaux numériques» Un jumeau numérique est une représentation virtuelle, basée sur le cloud, de son homologue du monde réel («things» du monde réel, par exemple des appareils comme les capteurs, le chauffage intelligent, les voitures connectées, les réseaux intelligents, les bornes de recharge pour véhicules électriques,…).

La technologie reflète potentiellement des millions et des milliards de jumeaux numériques résidant dans le monde numérique avec des «choses» physiques. Cela simplifie le développement de solutions IOT pour les développeurs de logiciels car ils n'ont pas besoin de savoir comment ni où exactement les «choses» physiques sont connectées.

Avec DITTO, un « thing » peut simplement être utilisée comme n'importe quel autre service Web via son jumeau numérique. [11]

I.5.1. Quand utiliser DITTO

Utilisez-le afin d'obtenir une API à part entière et sensible aux autorisations (HTTP, Web Socket et autres protocoles de messagerie) pour interagir avec vos jumeaux numériques et tous les aspects qui les entourent. [10,11]

I.6. OBJECTIFS DE DEVELOPPEMENT DE DIGITAL TWIN

Un jumeau numérique, c'est d'abord des capteurs sur le monde réel, des algorithmes qui traitent les données captées pour simuler comment pourrait évoluer une situation, puis une interprétation des résultats pour informer des utilisateurs.

- **La fiabilité :**

La fiabilité d'une prévision numérique tient d'abord à la qualité des données. Dans beaucoup de domaines, les modèles scientifiques qui sous-tendent les algorithmes sont disponibles et offrent des niveaux de précision pour "copier" le comportement réel amplement suffisants au regard du besoin de prédiction.

Dans ces cas, c'est donc exclusivement la qualité des données en entrée qui déterminera la qualité de la prédiction.

L'évolution exponentielle des capacités de stockage de données et des capacités de calcul est donc un inducteur majeur du développement des jumeaux numériques. [12]

- **Prédire le futur :**

La caractéristique la plus intéressante des jumeaux numériques, c'est qu'ils permettent de prédire le futur, donc de prendre les décisions qui changeront ce futur.

Prédire le futur, être capable de changer le cours des choses pour le meilleur. [12]

- **Améliorer la qualité :** Le but des images numérique est d'amélioré la qualité des informations ou d'en extraire de l'information.
- **Capter :** A l'aide des sonsors, l'image numérique peut capter tous changement sur le système réel.
- **Analyse :** Analyses d'hypothèses et simulations opérationnelles pour assurer la disponibilité et minimiser les coûts de maintenance.
- **Contrôle de la durée de vie et de la maintenance des produits et systèmes :** Améliorer la connaissance de l'état réel de produit grâce à la surveillance en temps réel [13].

I.7. SYSML

SYSML est un langage de modélisation de systèmes à usage général qui peut être utilisé pour créer et gérer modèles de systèmes utilisant des constructions graphiques bien définies avec une sémantique sous-jacente. SysML réutilise un sous-ensemble d'UML2 les construit et les étend en ajoutant de nouveaux éléments de modélisation et deux nouveaux types de diagrammes.

L'ensemble des diagrammes comportementaux et structurels combinés avec le diagramme des exigences et le diagramme paramétrique offrent une vue intégrée d'un système. Mais SysML représente bien plus qu'un simple ensemble de diagrammes.

Derrière les diagrammes, il y a un référentiel de modèles de syntaxe abstraite qui représente formellement toutes les constructions de modélisation.[20] Le comportement du système dans SysML est capturé par une combinaison d'activités des diagrammes, des diagrammes de machine d'état et / ou des diagrammes de séquence et la sémantique associée.

Le sous-ensemble fondamental de la spécification UML fournit la sémantique supplémentaire pour permettre aux diagrammes d'activité SysML d'être exécutés de manière standard.

De plus, SysML comprend des diagrammes paramétriques pour capturer des modèles de comportement basé sur des contraintes, comme la dynamique en temps continu en termes de flux d'énergie. Cependant, la syntaxe et la sémantique de ces descriptions de comportement en paramétrique ont été laissées ouvertes pour s'intégrer à

d'autres simulations et des capacités de modélisation d'analyse pour soutenir l'exécution de ces modèles. [20]

I.7.1. Relation entre UML et SYSML

La demande de propositions UML pour l'ingénierie des systèmes a été élaborée conjointement par l'OMG et le Conseil international d'ingénierie des systèmes (INCOSE) et publiée par l'OMG en mars 2003. La demande de propositions précisait les exigences pour étendre UML afin de répondre aux besoins de la communauté de l'ingénierie des systèmes. La spécification SysML a été développée en réponse à ces exigences par le groupe diversifié de fournisseurs d'outils, d'utilisateurs finaux, d'universités et de représentants du gouvernement. [22].

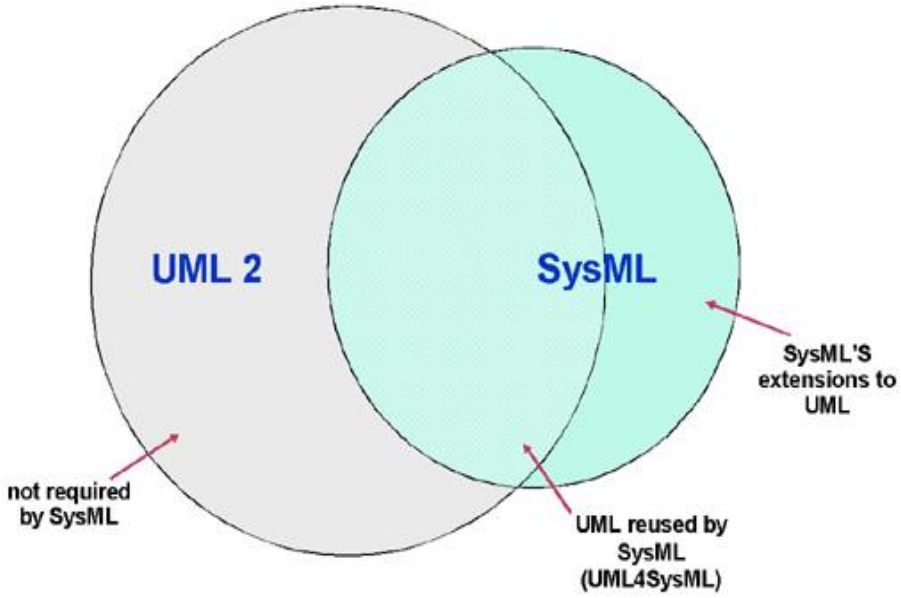


Figure I.6: Relation entre UML et SYSML

I.7.2. Résumé du diagramme SYSML

Les types de diagrammes SysML sont identifiés dans la figure I.7 et résumés ci-dessous [22].

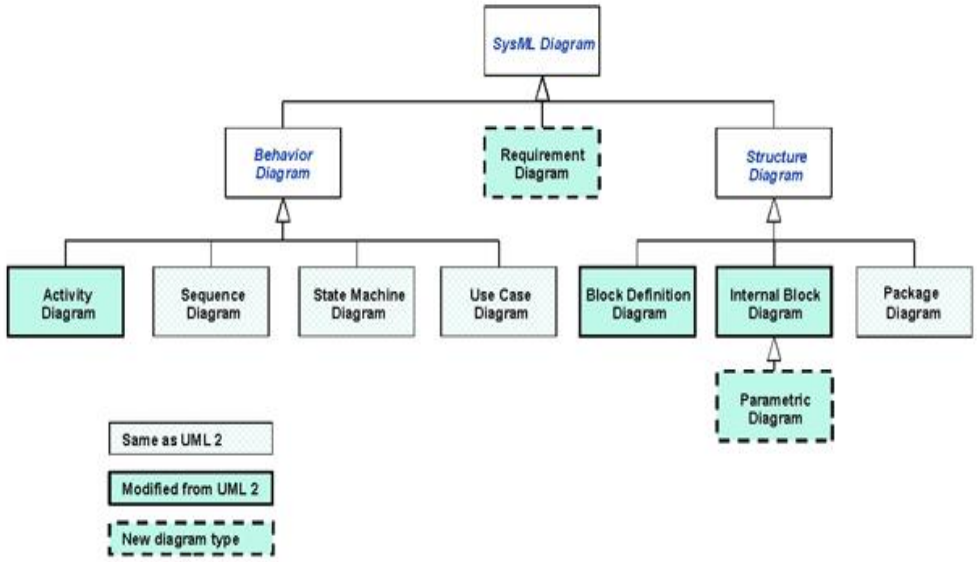


Figure I.7: Types de diagramme SYSML

Le bloc est l'unité de base de la structure dans SYSML et peut être utilisé pour représenter le matériel, les logiciels, les installations, le personnel ou tout autre élément du système. La structure du système est représentée par des diagrammes de définition de blocs et des diagrammes de blocs internes. Un diagramme de définition de bloc décrit la hiérarchie du système et les classifications du système / composant. Le schéma de principe interne décrit la structure interne d'un système en termes de pièces, de ports et de connecteurs.

Le diagramme de package est utilisé pour organiser le modèle. Les diagrammes de comportement incluent le diagramme de cas d'utilisation, le diagramme d'activité, le diagramme de séquence et le diagramme de machine d'état. Un diagramme de cas d'utilisation fournit une description de haut niveau des fonctionnalités obtenues grâce à l'interaction entre des systèmes ou des parties de système. Le diagramme d'activité représente le flux de données et le contrôle entre les activités.

Un diagramme de séquence représente l'interaction entre les parties collaboratrices d'un système. Le diagramme de la machine d'état décrit les transitions d'état et les actions qu'un système ou ses composants effectuent en réponse à des événements.

SYSML comprend une construction graphique pour représenter les exigences basées sur du texte et les relier à d'autres éléments du modèle. Le diagramme des exigences capture les hiérarchies des exigences et la dérivation des exigences, et les relations satisfaire et vérifier permettent au modélisateur de relier une exigence à un élément de modèle qui satisfait ou vérifie les exigences. Le diagramme des exigences

fournit un pont entre les outils de gestion des exigences typiques et les modèles de système.

Le diagramme paramétrique représente les contraintes sur les valeurs des propriétés du système telles que les performances, la fiabilité et les propriétés de masse, et sert de moyen d'intégrer les modèles de spécification et de conception aux modèles d'analyse technique.

SYSML comprend également une relation d'allocation pour représenter divers types d'allocation, y compris l'allocation de fonctions aux composants, logiques aux composants physiques et les logiciels au matériel.

Un exemple simple de certains des types de diagrammes clés est mis en évidence dans la figure I.8. [22]

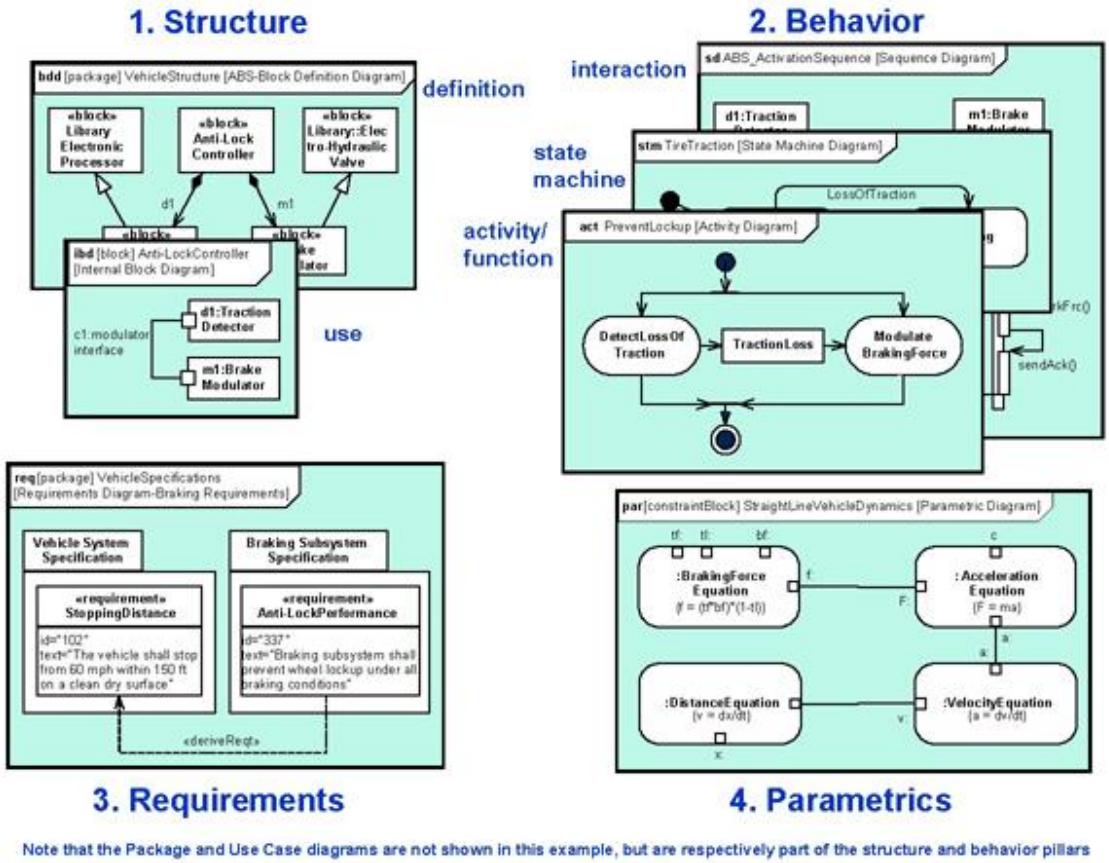


Figure I.8: Les quatre piliers du SYSML

I.7.3. Implémentation et exécution

SYSML est destiné à capturer des modèles paramétriques réels, pas seulement documentation qui peut être résolue par des outils externes. Nous décrivons ici une façon

d'atteindre cette intention exécutable. Comme le montre la figure I.8, les fournisseurs commencent à prendre en charge SYSML dans leurs outils de modélisation, souvent en exploitant leurs capacités UML2 existantes.

Sous les diagrammes graphiques et les navigateurs d'arborescence de modèles, ces outils capturer la structure du modèle d'une manière sémantiquement riche. Cela permet une exécution rapide des constructions SYSML basées sur UML2 tels que les diagrammes de séquence. Cependant, comme les constructions paramétriques sont nouvelles, les outils actuels ne soutenir l'exécution des paramètres [21].

I.8. DIGRAMME DE BLOC

Comme on a voir que le langage SysML est un langage graphique qui comporte 9 diagrammes chacun décrivant le système selon un point de vue particulier. Parmi les on a le diagramme de bloc diagramme de définition de bloc ».

- Modéliser l'architecture du système,
- décrire le système et ses composants via des blocs : composant matériels, logiciels,
- représenter la hiérarchie entre les composants des systèmes,
- représenter les relations entre les composants.

I.8.1. Un Bloc

Le bloc représente le système complet, ou sous système ou un élément du système. Cela peut même être un composant. Un bloc est représenté graphiquement par un rectangle comportant plusieurs compartiments. [23]

- Le compartiment supérieur contient le nom du bloc
- Le compartiment propriété permet de définir les caractéristiques de l'élément.
- Le compartiment opération détaille les actions réalisées par le bloc.
- le compartiment contraint permet de définir les contraintes de la fonction.

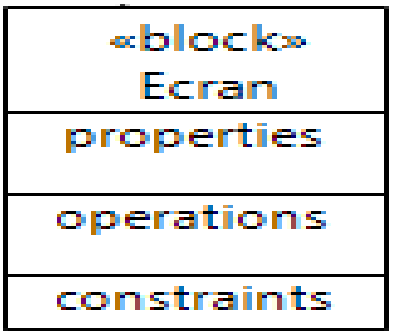


Figure I.9: Exemple d'un bloc

I.8.2. Relation

Dans un diagramme de définition bloc, il y a 3 relations possibles :

- La composition, l'un des blocs est le contenant et l'autre est le contenu, mais le contenant n'a pas de raison d'être sans le contenu. Exemple la guitare et les cordes de guitare : Sans cordes, la guitare n'a aucun intérêt mais les cordes sont contenues dans la guitare. Une relation de composition est représentée par un losange plein [23].



Figure I.10: Relation de composition

- L'association représentée par un trait simple. Elle lie deux blocs égaux. Si on prend comme exemple, l'ensemble guitare + amplificateur, chaque sous-système est indépendant, mais ils sont liés [23].



Figure I.11: Relation de l'association

- L'agrégation est identique à la composition, à la différence prête que le contenu a une existence sans le contenant. Par exemple une salle de classe est composée de tables et de chaises mais sans tables ni chaise la salle peut encore être utilisée. Une relation d'agrégation est représentée par un losange vide [23].

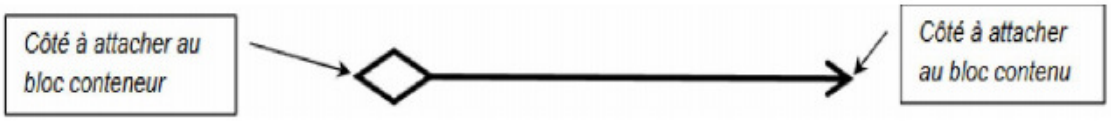


Figure I.12: Relation de l'agrégation

I.9. META MODELESATON

Un méta modèle est une abstraction permettant de décrire des modèles. Généralement, la pratique de la méta-modélisation consiste à spécifier des méta-modèles contemplatifs qui reflètent l'aspect statique des modèles. L'aspect statique d'un modèle représente l'ensemble des concepts qui ont servi à décrire ce modèle ainsi que les liens entre ces concepts.

On utilise aussi le terme aspect structurel. L'aspect dynamique complète l'aspect statique et renseigne sur la manière avec laquelle les instances des concepts vont interagir lors de l'exécution du méta-modèle. Cela revient à décrire le comportement dynamique des instances des concepts [24].

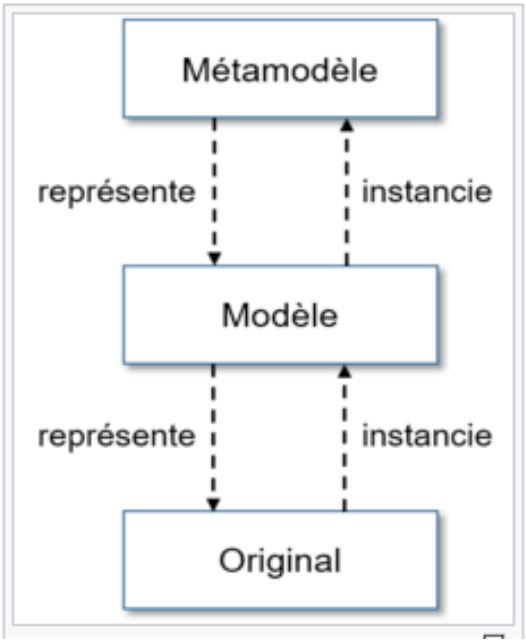


Figure I.13 : Liens d'instanciation et de représentation entre un méta modèle, un modèle et un original

I.9. PRINCIPE DU MODEL DRIVEN ARCHITECTURE (MDA)

L'architecture dirigée par les modèles ou MDA est une démarche de réalisation de logiciel, proposé et soutenue par l'OMG. C'est une variante particulière de l'ingénierie dirigée par les modèles IDM, ou MDE.

Le principe de base du MDA est l'élaboration de modèles indépendants de plate-forme (PLM) et la transformation de ceux-ci en modèles dépendants de plates-formes (PSM) pour l'implémentation concrète du système. Les techniques employées sont donc principalement des techniques de modélisation et des techniques de transformation de modèles. [26]

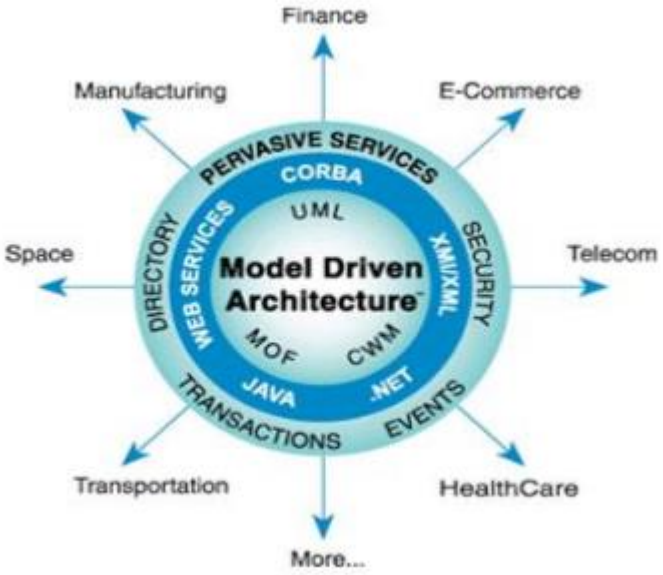


Figure I.14: Architecture MDA

I.10. TRANSFORMATION DES MODELES

Une transformation est une opération qui :

- Prend un modèle source en entrée
- Fournit un modèle cible en sortie

❖ Transformation endogène

❖ Dans le même espace technologique

❖ Les modèles source et cible sont conformes au même méta modèle

o Transformation d'un modèle UML en un autre modèle UML

❖ Transformation exogène

- Entre 2 espaces technologiques différents
- Les modèles source et cible sont conformes à des méta-modèles différent

- Transformation d'un modèle UML en programme java
- Transformation d'un fichier XML en schéma de BDD [25]

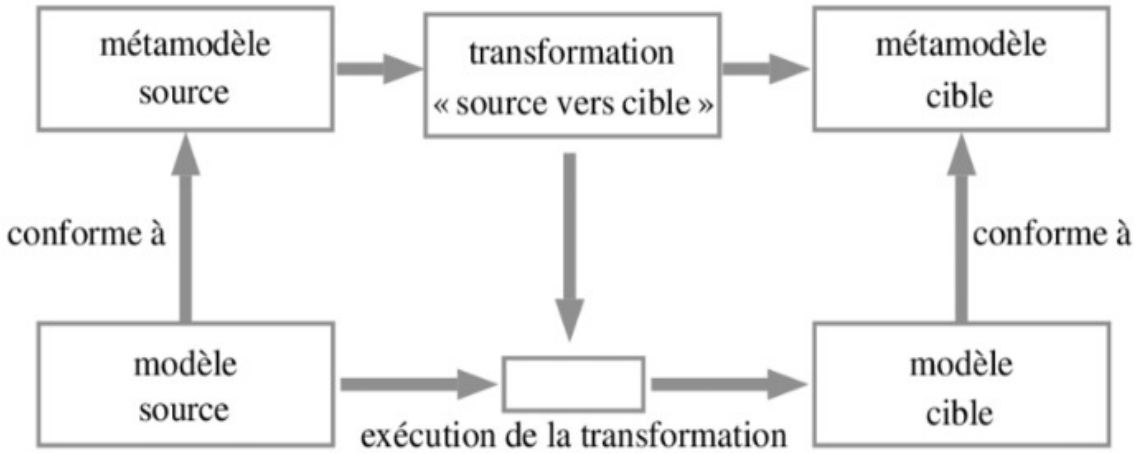


Figure I.15: Principe de transformation des modèles

I.11. MODEL DRIVEN ENGINEERING (MDE)

L'ingénierie dirigée par les modèles (MDE) est une méthodologie de développement de logiciels qui se concentre sur la création et l'exploitation de modèles de domaine, qui sont des modèles conceptuels de tous les sujets liés à un problème spécifique. Par conséquent, il met en évidence et vise des représentations abstraites des connaissances et des activités qui régissent un domaine d'application particulier, plutôt que les concepts informatiques.

L'approche MDE vise à augmenter la productivité en maximisant la compatibilité entre les systèmes (via la réutilisation de modèles standardisés), en simplifiant le processus de conception (via des modèles de modèles de conception récurrents dans le domaine d'application) et en favorisant la communication entre les individus et les équipes travaillant sur le système (via une standardisation de la terminologie et des bonnes pratiques utilisées dans le domaine applicatif).

Un paradigme de modélisation pour MDE est considéré comme efficace si ses modèles ont un sens du point de vue d'un utilisateur familier avec le domaine et s'ils peuvent servir de base pour la mise en œuvre de systèmes. Les modèles sont développés grâce à une communication étendue entre les chefs de produit, les concepteurs, les développeurs et les utilisateurs du domaine d'application. À mesure que les modèles

approchent de leur achèvement, ils permettent le développement de logiciels et de systèmes [27].

I.12. CONCLUSION

Les jumeaux numériques sont industrielles faisant parties d'importants enjeux d'innovation actuelle il apportera à la société les solutions pour une meilleure durabilité de la planète, des personnes et de l'entreprise. Dans ce chapitre, on a essayé de présenter la notion de digital twin et les concepts de SYSML, la transformation de modèles et la notion de MDE.

Le but d'introduire ces concepts est de permettre la compréhension de notre travail qui consiste à la proposition d'une transformation des modèles UML vers un modèle conforme au méta-modèle de ditto et une transformation vers un fichier texte format JSON. Le chapitre suivant porte plus de détail dans ce cadre.

CHAPITRE II

Distributeurs automatiques des produits

II.1. INTRODUCTION

La distribution automatique regroupe l'ensemble des biens distribués par le biais des distributeurs automatiques placés au sein d'établissement privés ou publiques. Un distributeur automatique est une machine qui permet d'obtenir des services, sans intervention humaine (ce que l'on appel libre service), tout cela est grâce aux techniques d'automatisme. Depuis la naissance des distributeurs automatique, ce système de distributeur automatique de produits ne cesse d'évoluer grâce à l'innovation et à l'évolution des modes de vie.

II.2. HISTOIRE DE DISTRIBUTEUR AUTOMATIQUE

II.2.1. Naissance de distributeur automatique

L'histoire des distributeurs automatiques est apparue vers 215 avant J-C, ce distributeur aurait été inventé par Héron qui était un mathématicien grec, peut de temps après la création de la monnaie et elle fonctionnait avec une pièce de 4 drachmes. Ce premier distributeur était conçu pour vendre de l'eau dans les temples. Puis ensuite au tour des égyptien d'inventer un distributeur automatique qui permet de vendre des coton imbibée de vin. Mais réellement le premier distributeur automatique avec une fonction commerciale apparu en 1880 à Londres, qui celui-ci commercialise des cartes postales.

La première machine qui va distribuée des sodas et des sucreries va naître en 1920 aux Etats-Unis. Le premier distributeur de paquet de cigarette va apparaître vers les années 1930. C'est à cette époque là que les entreprises vont penser à diversifié les produits pour les consommateurs. [14]

II.2.2. Définition de distributeur automatique

Un distributeur automatique est un automate permettant la vente d'un produit sans intervention d'un vendeur. L'utilisation d'un distributeur automatique permet de rentabiliser de faibles volumes d'achats là où un point de vente physique ne serait pas rentables et d'assurer une vente 24H/24H lorsque le distributeur est approvisionné. La distribution automatique permet également la vente dans des espaces privés tels que les entreprises ou les collectivités. [16]

II .3. Avantage et inconvénient des distributeurs automatiques

Les distributeurs automatiques plaisent aux consommateurs car ceux-ci peuvent se satisfaire 365 jours par année et 24 heures sur 24 avec des produits frais. En entreprise l'avantage des distributeurs, c'est le symbole de la convivialité entre collègues. Les distributeurs automatiques offrent des choix de produits très innovant et très intéressant. Ces machines sont

très utiles et pratiques quand le temps manque pour une petite faim. Il suffit pour cela de mettre des pièces dans la machine.

Malgré ces avantages, il existe quelques inconvénients comme les coûts élevé des produits qui sont beaucoup plus chère que dans les commerces de détail. Il existe un autre inconvénient, les distributeurs automatiques qui sont implantés dans des lieux comme les écoles qui pousse fortement les jeunes à consommer des produits qui pousse à l'obésité. Pour les acheteurs de distributeurs automatiques représente aussi des avantages, les distributeurs permettent de réaliser des économies et mêmes des profits surtout dans les hôtels et les piscines. Ils sont maître de ce qu'elles proposent dans les distributeurs automatiques elle fixe librement ses prix. Mais il y a aussi des inconvénients à l'autogestion, gérer les distributeurs automatiques prennent du temps. L'approvisionnement peut se révéler très onéreux. [14]

Les avantages sont également nombreux pour l'entreprise qui décide d'installer des distributeurs automatiques de boissons ou de snacks :

- Indirectement, elle bénéficie de salariés plus productifs et de visiteurs mieux disposés,
- Directement, l'exploitation rapporte : entre le prix d'achat des produits distribués en gros et leur prix de vente aux consommateurs finaux, la marge bénéficiaire peut être importante, et ce même après intégration des coûts d'exploitation comme la maintenance et l'entretien de la machine,
- Ces bénéfices peuvent être utilement redistribués aux salariés, par exemple en finançant les activités du comité d'entreprise. Ce dernier les imputera à son budget œuvres culturelles et sociales. [18]

II.4. PRESENTATION DES PRODUITS SECS

Les produits sont à charger dans les colonnes. Les colonnes peuvent être simples ou doubles. Ainsi les produits les plus populaires seront chargés dans les colonnes doubles et les produits les moins populaires dans les colonnes simples [15]. (Schéma vue de face)

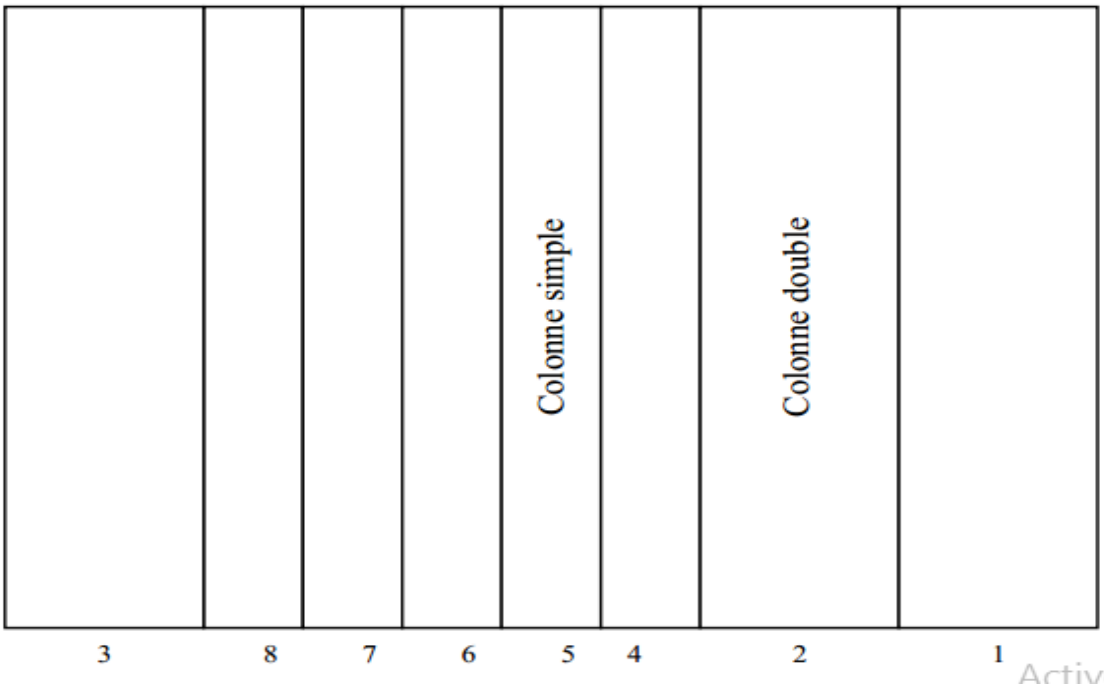


Figure II.1 : Présentation des colonnes simples/doubles des distributeurs automatiques des produits

II.5. ANATOMIE D'UN DISTRIBUTEUR AUTOMATIQUE DES PRODUITS

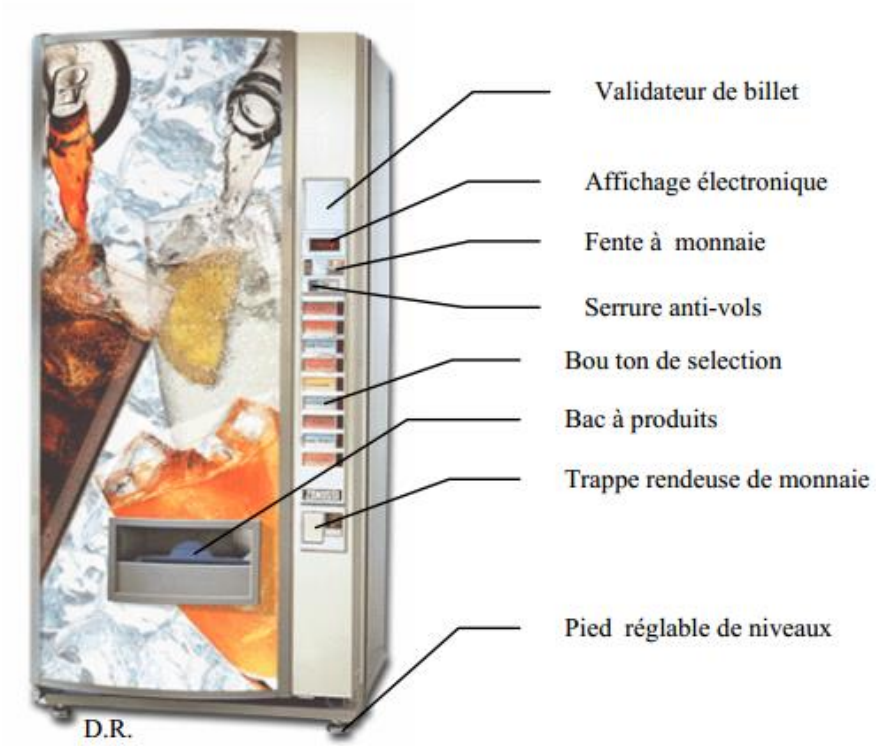


Figure II.2 : Anatomie d'un distributeur automatique des produits

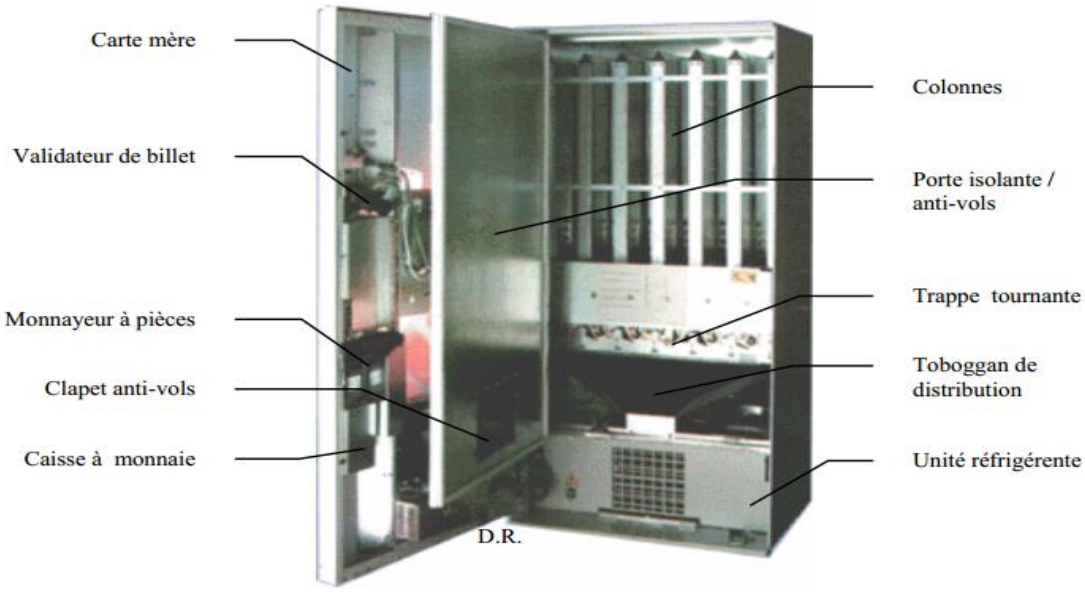


Figure II.3 : Anatomie internes d'un distributeur automatique des produits

II.6. COMPOSANTS INTERNES D'UN DISTRIBUTEUR AUTOMATIQUE DES PRODUITS

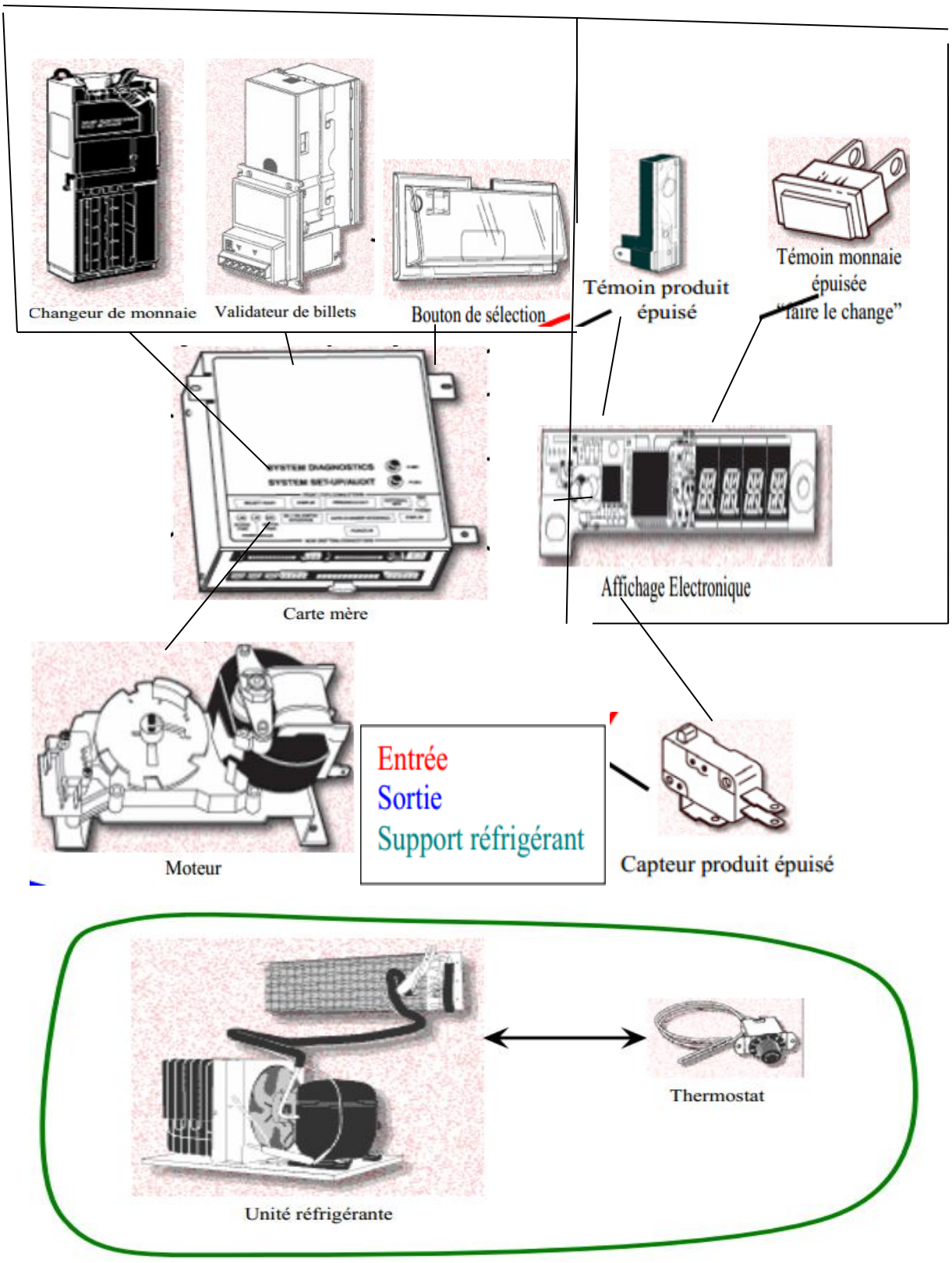


Figure II.4 : Les composants internes d'un distributeur automatique des produits

II.7 LES DIFFERENTS TYPES DE DISTRIBUTEURS AUTOMATIQUES DES PRODUITS

Les distributeurs automatiques des produits peuvent aujourd'hui distribuer absolument tous les produits imaginables. Et même ceux auxquels vous n'auriez jamais pensé.

A. Les distributeurs de boissons fraîches

Ce sont les distributeurs les plus connus, les plus traditionnels.

Les distributeurs de boissons fraîches proposent toutes sortes d'eaux minérales, de jus de fruits, de sodas, de boissons alcoolisées et non alcoolisées dans différents formats, en bouteilles de contenance variables ou en canettes.

B. Les distributeurs électriques de boissons chaudes

Les distributeurs de boissons chaudes sont les autres grands classiques des halls de gare, mais eux aussi évoluent.

C. Les distributeurs de snacks et friandises

Les distributeurs automatiques de confiseries et distributeurs de snacks sont bien connus pour leurs barres chocolatées ou céréalières, ou encore les biscuits ou chips.

D. Les distributeurs combinés

Des distributeurs combinés existent désormais, avec des compartiments et circuits séparés qui permettent aussi bien la conservation de produits froids, que celle de produits frais, et le réchauffage de snacks. [19]

II.8. PROCESSUS DE MISE EN MARCHE INITIALE

- Brancher le distributeur automatique sur le secteur
- Ouvrir le distributeur automatique
- Régler le thermostat
- Régler les cales de réglage en profondeur pour chaque colonne
- Régler les cames de réglage des moteurs pour chaque colonne
- Charger tous les produits dans chaque colonne
- Lancer le mode "setup"
- Régler individuellement les prix de vente de l'ensemble des produits
- Appuyer sur le bouton "reset" pour relancer le mode production

- Charger les pièces de change dans le monnayeur
- Etiqueter correctement les boutons de sélection
- Fermer la machine
- Effectuer des essais d'achat pour toutes les sélections
- Ouvrir le distributeur automatique
- Lancer le mode "setup"
- Réinitialiser tous les compteurs des colonnes
- Appuyer sur le bouton "reset" pour relancer le mode production
- Fermer la machine [15]

II.9.SECURITE

La plupart des distributeurs automatiques modernes ont été construits et testés pour éviter le vol. La majorité d'entre eux sont donc très massifs afin d'éviter qu'ils soient emportés. Ainsi, la masse d'un distributeur de boissons gazeuses rempli à pleine capacité peut excéder 400 kg. [17]

II.10.CONCLUSION

Alors, Les distributeurs automatiques de produits sont maintenant bien connus du grand public. Ils sont en effet présents dans nombreux lieux publics comme par exemple dans des grandes surfaces, aéroports, gares, parkings, écoles (malheureusement), entreprises etc. Ces machines délivrent automatiquement, après payement, plusieurs variétés de produits.

CHAPITRE III

Conception

III.1. INTRODUCTION

Les jumeaux numériques sont utilisés tout au long du cycle de vie des systèmes physiques pour simuler, prédire et optimiser le fonctionnement de ces derniers. Un jumeau numérique pour un système des distributeurs automatiques de produits est une représentation virtuelle qui permet de superviser, de comprendre et de prédire les caractéristiques de performances du distributeur.

Le développement de jumeaux numériques pour des distributeurs automatiques de produits permet une meilleure gestion de ces derniers que ce soit pour l'amélioration des performances des distributeurs automatiques ou pour la supervision de l'opération de commercialisation des produits à travers les distributeurs automatiques.

Dans ce chapitre, nous allons proposer une méthodologie permettant le développement et l'exploitation des jumeaux numériques pour la gestion et la supervision des distributeurs automatiques de produits.

III.2. IDEE GENERALE

Le développement d'un jumeau numérique consiste à générer une image numérique d'un système réel. Le système réel est alors connecté au jumeau numérique à travers les technologies des objets connectés (Internet of things ou IOT). Dans la méthodologie que nous proposons pour le développement de jumeaux numériques pour les distributeurs automatiques de produits, nous préconisons une démarche basée sur l'ingénierie dirigée par les modèles. L'objectif est d'avoir une démarche générique utilisable avec plusieurs types de distributeurs automatiques de produits.

La démarche consiste à transformer une modélisation du distributeur automatique en jumeau numérique représenté sous forme d'un ensemble d'objets connectés (Things). Le jumeau numérique permet de représenter toutes les caractéristiques et propriétés du distributeur, capter toutes les modifications sur le système réel à travers des capteurs et commander le système via des actionneurs.

La figure ci-dessus explique l'intégration avec le système des distributeurs automatiques.

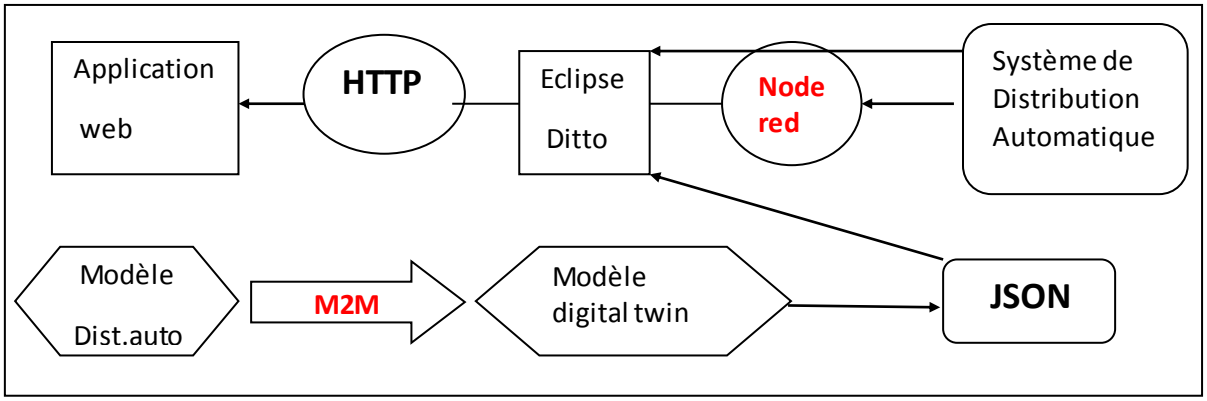


Figure III.1 : Intégration de l'application avec le système de distributeurs automatiques des produits

La figure suivante présente l'architecture de l'application.

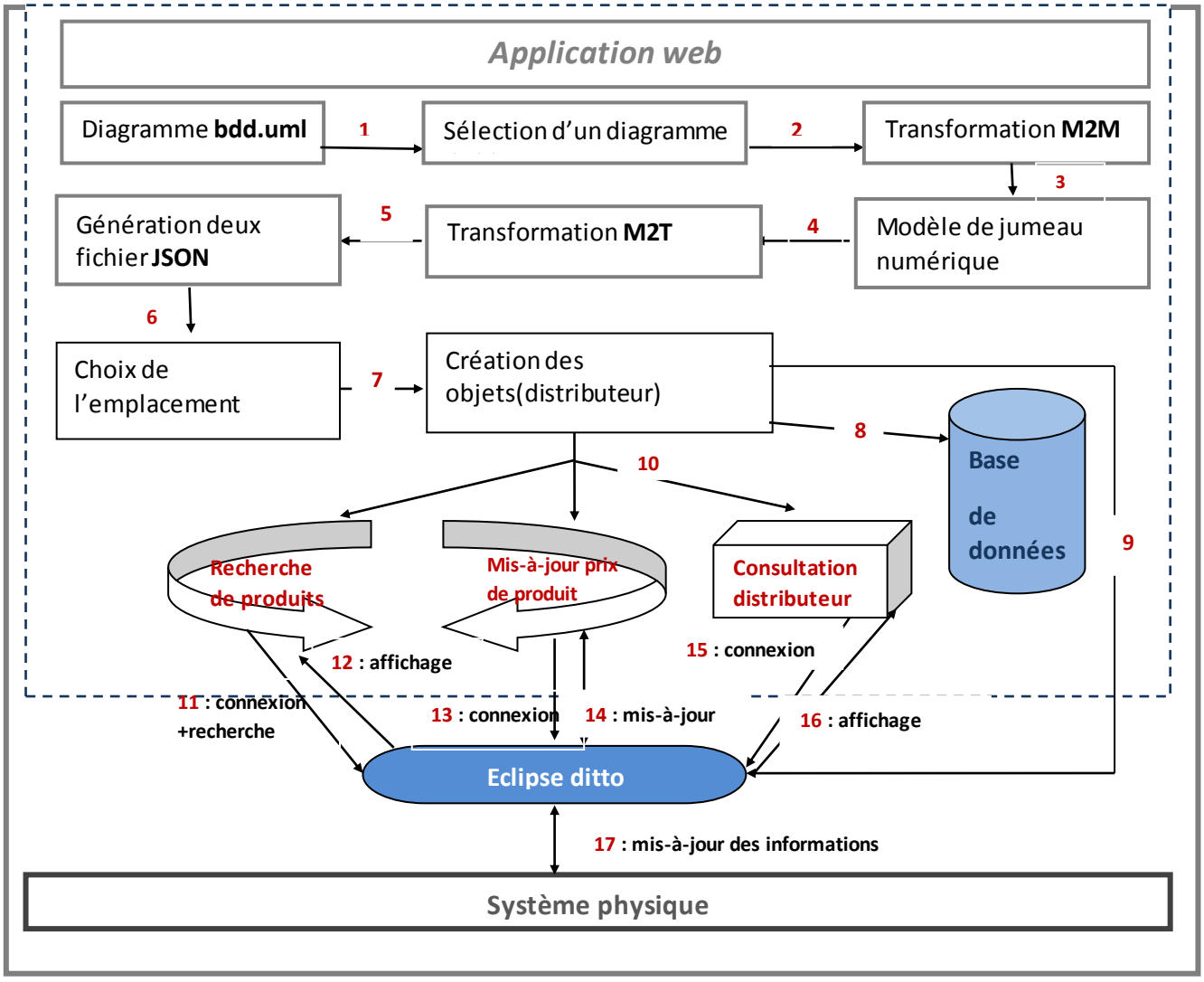


Figure III.2: Architecture de l'application web

Suivant la figure précédente, les étapes de fonctionnalité de l'application sont :

- ✓ L'application prend le diagramme de bloc de système comme un paramètre d'entrée en format UML et le transforme à un modèle de jumeau numérique. Puis elle le transforme une deuxième transformation qui donne deux fichiers JSON comme résultat « les étapes 1, 2, 3, 4et 5comme montre Figure III.2 **».**
- ✓ Après l'application donne le droit de choisir l'emplacement pour pouvoir créer le thing (distributeur) avec son policy qui sera ajouté dans la base de données et dans la plateforme eclipse ditto « comme les étapes 6 ,7 ,8 et 9 indique sur Figure III.2 ».
- ✓ Cela permet la consultation de ces objets (distributeurs automatiques), la recherche des produits et modification des prix des produits, alors afficher tous changement dans le système en temps réel «étape 17 sur la Figure III.2 ».

Le développement d'un jumeau numérique d'un système des distributeurs automatiques nécessite un environnement de développement spécial pour l'échange des informations entre l'image numérique et le système réel. La création des objets se fait au niveau de la plateforme eclipse DITTO. On va développer une application web qui permet de créer des Things et contrôler les distributeurs automatiques à travers cette application web.

III.3. MODELISATION DES SYSTEMES DES DESTRIBUTEURS AUTOMATIQUES DES PRODUITS

Nous avons choisi de modéliser le système de distributeurs automatiques de produits par le diagramme BDD (diagramme de définition de bloc) un des diagrammes de modélisation SysML. Le diagramme de définition de bloc permet d'exprimer la structure d'un système, d'un sous-système ou d'un composant. Les blocs peuvent représenter des entités physiques ou logiques. Ils sont décomposables, ils peuvent posséder des propriétés et un comportement.

Chaque bloc dans le diagramme représente un élément de système du distributeur automatique (écran, moteur, boite monnaie, etc.). Un bloc est représenté graphiquement par un rectangle comportant plusieurs compartiments qui contient les détailles sur le composent (élément) présenté par le bloc.

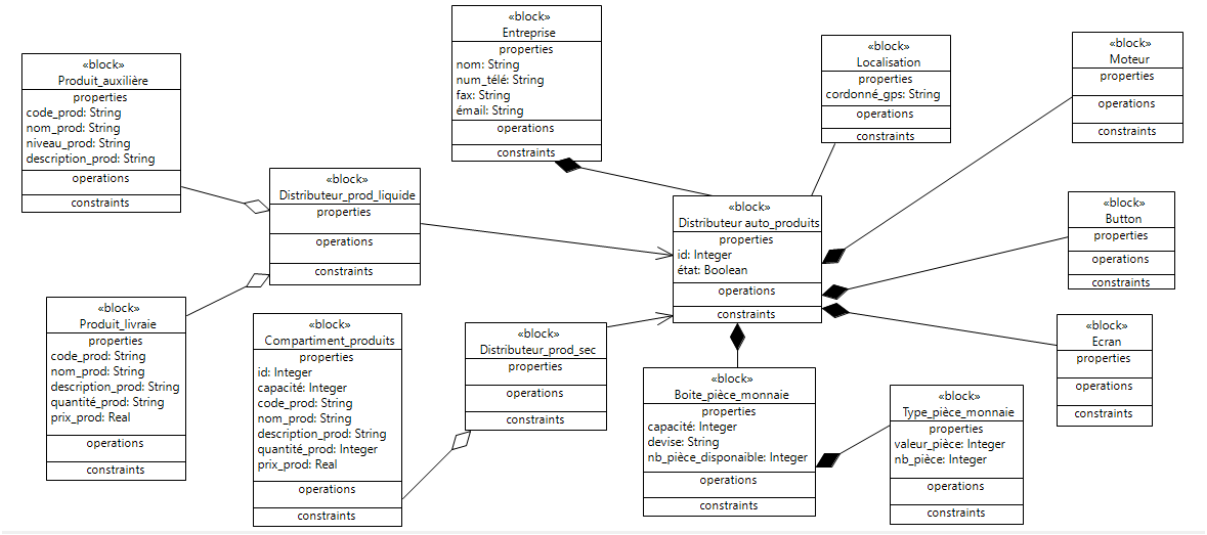


Figure III.3 : Modèle de blocs de distributeur automatique des produits

III.4. META-MODELE DES JUMEAUX NUMERIQUES

Pour la transformation des modèles aux jumeaux numériques, on doit utiliser le méta-modèle des jumeaux numériques, on a utilisé le méta-modèle présenté dans la figure 04 suivante :

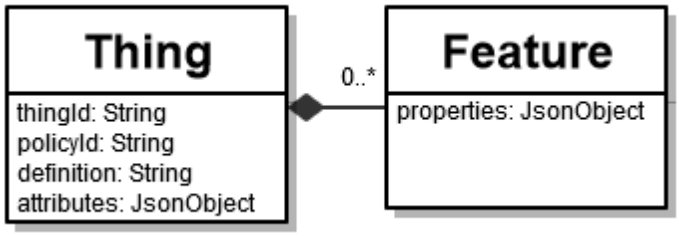


Figure III.4 : Méta-modèle des jumeaux numériques

Le méta-modèle décrit :

- Les objets (Things) : permettent de modéliser les composants physiques du système.
- Le Thing contient uniquement un policyId qui est lié à une stratégie contenant les informations d'autorisation. Ce méta modèle montre la structure dont Ditto a besoin pour l'API version 2.
- Les attributs et les caractéristiques (features) : permettent de décrire les propriétés des objets et les regrouper en caractéristiques.

III.4.1 THING

Les objets (things) sont des entités très génériques et sont principalement utilisées pour de multiples fonctionnalités appartenant à cet objet.

Exemple :

- Dispositif physique : une tondeuse à gazon, un capteur, un véhicule, une lampe.
- Périphérique virtuel : une pièce dans une maison, une centrale électrique virtuelle couvrant plusieurs centrales électriques, les informations météorologiques pour un emplacement spécifique collectées par divers capteurs.
- Entité transactionnelle : visite d'un véhicule (du début à l'arrêt), série de mesures d'une machine.
- Entité de données de base : un fournisseur d'appareils ou un fournisseur de services pour les appareils, une entité représentant une ville / région.
- Autre chose - si elle peut être modélisée et gérée de manière appropriée par les concepts / capacités pris en charge.

- **Thing ID** : Identifiant unique d'objet.
- **Policy ID :** Liens vers l'ID d'une policy existante qui contient les informations d'autorisation appliquées pour cet objet.
- **Définition :** Un objet peut contenir une définition. La définition peut également être utilisée pour rechercher des objets. La définition est utilisée pour lier un objet à un modèle correspondant définissant ses capacités / caractéristiques.
- **Attributes :** Les attributs décrivent l'objet plus en détail et peuvent être de tout type. Les attributs peuvent également être utilisés pour rechercher des objets. Ils sont généralement utilisés pour modéliser des propriétés plutôt statiques au niveau Thing. Statique signifie que les valeurs ne changent pas aussi fréquemment que les valeurs de propriété des fonctionnalités.

III.4.2. Features

Un objet peut contenir une quantité arbitraire de fonctionnalités.

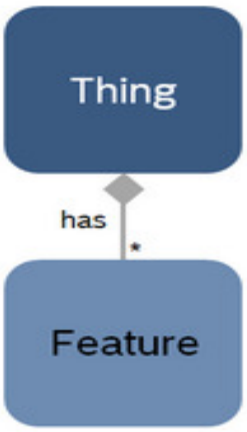


Figure III.5 : Relation entre Thing et Feature

- **Properties :** Les données relatives aux fonctionnalités sont gérées sous la forme d'une liste de propriétés. Ces propriétés peuvent être classées, par exemple pour gérer l'état, la configuration ou toute information de défaut. Les propriétés d'entité sont représentées comme un seul objet JSON.

III.4.3. Policy

Une stratégie permet aux développeurs de configurer facilement un contrôle d'accès à granularité fine pour Things et d'autres entités. Un objet (Thing) dans l'API version 2 contient un lien vers une policy sous la forme d'un policyId. Cette policy définit quels sujets authentifiés peuvent LIRE et ÉCRIRE l'objet ou même des parties de celle-ci (spécifiées hiérarchiquement).

- Une policy spécifique fournit à quelqu'un (appelé sujet) la permission de lire et / ou d'écrire une ressource donnée. La plupart des cas, il est logique d'accorder une autorisation de lecture en plus d'une autorisation d'écriture, car l'écriture n'implique pas la lecture.
- Une policy peut contenir des définitions de contrôle d'accès pour plusieurs ressources:
  - **Policy :** un personne qui a obtenu une autorisation d'écriture sur la ressource racine de la policy (c'est-à-dire la policy : /) est autorisée à gérer la policy elle-même.
  - **Thing :** La ressource « thing : / » est une ressource de niveau supérieur, « Thing : / features » est une sous-ressource. Au moment de l'exécution, les autorisations sont propagées à toutes les sous-entités de l'objet.

- Si vous accordez une autorisation de lecture au niveau supérieur et la révoquez dans une sous-entité, le sujet ne peut lire que la partie supérieure.
- Si vous omettez un sujet au niveau supérieur mais accordez une autorisation à une sous-entité, le sujet peut accéder uniquement à la partie inférieure (et à l'ID de Thing).

- La ressource Policy (adressable en tant que policy : /) définit le contrôle d'accès pour la policy elle-même.

III.4.4. Distinction entre les blocs

Nous devons mettre un mécanisme pour différencier entre les blocs, car certains blocs sont transformés à des objets (things) avec leurs attributs, d'autres à des fonctionnalités (features) avec leurs propriétés et les restes sont transformés à des propriétés complexes.

Pour cela on utilise des commentaires. A chaque bloc on rajoute un commentaire et on écrit soit « thing », « feature » ou « complex property ».

- Le bloc qui est associé par le commentaire « thing » est transformé à un objet.
- Le bloc qui est associé par le commentaire « feature » est transformé à une fonctionnalité.
- Le bloc qui est associé par le commentaire « complex property » est transformé à une propriété complexe.

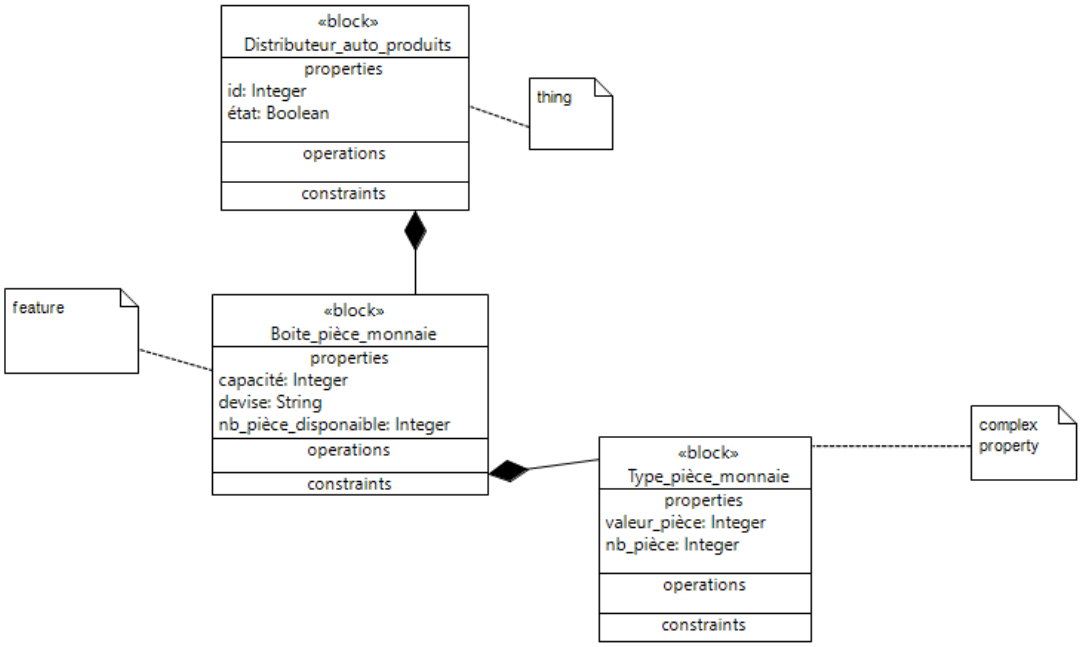


Figure III.6 : Exemple de distinction entre les blocs

Les commentaires peuvent être aussi utilisés pour décrire une définition sur l'objet et pour spécifier l'ID du policy.

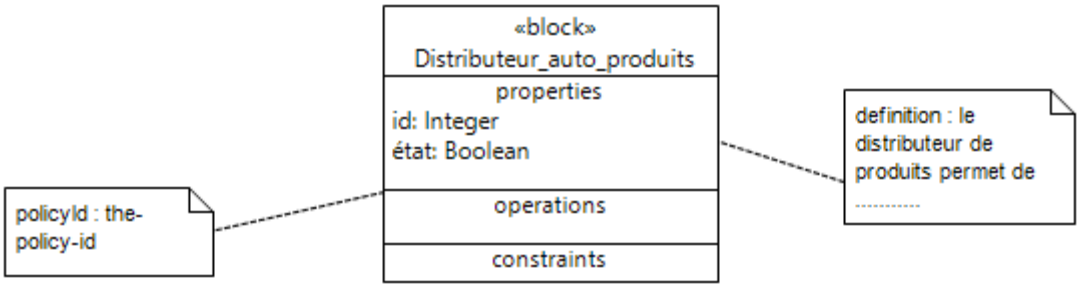


Figure III.7 : Objet annoté par des commentaires

III.4.5. Comment définir les droits d'accès ?

A chaque acteur ou sujet (subject) est associée plusieurs commentaires. Chaque commentaire contient le nom de ressource avec ses droits d'accès.

- Si l'acteur a la permission de lecture ou d'écriture on met « grantRead » ou « grantWrite » respectivement sur une ressource donnée.
- Si l'acteur n'a pas la permission de lecture ou d'écriture on met « revokeRead » ou « revokeWrite » respectivement sur une ressource donnée.

L'acteur a un type (subjectType) et fait partie à une entrée, qui doivent être spécifiés.

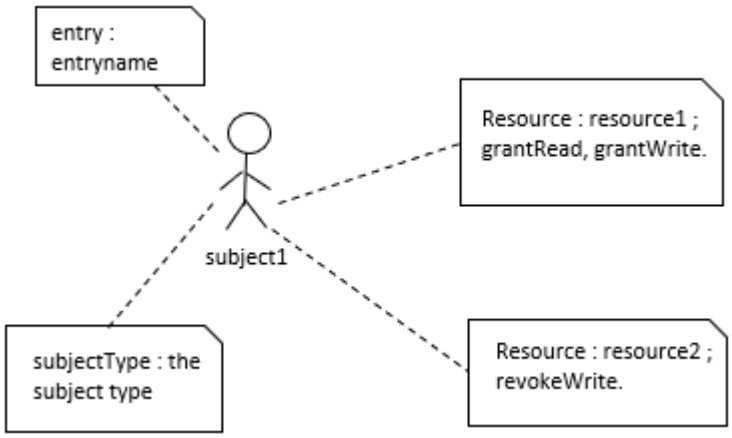


Figure III.8 : Modélisation des droits d'accès

Pour mieux comprendre on donne un exemple : On va dérouler le scenario suivant :

- Propriétaire : c'est le propriétaire de l'objet, alors il a l'accès complet et des droits d'administrateur pour cet objet. Son id est « proprétaireObjet ».
- Observateur du mis-a-jours de ''featureX'' : Une application doit être informée de chaque modification de la fonctionnalités ''featureX''. Son id est « clientObservateur ».

La figure ci-dessous explique comment modéliser ce scenario :

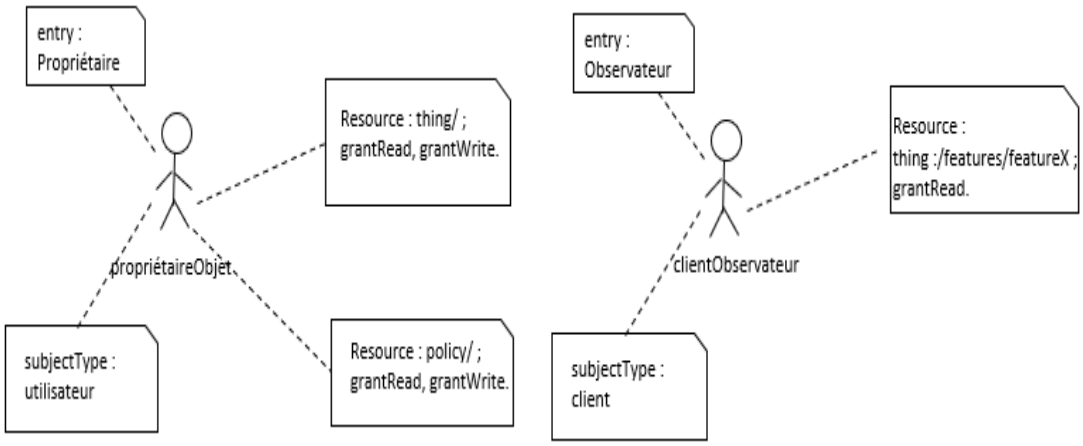


Figure III.9 : Exemple de modélisation des droits d'accès sur un objet

III.5. APPLICATION DE TRANSFORMATION

La transformation est la première étape pour la création de l'image numérique de systèmes réels. Cette transformation prend en entries le model de distributeur automatique de produits présenté par le diagramme de définition des blocs et le méta-modèle de jumeaux numérique et donne en sortie un fichier texte format JSON. La transformation de modèle vers un fichier JSON passe par deux transformations.

-**M2M :** La transformation de modèle de bloc vers le modèle jumeau numérique, en utilisant le langage de transformation ATL.

-**M2T :** La transformation du modèle de jumeau numérique vers deux fichiers JSON, en utilisant l'outil Acceleo.

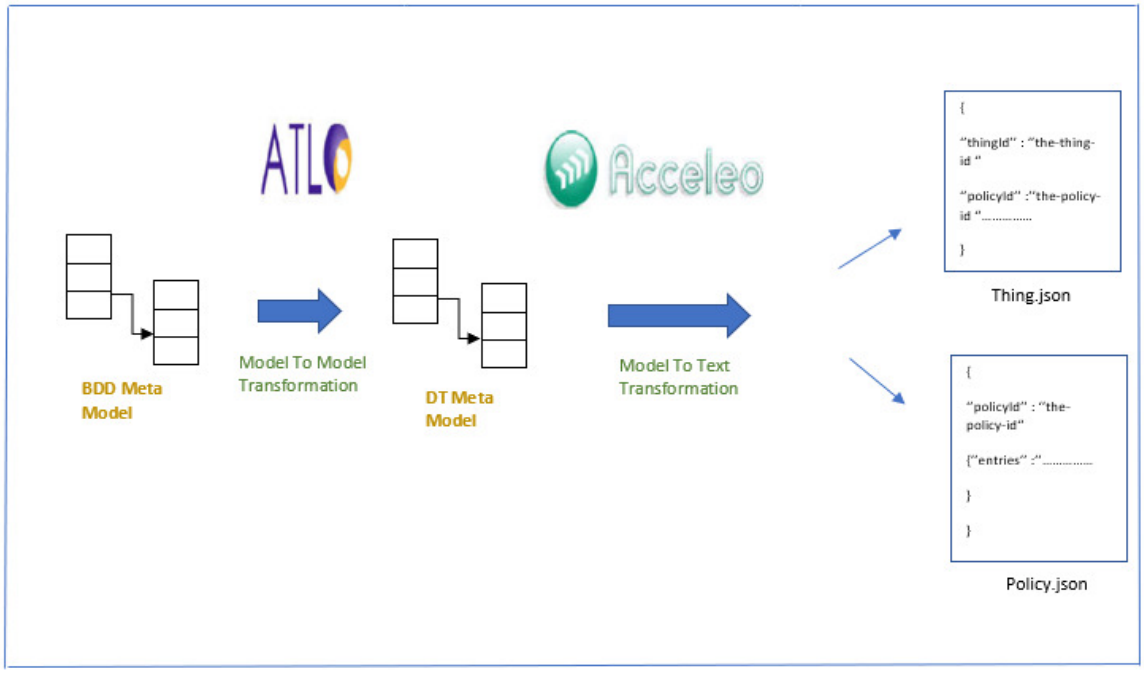


Figure III.10 : Etapes de transformation

III.5.1. Transformation modèle vers modèle

La transformation de diagramme de bloc vers le modèle de jumeau numérique se fait comme suivent :

-La transformation de bloc de distributeur automatique de produits vers un objet (thing) et leurs attributs. Chaque distributeur est présenté comme un thing.

-La transformation des autres blocs vers des fonctionnalités (Feature) et leur propriété (Property).

Explication

- Pour la transformation d'objet (thing), on transforme le nom de bloc qui est distributeur automatique de produits vers le thingId d'objet, et les caractéristiques de ce bloc vers les attributs d'objet (thing). La transformation de ses caractéristiques se fait par la transformation des noms de la caractéristique vers les noms des attributs et les types des caractéristiques vers les types des attributs. Comme le tableau suivant l'indique :

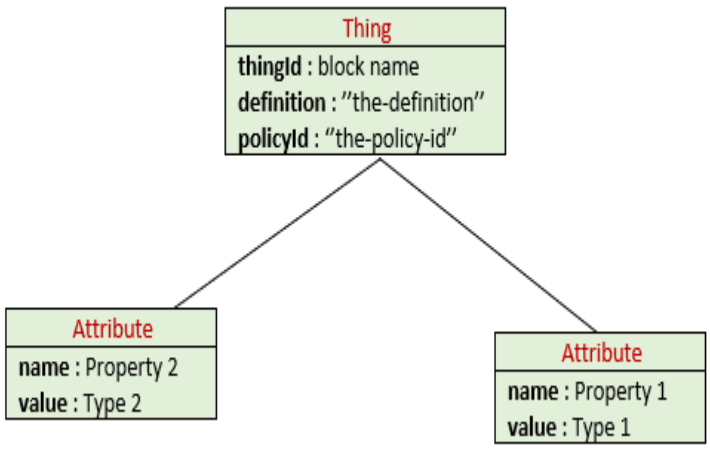
BDD	Représentation jumeaux numérique
« block » block name Properties Property 1 : Type 1 Property 2 : Type 2 operations constraints	

Tableau III.1 : Transformation d'objet

- On transforme les autres blocs restent vers des fonctionnalités (features). On a transformées les noms des blocs vers les noms des fonctionnalités, et les caractéristiques de bloc vers les propriétés (property) de la fonctionnalité (Feature). Pour la transformation des caractéristiques, il y a deux types des propriétés (simple et complexe).
- Pour les propriétés simples, on transforme les noms des caractéristiques de bloc vers les noms des propriétés d'objet et les types des caractéristiques vers les types des propriétés, et pour les propriétés complexes, on transforme les noms de bloc vers les noms de la propriété complexe, et les caractéristiques de bloc vers des propriétés de la propriété complexe. Comme le tableau suivant l'indique :

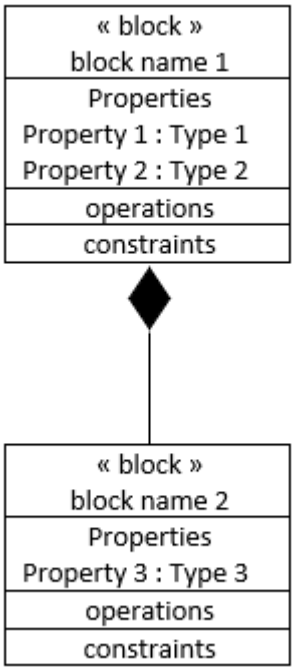
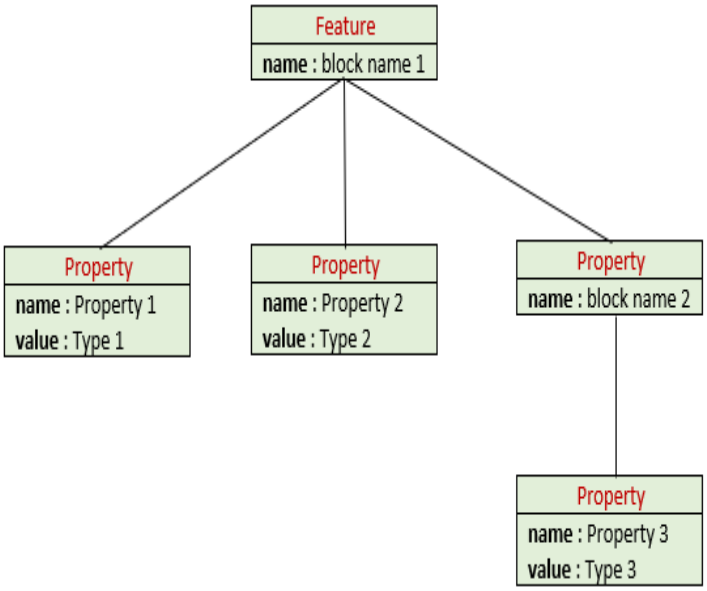
BDD	Représentation jumeaux numérique
	

Tableau III.2 : Transformation des fonctionnalités

Détail : Cette transformation donne un modèle conforme au méta-modèle de jumeau numérique en format xml.

Exemple :

La figure ci-dessous représente un modèle jumeau numérique obtenu par la transformation M2M d'un distributeur automatique de produits secs.

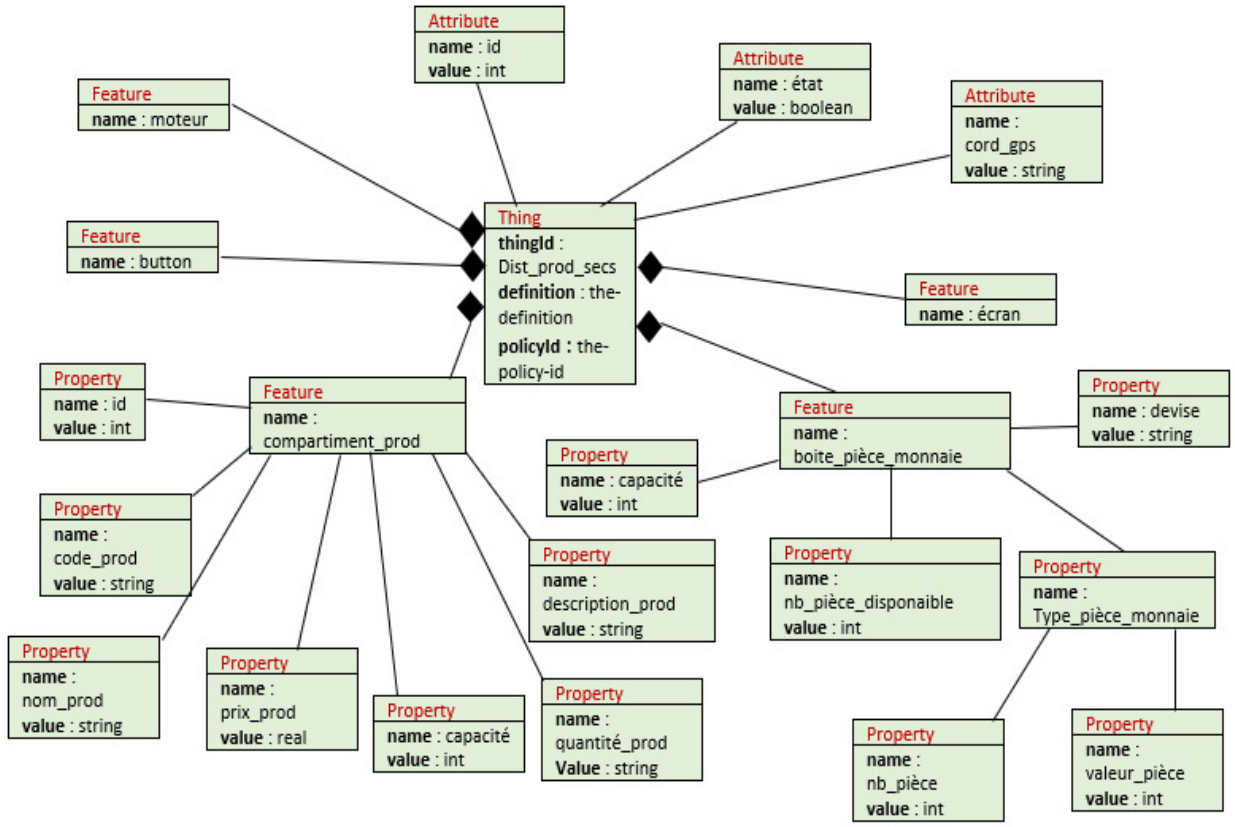


Figure III.11 : Modèle jumeau numérique d'un distributeur automatique des produits secs

III.5.2. Transformation modèle vers texte

La deuxième étape de la transformation qui est la transformation de modèle de jumeau numérique vers deux fichiers texte en format JSON. Cette étape nécessite le modèle obtenu par la transformation M2M, on transforme chacune de ses composants vers un autre composant dans le format JSON. Cette transformation se fait par l'outil acceleo en utilisant eclipse.

Comme on a utilisé la plateforme de API 2, a la fin de transformation on obtient deux fichiers format JSON (thing.json et policy.json).

```
{
    "thingId": "www.eclipse.org:Distributeur_prod_sec",
    "policyId": "www.eclipse.org:policy-a",
    "definition": "definition1",
    "attributes": { "id":0 ,
                    "etat":true ,
                    "cordonne_gps":"Undefined"
    },
    "features": {
        "Compartiment_produits": {
            "properties": {"id":0,
                           "capacite":0,
                           "code_prod":"Undefined",
                           "nom_prod":"Undefined",
                           "description_prod":"Undefined",
                           "quantite_prod":"Undefined",
                           "prix_prod":0.0}},
        "Boite_piece_monnaie": {
            "properties": {"capacite":0,
                           "devise":"Undefined",
                           "nb_piece_disponaible":0,
                           "type_piÃ¨ce_monnaie": {
                           "valeur_piece":0}}},
        "Moteur": { "properties": {}},
        "Button": { "properties": {}},
        "Ecran": { "properties": {}
        }
    }
},
```


Figure III.12 : Fichier format JSON d'un objet

```
{
    "policyId": "www.eclipse.org:policy-a",
    "entries": {
      "administrateur_app": {
        "subjects": {
          "nginx:ditto": {
            "type": "nginx basic auth user"
          }
        },
        "resources": {
          "thing:/": {
            "grant": ["READ", "WRITE"],
            "revoke": []
          },
          "policy:/": {
            "grant": ["READ", "WRITE"],
            "revoke": []
          },
          "message:/": {
            "grant": ["READ", "WRITE"],
            "revoke": []
          }
        }
      },
      "propriétaire_dist_auto": {
        "subjects": {
          "nginx:propiétaire": {
            "type": "user"
          }
```

Figure III.13 : Fichier format JSON d'une policy

Dés la terminaison de ces transformations, on va commencer de développer une application web pour créer les jumeaux numériques.

III.6. APPLICATION DE SUPERVISION A TRAVERS LES JUMEAUX NUMERIQUES

Le but de la création de cette application est de faciliter la création des jumeaux numériques pour les systèmes des distributeurs automatiques des produits par les entreprises, ainsi pour leurs aider à contrôler et superviser les distributeurs facilement.

III.6.1. Objectif de l'application

Les objectifs de développement de cette application sont les suivants :

- ✓ La création d'interface de jumeau numérique de système de distributeurs automatiques de produits et la connecter à la plateforme eclipse ditto.
- ✓ Capter tous changement sur le système physique (système de distributeurs automatiques de produits).
- ✓ Echange des données entre les deux jumeaux et être capable d'évoluer en temps réel.
- ✓ facilite contrôle et la correction automatique
- ✓ Diminution du temps de changement de prix d'un produit.
- ✓ Développements des interfaces d'affichages des distributeurs et leur emplacement.
- ✓ La gestion de localisation des nouveaux distributeurs automatiques.

III.6.2. Diagramme de cas d'utilisation

Un diagramme de cas d'utilisation capture le comportement d'un système tel qu'un utilisateur extérieur le voit. Il scinde la fonctionnalité du système en unités cohérentes, les cas d'utilisation, ayant un sens pour les acteurs. Les cas d'utilisation permettent d'exprimer le besoin des utilisateurs d'un système, ils sont donc une vision orientée utilisateur de ce besoin au contraire d'une vision informatique.

Dans notre cas, nous avons trois acteurs qui sont :

- ❖ L'administrateur qui est le responsable de la création et la suppression des jumeaux numériques et la transformation de model-jumeau.
- ❖ Le superviseur pour la modification des prix des produits.
- ❖ Le client pour la consultation et recherche des produits.

Ci-dessous, nous présentons le diagramme de cas d'utilisation de système de distributeur automatique des produits, en montrant les cas d'utilisation d'un utilisateur qui peut l'utilise (voir Figure 14).

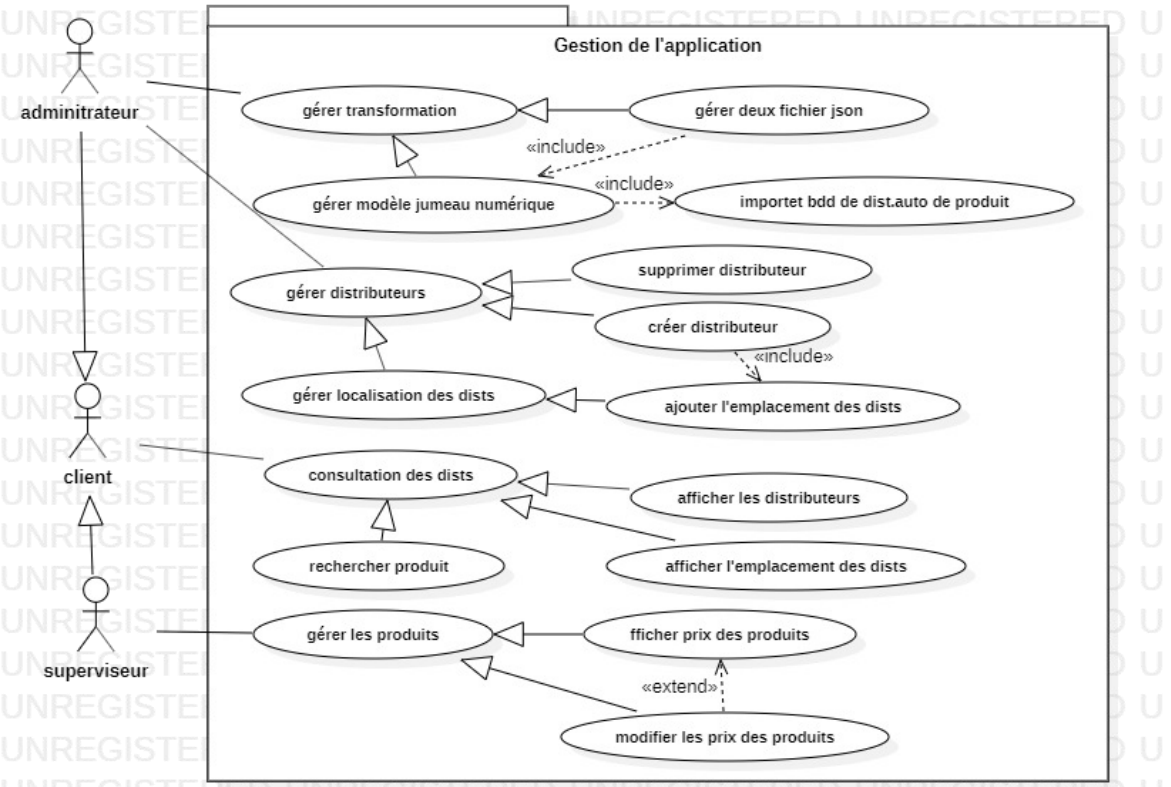


Figure III.14 : Diagramme de cas d'utilisation

Scénario :

- La gestion de l'interface des jumeaux numériques nécessite une exécution d'une série de transformation. Cela nécessite la sélection de diagramme de bloc de système de distributeurs automatiques de produits par l'administrateur, pour pouvoir faire la transformation M2M et générer un modèle jumeau numérique conforme au méta-modèle de la plate forma utilisée. Une deuxième transformation va se passer dés que la première transformation sera fini pour obtenir la présentation en format JSON du modèle jumeau numérique (on obtient deux fichier JSON, un fichier JSON pour le thing et l'autre pour la création des policies).

- Le cas de gestion de distributeur (thing) permet la création après le saisi de l'emplacement, ou la suppression d'un distributeur par l'administrateur. Cela nécessite de connecter à la plate forme de jumeaux numériques.

- Le cas de gestion des produits permet l'affichage des produits pour la modification de prix d'un produit par le superviseur.

- Le cas de consultation des distributeurs permet l'affichage des distributeurs avec leurs emplacements par les trois types d'utilisateurs (administrateur, superviseur, client). Ainsi la recherche des produits.

III.6.3. Diagramme de séquence

Les diagrammes de cas d'utilisation modélisent à QUOI sert le système, en organisant les interactions possibles avec les acteurs. Par contre les diagrammes de séquences permettent de décrire COMMENT les éléments du système interagissent entre eux et avec les acteurs :

- Les objets au cœur d'un système interagissent en s'échangent des messages.
- Les acteurs interagissent avec le système au moyen d'IHM (Interfaces Homme-Machine).

La figure III.15, présente les séquences à effectuer pour la gestion de l'application.

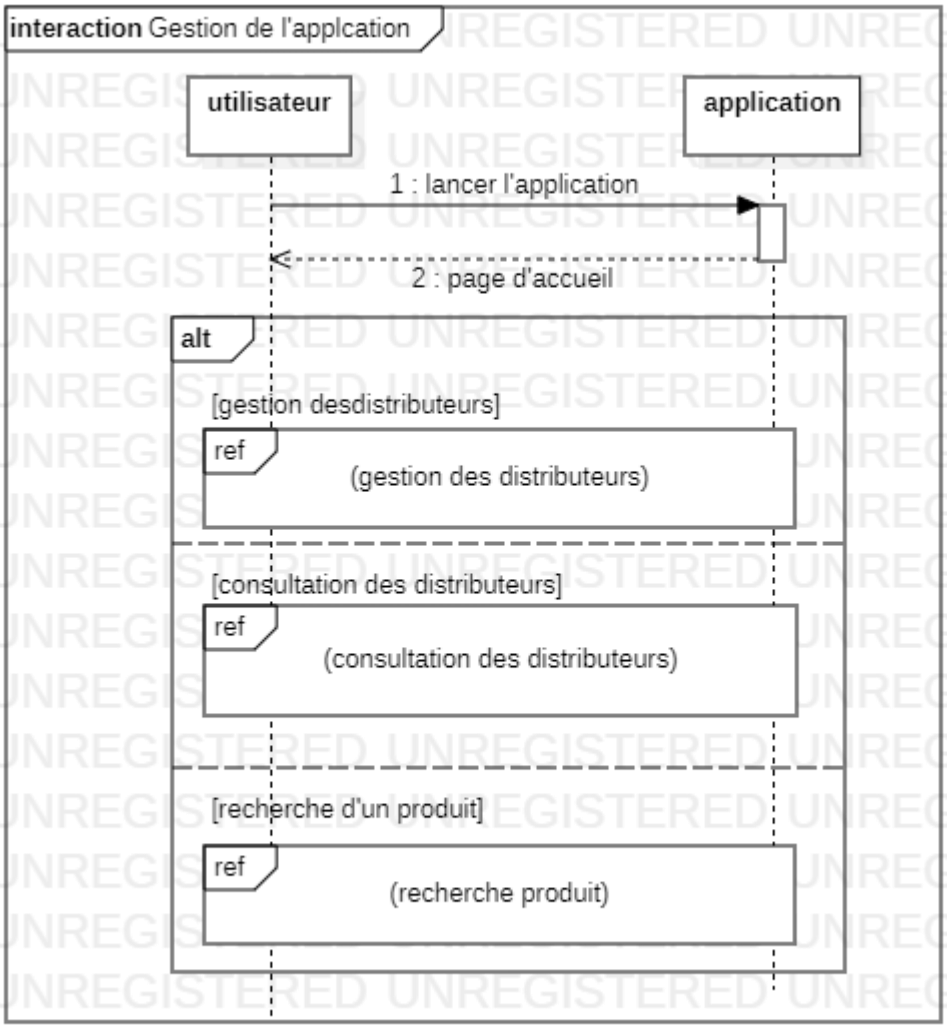


Figure III.15 : Diagramme de séquence de gestion de l'application

La figure III.16, montre les séquences à effectuer pour la gestion d'un distributeur automatique.

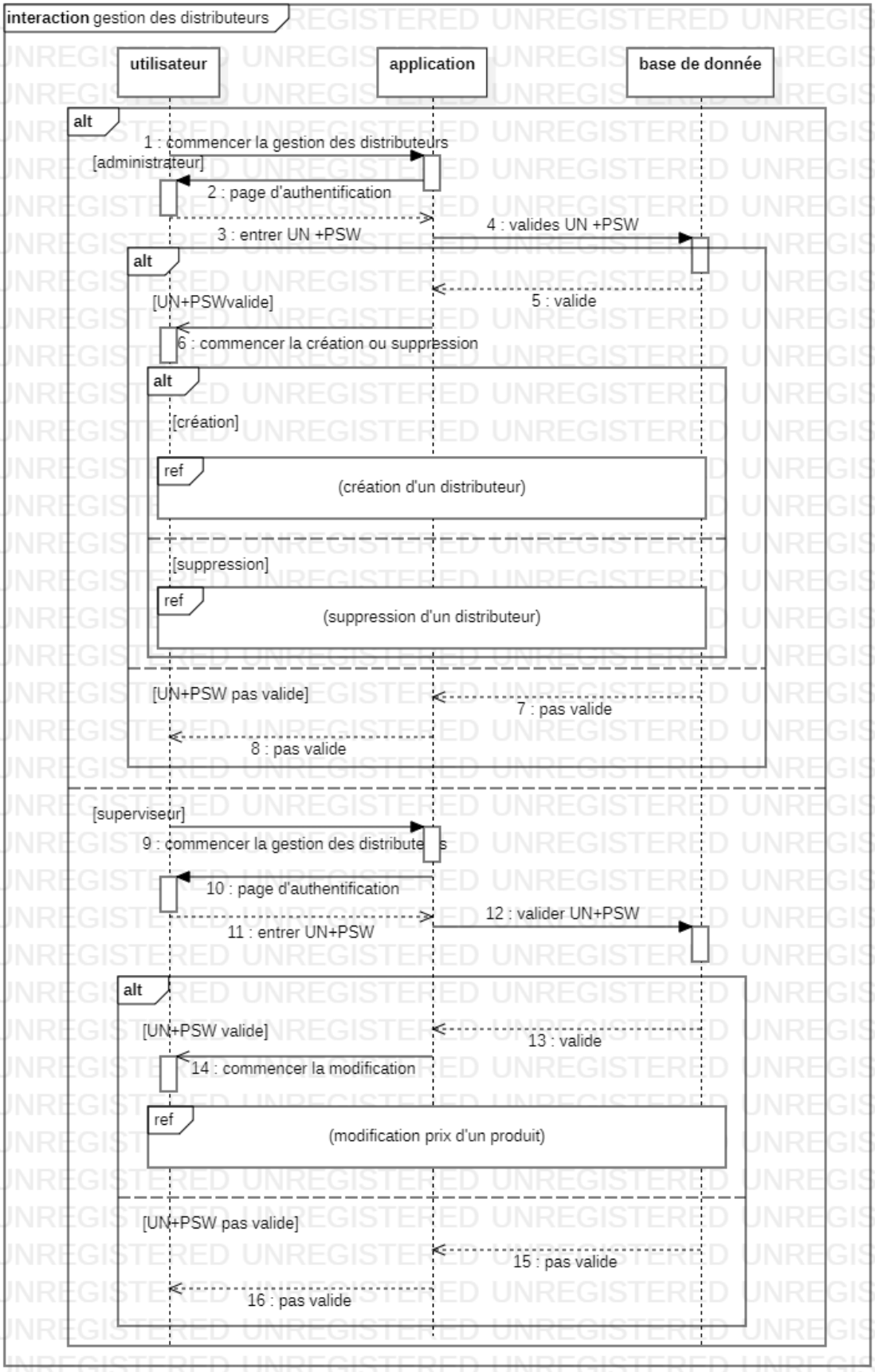


Figure III.16 : Diagramme de séquence de gestion de distributeur

La figure III. 17, montre les étapes à suivi pour entamer la phase de gestion de transformation.

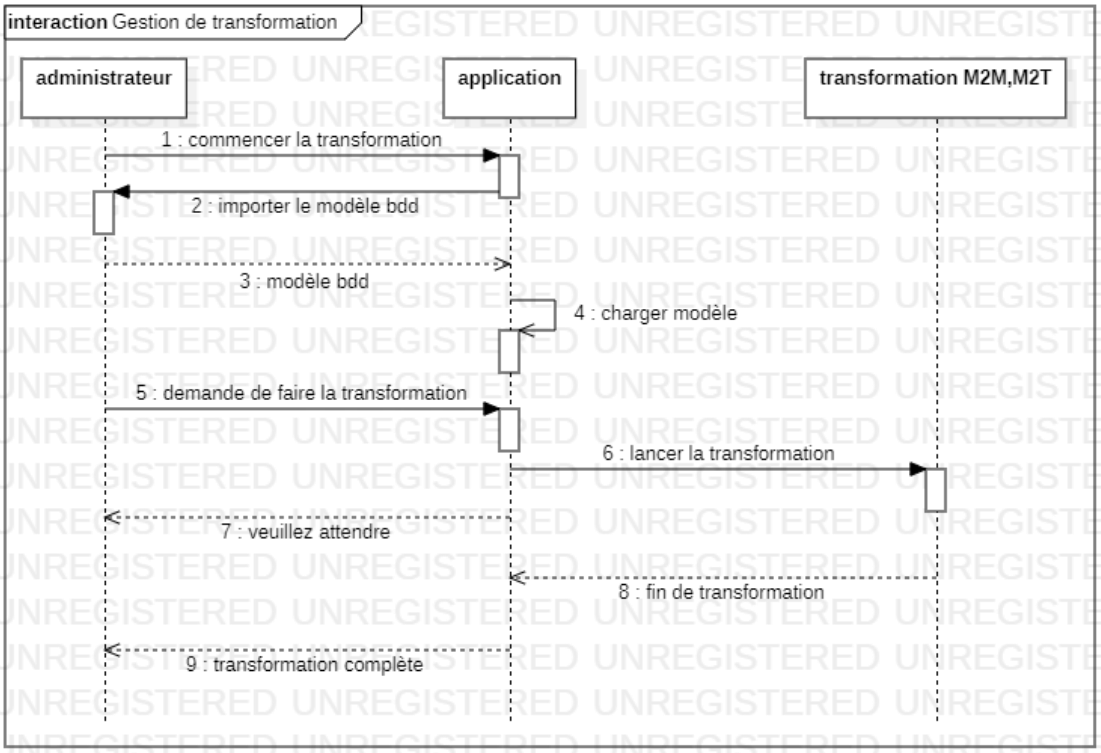


Figure III.17 : Diagramme de séquence de gestion de transformation

La figure III.18, présente les séquences de suppression d'un distributeur.

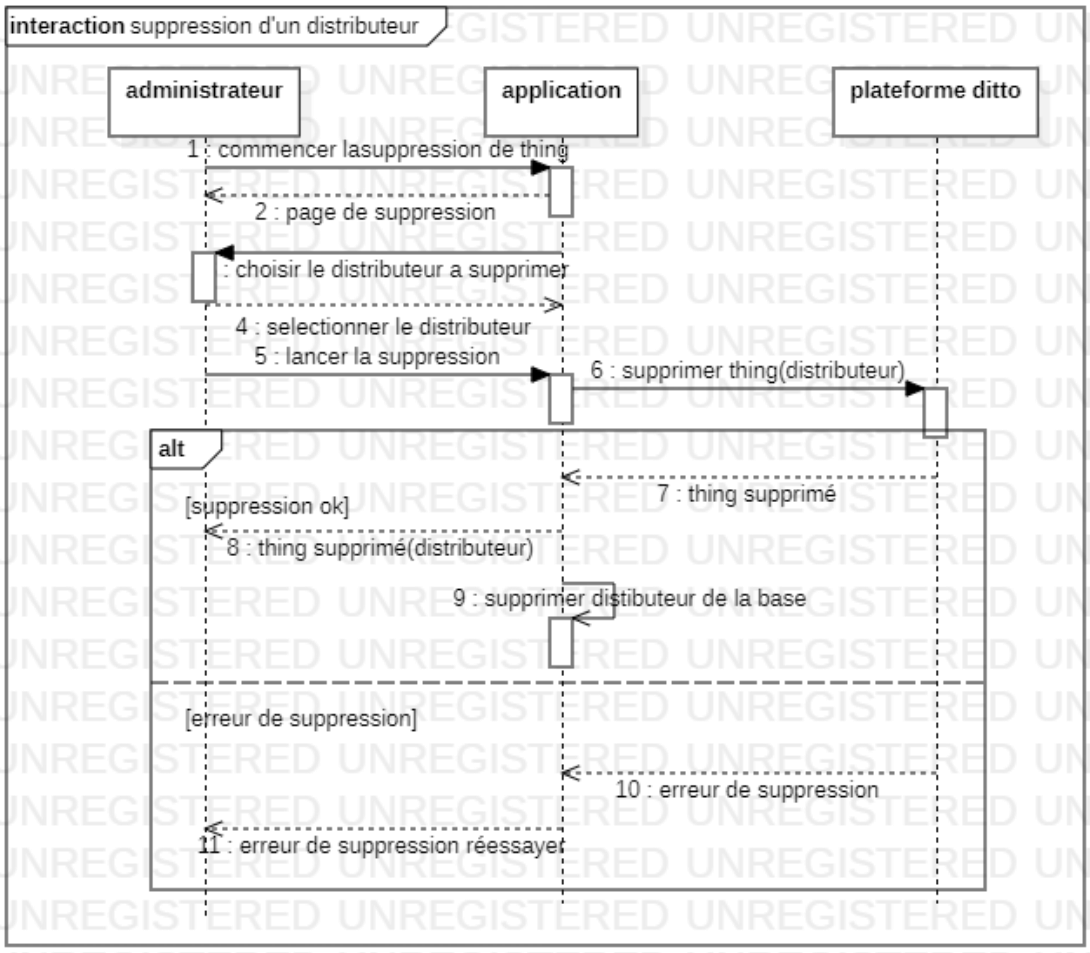


Figure III.18 : Diagramme de séquence de suppression de distributeur

La Figure III.19, montre les séquences de consultation des distributeurs.

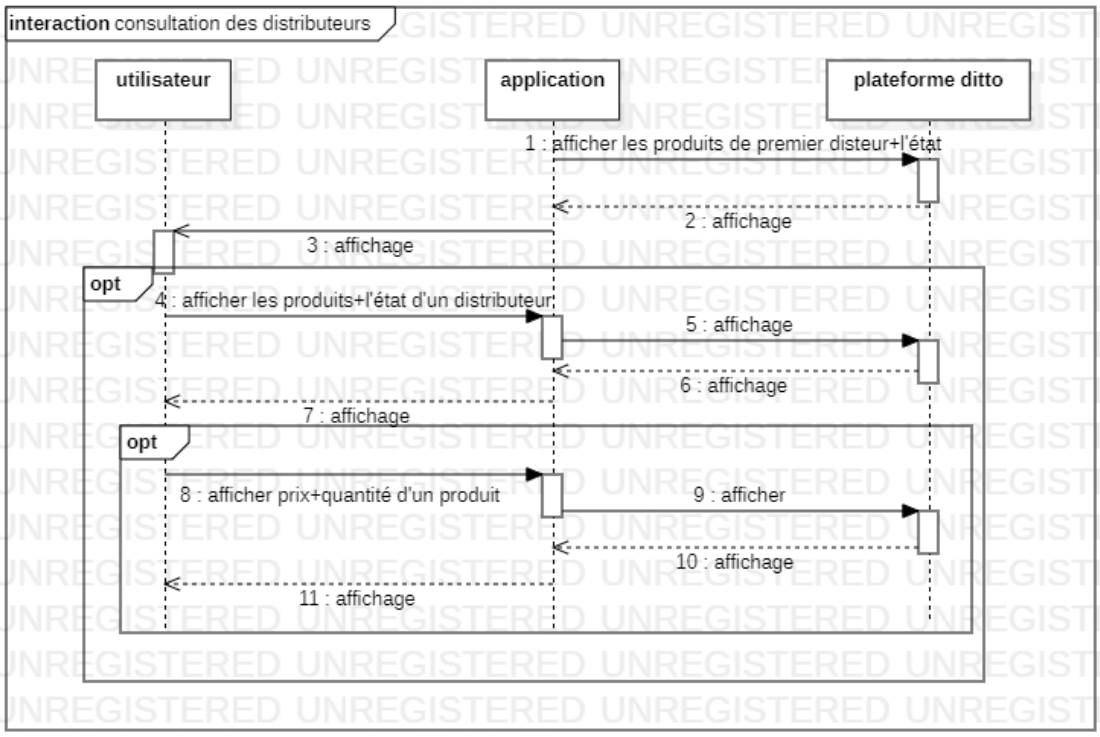


Figure III.19 : Diagramme de séquence de consultation des distributeurs

La figure III.20, montre les séquences a suivi dans la recherche d'un produit.

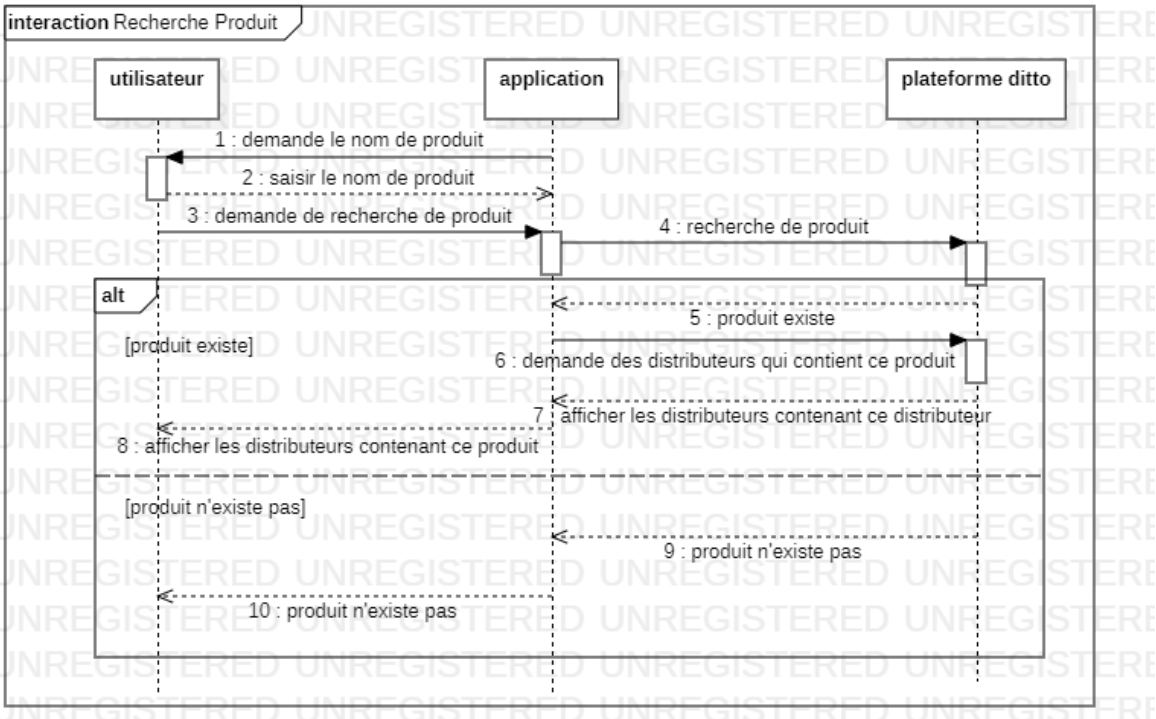


Figure III.20 : Diagramme de séquence de la recherche d'un produit

La figure III.21, montre les étapes de création d'un distributeur.

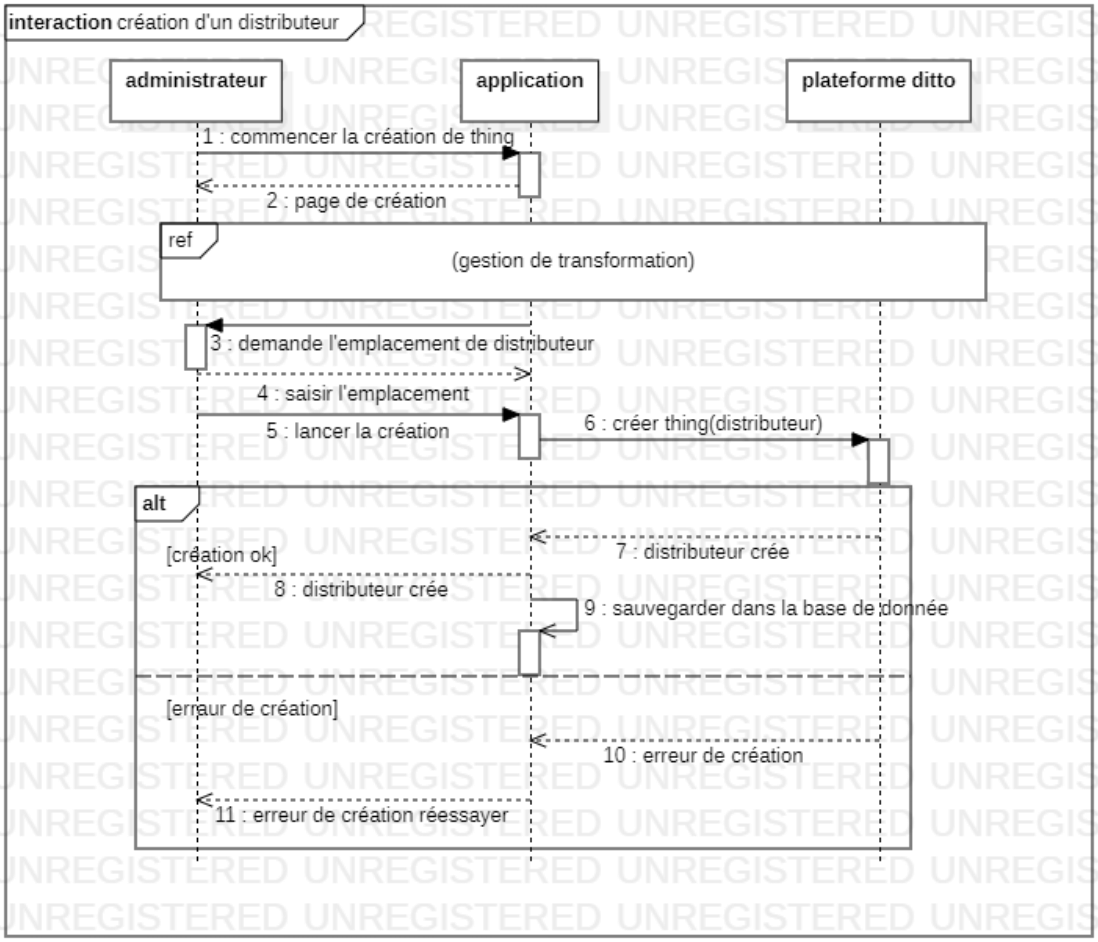


Figure III.21 : Diagramme de séquence de la création d'un distributeur

La figure 22, montre les étapes de modification de prix d'un produit

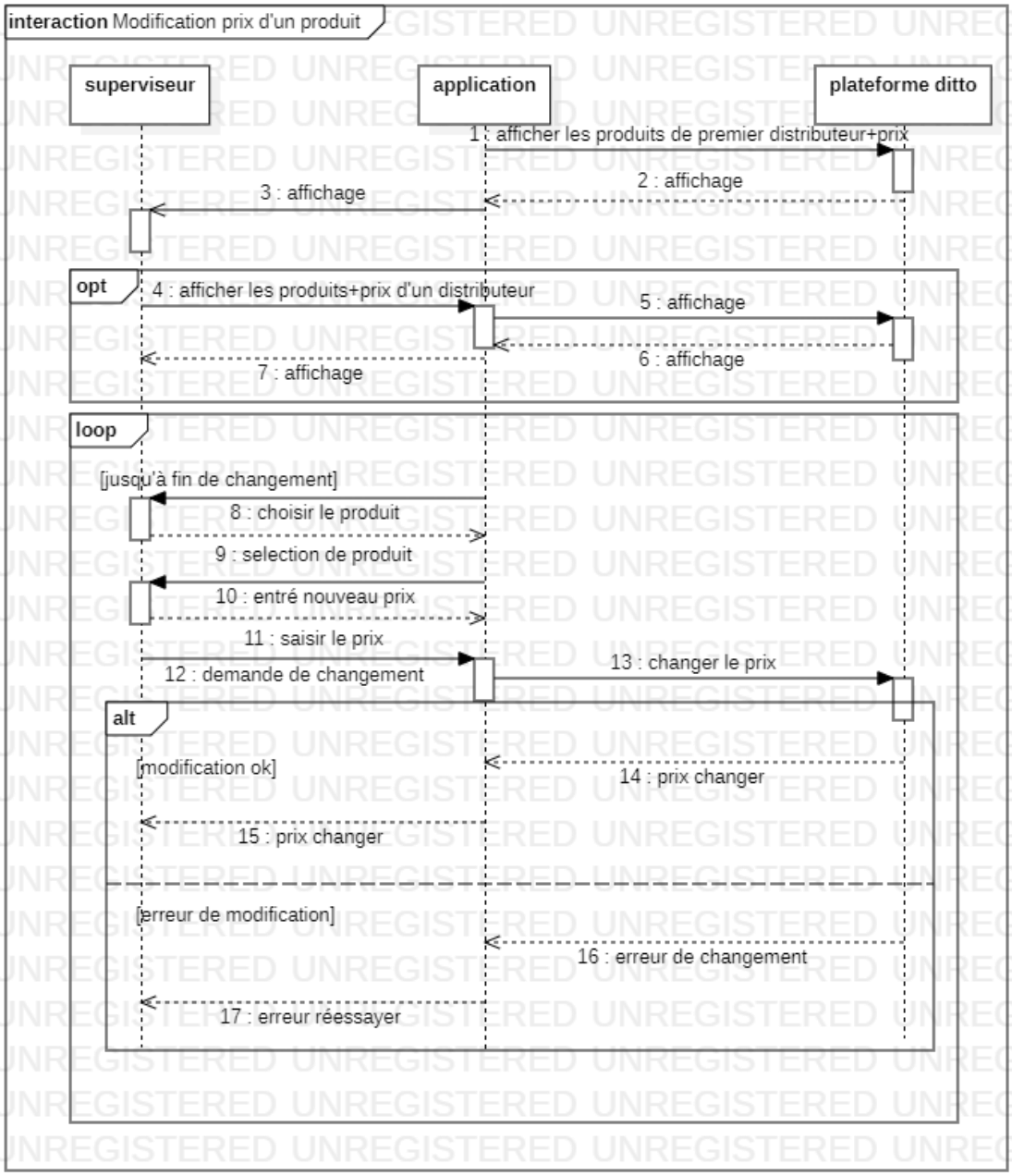


Figure III.22 : Diagramme de séquence de la modification de prix d'un produit

III.6.4. Intégration avec le système des distributeurs automatiques des produits

Le but de l'application web est de créer l'interface des jumeaux numérique pour le système de distributeurs automatiques de produits, en passant le modèle en entré da l'application qui va se passera sur une série des transformations pour générer deux fichier format JSON. Le système est lié avec une plateforme des jumeaux numérique « Eclipse Ditto ».

L'interface des jumeaux numérique crée une connexion avec la plateforme Eclipse Ditto et lui permet de capter tous changement sur les composants de système de distributeurs automatiques (système réel) au moment que les changements se produisent, aussi pour contrôler la durée de vie et de la maintenance des produits et systèmes, suivre et corriger les modifications facilement. L'échange des informations se fait à l'aide de l'outil node-red.

III.7. CONCLUSION

Dés les opérations de transformation modèle vers modèle et modèle vers texte sont terminer, On développe une application web pour créer l'image numérique de système réel de distributeurs automatiques des produits et contrôler toutes les modifications avec des capteurs.

CHAPITRE IV

Réalisation et test

IV.1. INTRODUCTION

Nous arrivons dans ce chapitre à la description de l'aspect pratique de notre travail. Notre but est l'implémentation de la démarche proposée dans le chapitre conception et méthodologie. Pour cela, nous avons développé une application web pour réaliser toutes étapes de notre conception. L'application web implémentant notre méthodologie facilite le contrôle et la supervision des distributeurs automatiques de produits à travers leurs images numériques. Nous décrivons aussi, dans ce chapitre l'ensemble des moyens technologiques utilisés dans le développement de notre plateforme.

IV.2. IMPLEMENTATION

Cette phase est composée de trois parties. La première partie est la transformation des diagrammes de bloc vers le modèle du jumeau numérique. La deuxième partie est la transformation du modèle du jumeau numérique vers deux fichiers JSON contient les propriétés de ce modèle, et finalement l'application web proposée pour la création du jumeau numérique et le contrôle du système de distributeurs automatiques de produits.

IV.2.1. Modélisation

La première des choses, pour la modélisation de système de distributeurs automatiques de produits via le diagramme de bloc, nous avons utilisées l'outil papyrus intégré dans PolarSys, qui est un environnement de modélisation de haut niveau basé modèles qui permet de mettre en place une ingénierie "dirigée" par les modèles. L'outil support les métalangages UML et SysML, et autorise la création de langages de modélisation spécifiques à une industrie ou à un domaine applicatif. Il offre aussi la possibilité de travailler sous plusieurs aspects : la simulation basée modèles, le test basé modèles, l'analyse de sûreté et l'exploration d'architecture [28]. Les figures (Figure 1, Figure2, Figure3) montrent les commentaires associés aux blocs et acteurs.

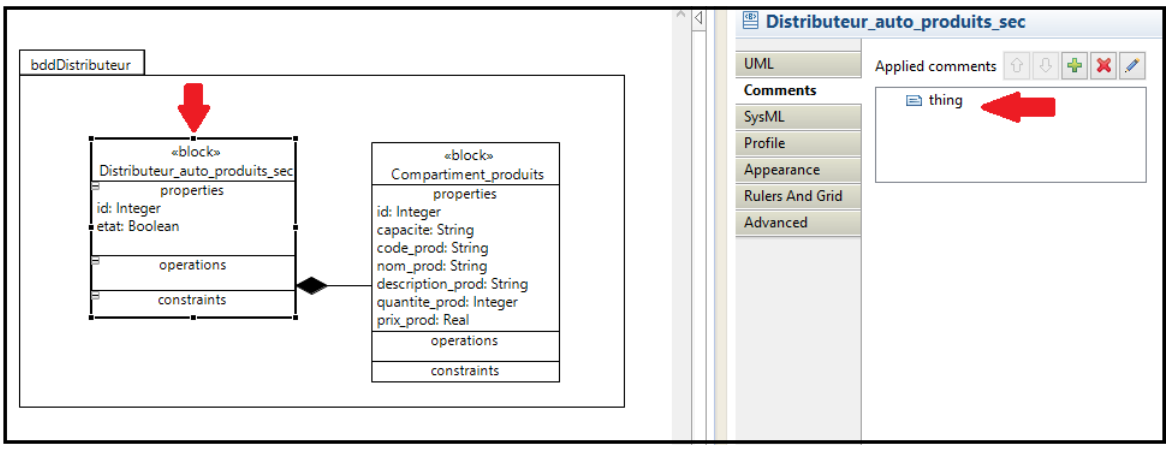


Figure IV.1 : Annotation de Thing

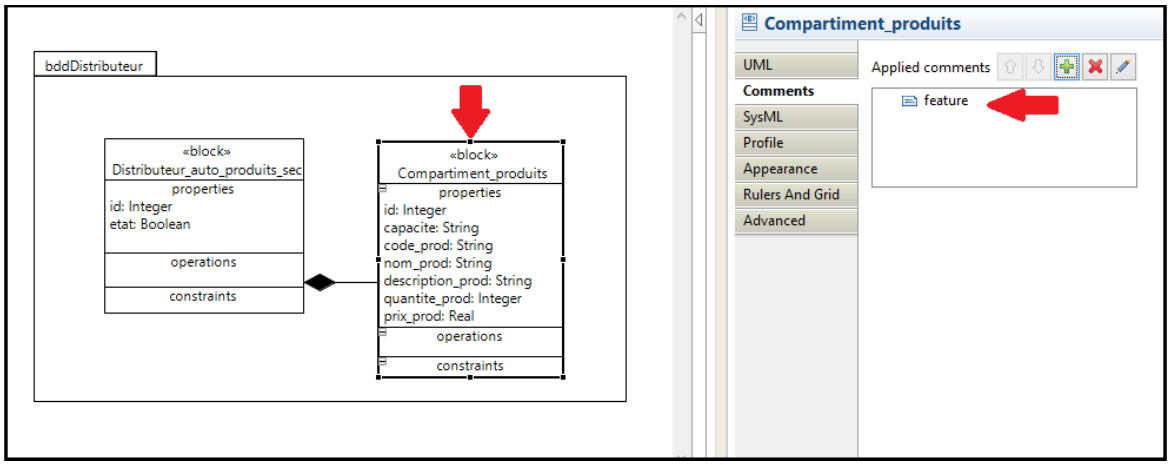


Figure IV.2 : Annotation des features

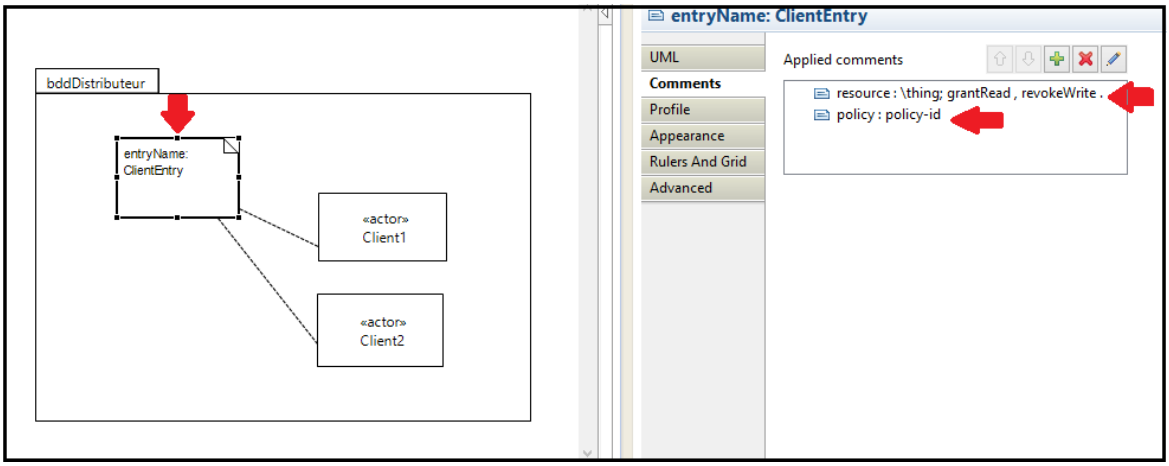


Figure IV.3 : Annotation de policy

IV.2.2. Transformation

L'éxécution de transformations de modèles nécessités d'un langage de transformation Qui pourra être défini via un méta-modèle de transformation

IV.2.2.1. Transformation ATL

Nous avons utilisé le langage de transformation atlas ATL (atlas transformation langage) qui est considéré comme un standard de transformation dans Eclipse, Ceci pour réaliser la transformation des concepts de diagramme de bloc vers les concepts du modèle de jumeau numérique, nous avons transformées chaque bloc de modèle via des règles.

❖ Transformation des objets

Nous avons utilisées la règle ATL suivante (distTOthing) pour transformer les distributeurs vers des objets.

```
58
59  rule distTOthing {
60      from
61          c :MM1!Class(c.ownedComment-> select(it | it.body = 'thing').notEmpty())
62
63      to
64          DT : MM2!Thing  (
65              thingId <- c.name,
66              definition <- c.ownedComment -> select(m | m.body.startsWith('definition'))
67              attributes <- (c.primitive() ->collect(t | thisModule.getatribute(t))).union
68              features <- c.helpergetFeature()->collect(e| thisModule.getfeature(e)),
69              policies <-  c.ownedComment -> select(j | j.body.startsWith('policy')) ->col
70
71              )
72  }
73
```

Figure IV.**4 :** Règle de transformation des objets

❖ Transformation des fonctionnalités

Nous avons utilisées la règle ATL suivante (blockTOfeature) pour transformer les blocs ayant le commentaire « feature » vers des fonctionnalités du jumeau numérique.

```
75
76  lazy rule blockTOfeature {
77      from
78          f :MM1!Class
79      to
80          ff : MM2!Feature  (
81              name <- if(f.name='Compartiment_produits') then 'Compartiment_produits'
82              properties <- (f.primitive() -> collect(p | thisModule.getproperty(p)))
83               )
84  }
85
```

Figure IV.**5 :** Règle de transformation des fonctionnalités

❖ Transformation des policy

Nous avons utilisé la règle ATL suivante (getPolicy) pour transformer les droits d'accès d'objets vers les concepts du jumeau numérique.

```
141
142 lazy rule getPolicy {
143     from
144         a : MM1!Comment(a.body.startsWith('policy'))
145     to
146         b : MM2!Policy (
147             policyId <- a.body.regexReplaceAll('policy : ', ''),
148             entries <- a.annotatedElement->select(f | f.oclIsTypeOf(MM1!Comment)) -> collect(h | thisModule.getEntry(h))
149             )
150 }
```

Figure IV.**6 :** Règle de transformation des policy

IV.2.2.2. Transformation ACCELEO

Afin de terminer la première transformation, nous avons générées deux fichiers JSON à l'aide d'un générateur de code source de la fondation Eclipse qui s'appelle ACCELEO.

❖ Transformation des droits d'accès et des objets

Les deux règles acceleo présentés dans la figure 7 et la figure 8, prend en entré le modèle de jumeau numérique sous forme xml pour générer les deux fichiers json.

```
[template public generateElement(aPolicy : Policy)]
[comment @main/]
[file ('policy.json', false, 'UTF-8')]
        {
           "policyId": "www.eclipse.org:[aPolicy.policyId/]",
           "entries": {
             [for (e : entriesname | aPolicy.entries)separator (',\n')]
             "[e.name/]": {
                "subjects": {
                   [for (s : Subject | e.subjects)separator (',\n')]
                   "[s.name/]": {
                   "type": "[s.type/]"
                   }[/for]

                },
                "resources": {
                   [for (r:Resource | e.resources)separator (',\n')]
                   "[r.name/]": {
                   "grant": [if (r.grant.READ = true and r.grant.WRITE = false)]['['/]"READ"[']'
                   "revoke": [if (r.revoke.READ = true and r.revoke.WRITE = false)]['['/]"READ"[
                   }[/for]

                }
             }[/for]

     }
}[/file]
[/template]
```

Figure IV.**7 :** Règle ACCELEO pour transformation des droits d'accès

```
[template public generateElement(aThing : Thing)]
[comment @main/]
[file ('thing.json', false, 'UTF-8')]
        {
            "thingId": "www.eclipse.org:[aThing.thingId/]",
            "policyId": "www.eclipse.org:[aThing.policies.policyId/]",
            "definition": "[aThing.definition/]",
            "attributes": {
                [for (att : Attribute | aThing.attributes)separator (',\n\t\t\t\t')]"[att.nam
            },
            "features": {
                [for (f : Feature | aThing.features) separator (',\n')]
                "[f.name/]": {
                    "properties": {
                        [for (pro : propertie | f.properties) separator (',\n\t\t\t\t\t\t')]"
                    }
                }[/for]

            }
        }[/file]
[/template]
```

Figure IV.**8 :** règle ACCELEO pour transformation des objets

IV.2.3. Intégration des transformations

Pour que les processus des transformations peuvent être exécutés par l'application web, nous avons besoin d'exporté les projets M2M et M2T sous la forme d'un fichier jar. Pour exporter les fichiers jar, nous avons suivi les étapes suivantes :

Nous avons crée un nouveau projet « plugin » a partir de projet atl. Ce nouveau projet est exporté au fichier jar.

File –> New –> ATL Plugin.

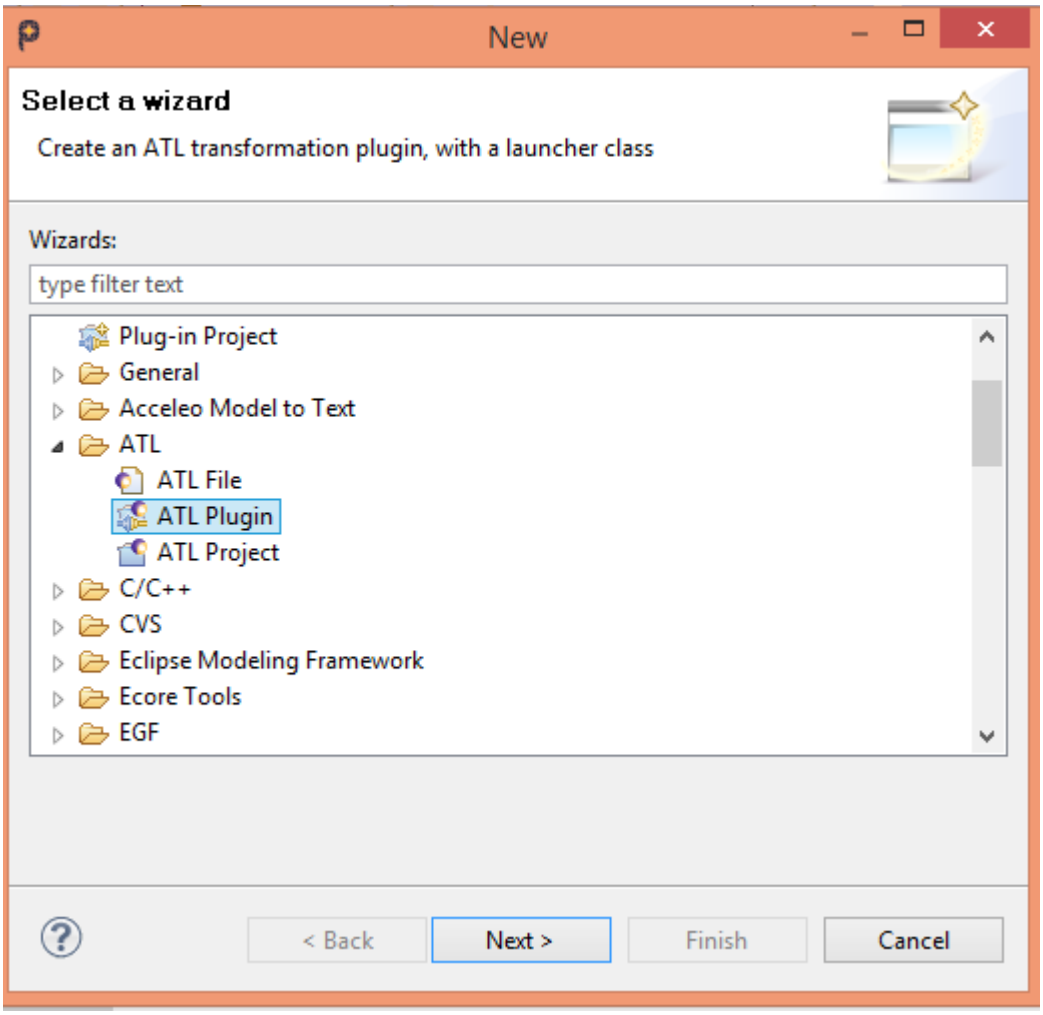


Figure IV.9 : Création de projet plugin

Pour exporter le projet plugin : Export –> Runnable jar file

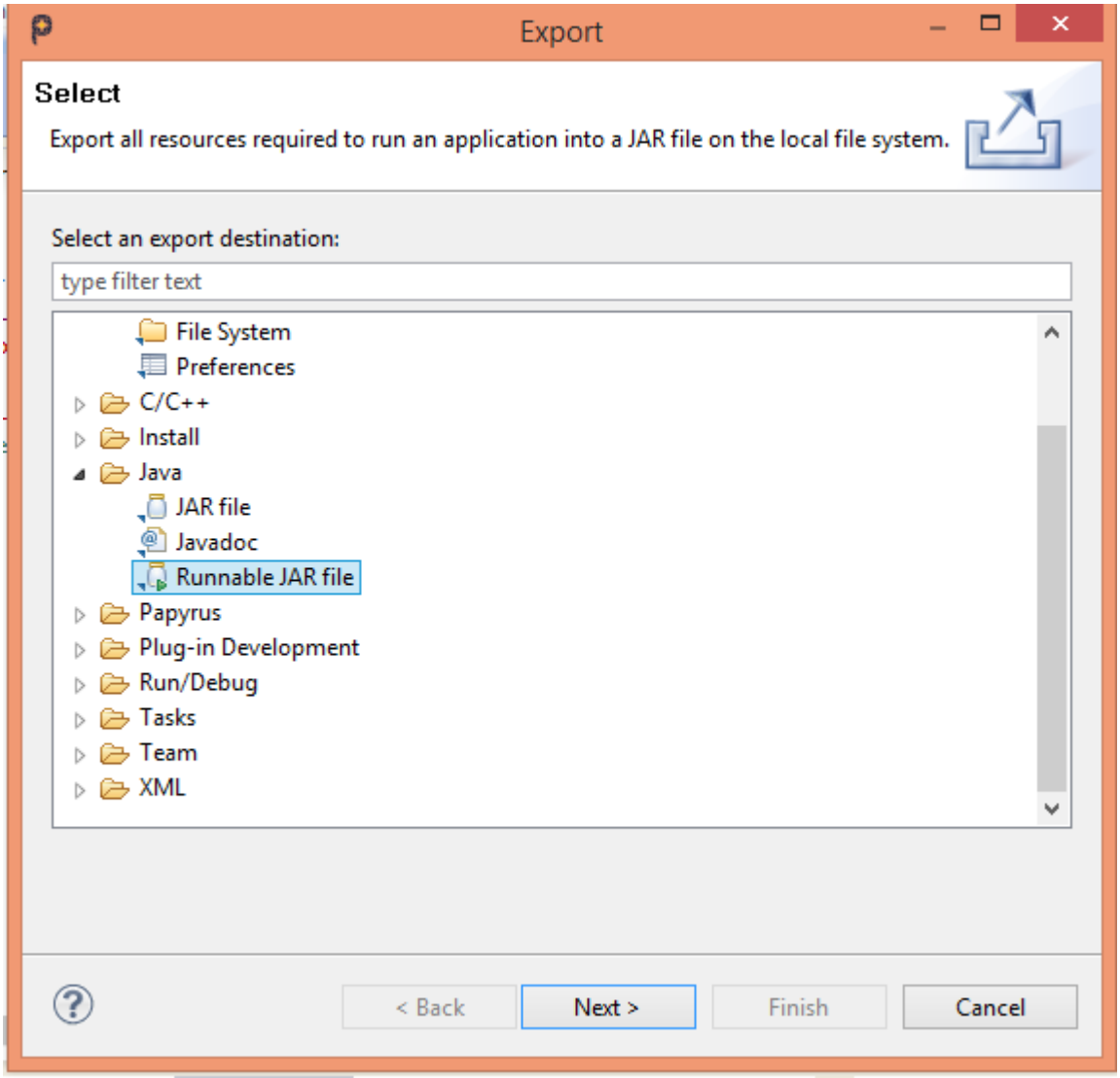


Figure IV.10 : Conversion à fichier jar

IV.2.4. Plateforme des jumeaux numériques utilisées

Nous avons utilisé dans notre travail la plateforme Eclipse Ditto qui est une technologie de l'IOT mettant en œuvre un modèle logiciel jumeau numérique [29]. Il est utilisé pour produire et fournir des outils pour la réalisation des logiciels, englobant les activités de programmation en s'appuyant principalement sur Java. IL sert comme un intermédiaire entre le système réel et la représentation numérique.

IV.3. ETUDE DES CAS

Dans l'étude de cas, nous avons fait la simulation du système de distributeur automatique de produits a l'aide de l'outil nodered et nous avons expliquées les interfaces de l'application web.

IV.3.1. Simulation de système des distributeurs automatiques des produits

Nous avons utilisé l'outil nodered pour la simulation et la modélisation de déférentes fonctionnalités de distributeur automatique des produits.

La figure IV.11, illustre la simulation des fonctionnalités d'un compartiment de produit.

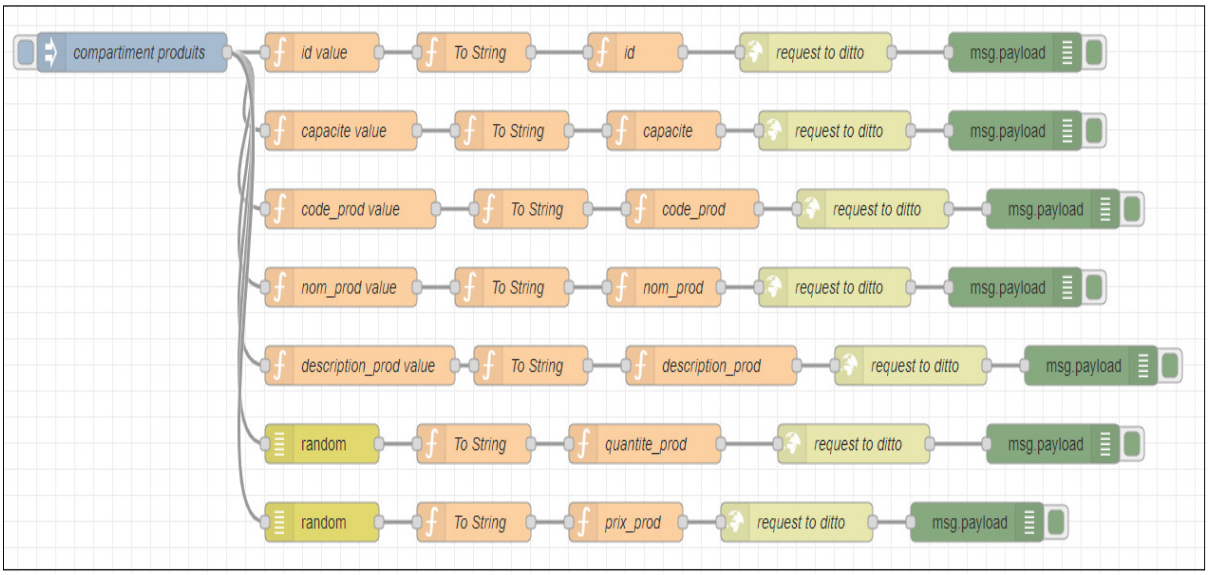


Figure IV.11 : Simulation des fonctionnalités du compartiment des produits

La Figure IV.12, illustre la simulation de l'ajout d'un produit auxiliaire.

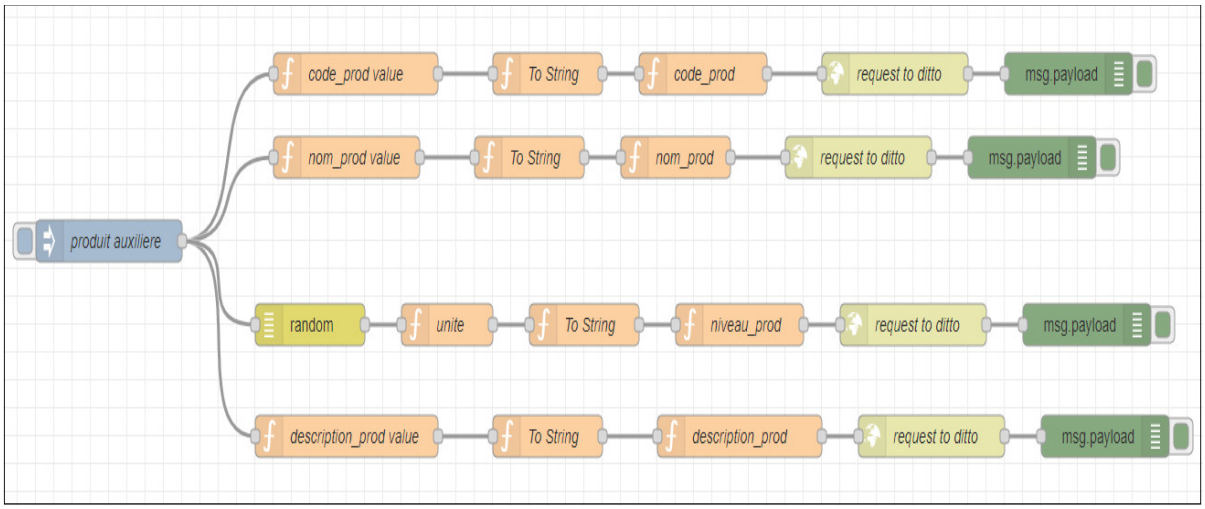


Figure IV.12 : Simulation de l'ajout des produits auxiliaires

La Figure IV.13, illustre la simulation de l'ajout d'un produit livré.

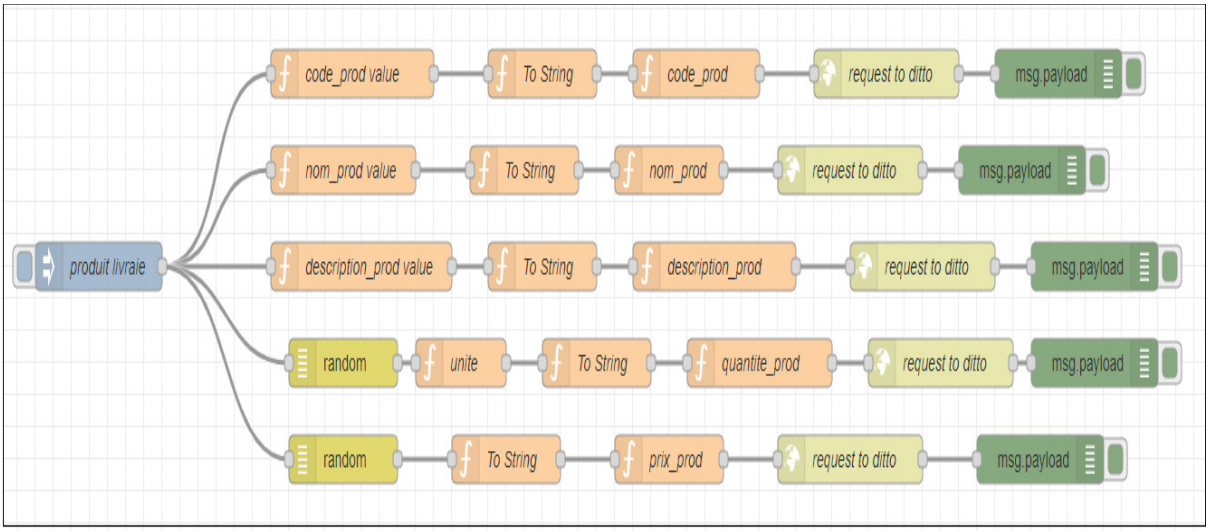


Figure IV.13 : Simulation de l'ajout des produits livrés

La Figure IV.14, illustre la simulation de l'achat d'un produit.

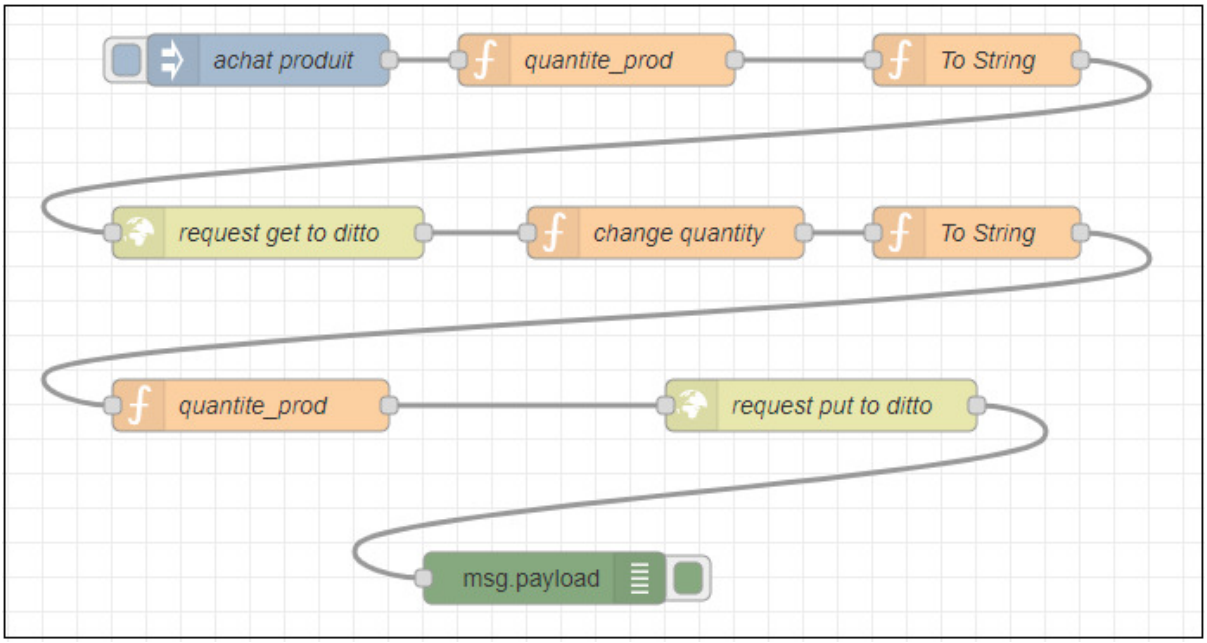


Figure IV.14 : Simulation de l'achat des produits

IV.3.2. Interfaces de l'application Web

Nous avons utilisé plusieurs langages dans la réalisation de notre application qui sont les suivants : le langage HTML[30] pour représenter les pages web, le php[30] et

JAVASCRIPT[30] pour produire des pages dynamiques, CSS[30] et BOOTSRAP[30] pour le style, phpmy admin pour la création de base de données et le serveur web local XAMPP.

IV.3.2.1. Interface d'accueil

L'interface de la page d'accueil de l'application web permet à l'utilisateur de :

- Savoir plus sur le site et commencer la gestion des distributeurs comme le montre les Figures IV.15 et IV.16.

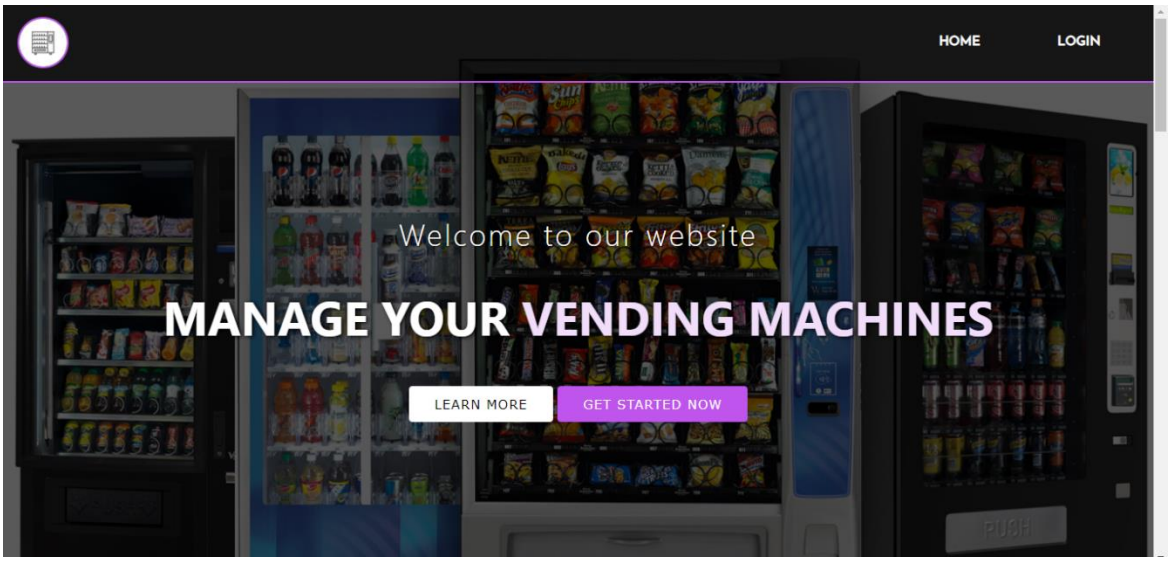


Figure IV.15 : Interface pour commencer la gestion des distributeurs

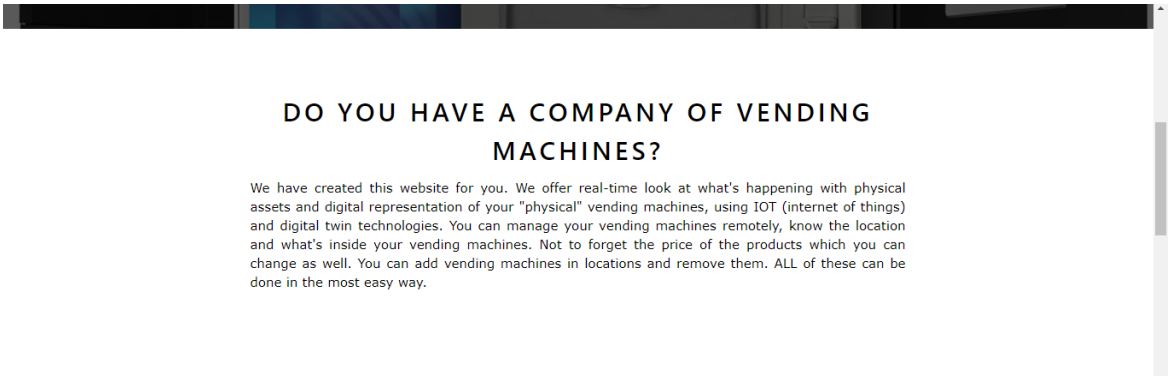


Figure IV.16 : Interface pour savoir plus sur le site

- Consulter la localisation de chaque distributeur automatique de produits comme montre la Figure IV.17.



Figure IV.**17 :** Interface de localisation des distributeurs

- Savoir tous les distributeurs automatiques de produits existants à l'aide d'une liste, leurs états éteint ou allumé, et consulter tous les produits d'un distributeur.

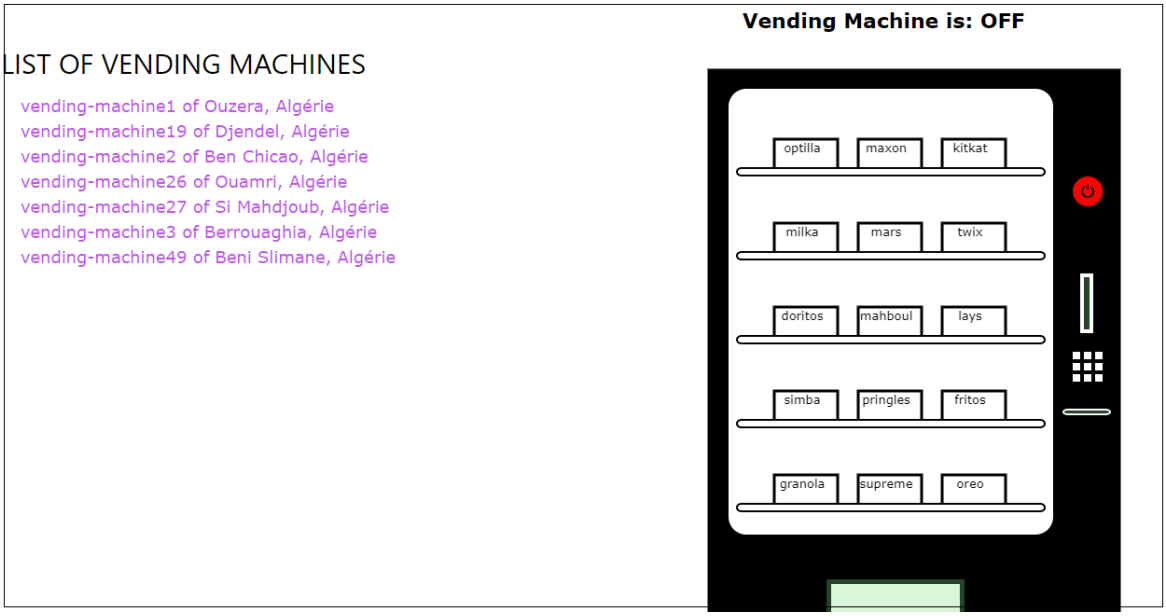


Figure IV.**18 :** Affichage de distributeur des produits secs

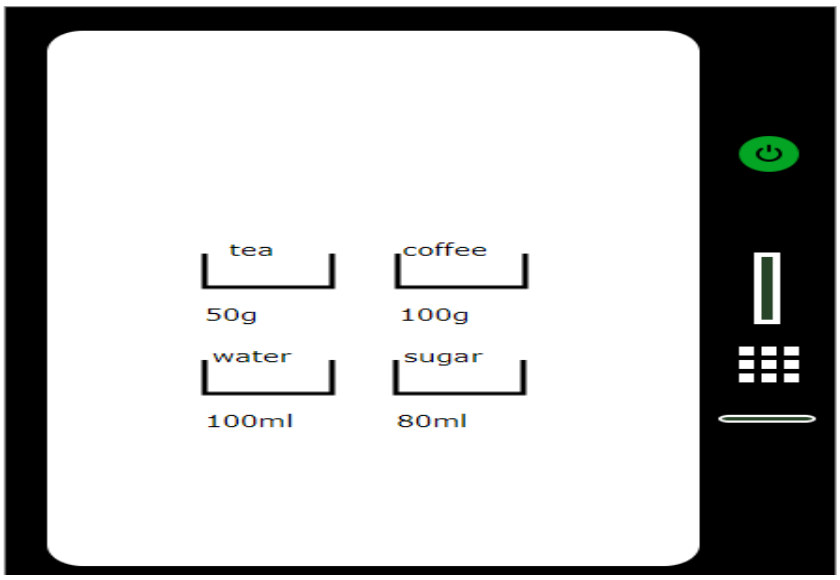


Figure IV.19 : Affichage de distributeur des produits liquides

- Afficher le prix et la quantité des produits lorsque l'utilisateur clique sur un produit :



Figure IV.20 : Affichage des prix et quantités des produits

- Quand le compartiment d'un produit est vide, il s'affiche (en rouge) comme la Figure21 montre.

-

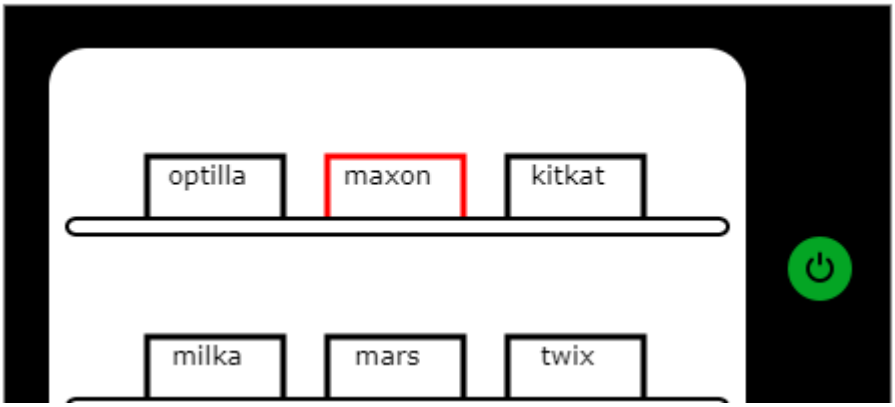


Figure IV.21 : Cas d'un compartiment vide

- La recherche d'un produit : afficher les distributeurs qui continents ce produit s'il existe.



Figure IV.22 : Recherche des produits existants



Figure IV.23 : Recherche des produits non existants

IV.3.2.2. Interface d'authentification

Cette interface s'affiche lorsque l'utilisateur (superviseur, administrateur) veut gérer les distributeurs. L'utilisateur doit saisi son nom et son mot de passe.

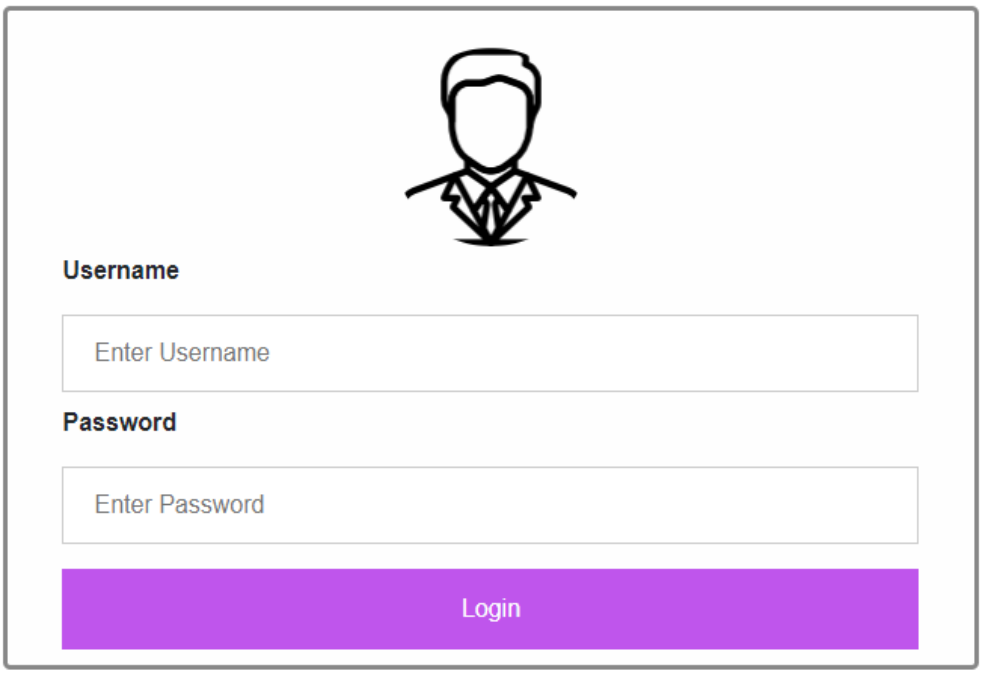


Figure IV.24 : Interface de page d'authentification

IV.3.2.3. Interface de transformation

Cette interface permet à l'administrateur de lancer la transformation M2M et M2T après la sélection de diagramme de blocs de système de distributeur automatique.

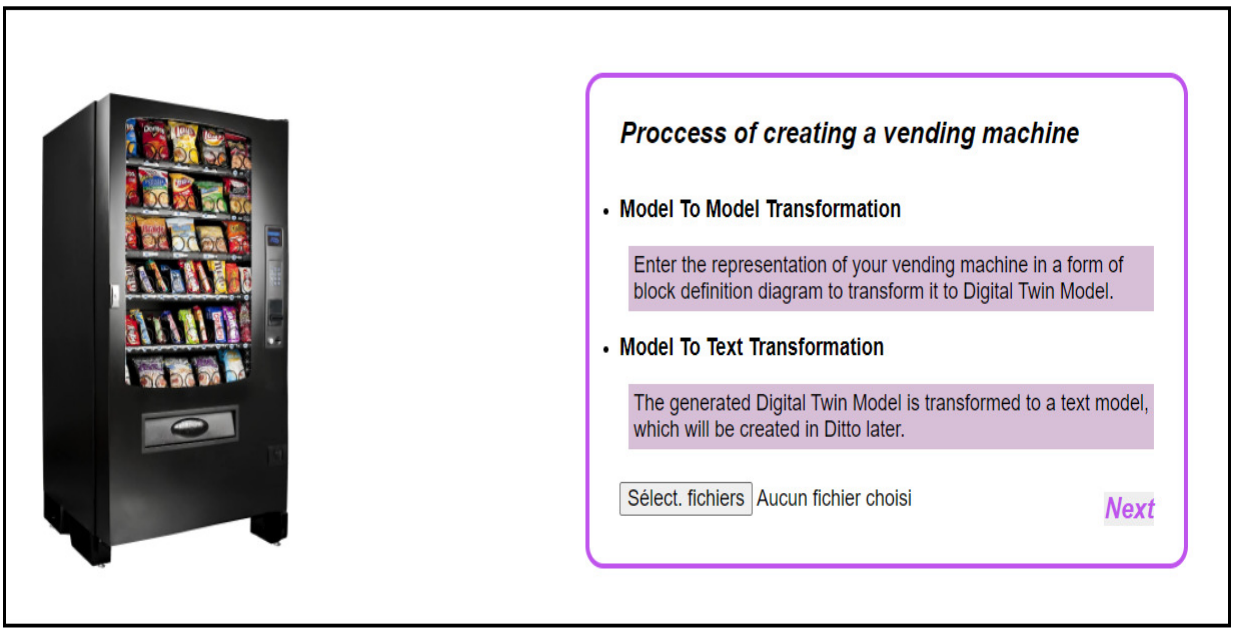


Figure IV.25 : Interface de page de transformation

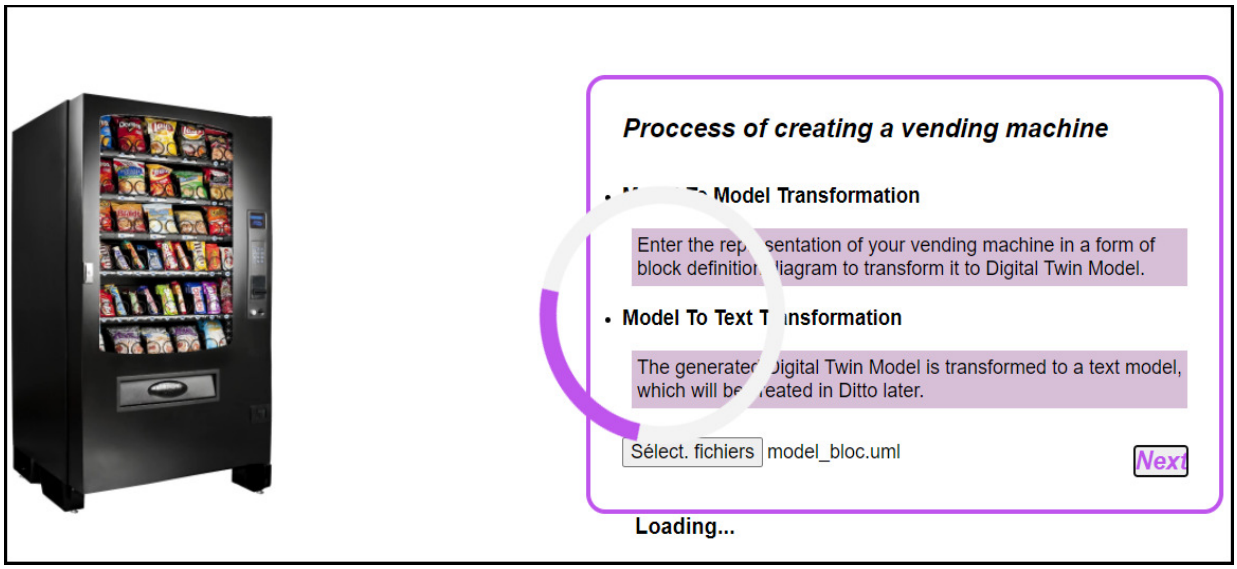


Figure IV.26 : Interface lors de la transformation

IV.3.2.4. Interface de création des distributeurs

Cette interface contient le résultat de la transformation qui sont deux fichiers JSON. L'administrateur doit saisir la localisation et cliquer sur le Button « create » pour crée le distributeur dans la plateforme du jumeau numérique ditto et l'ajouter dans la base de données.

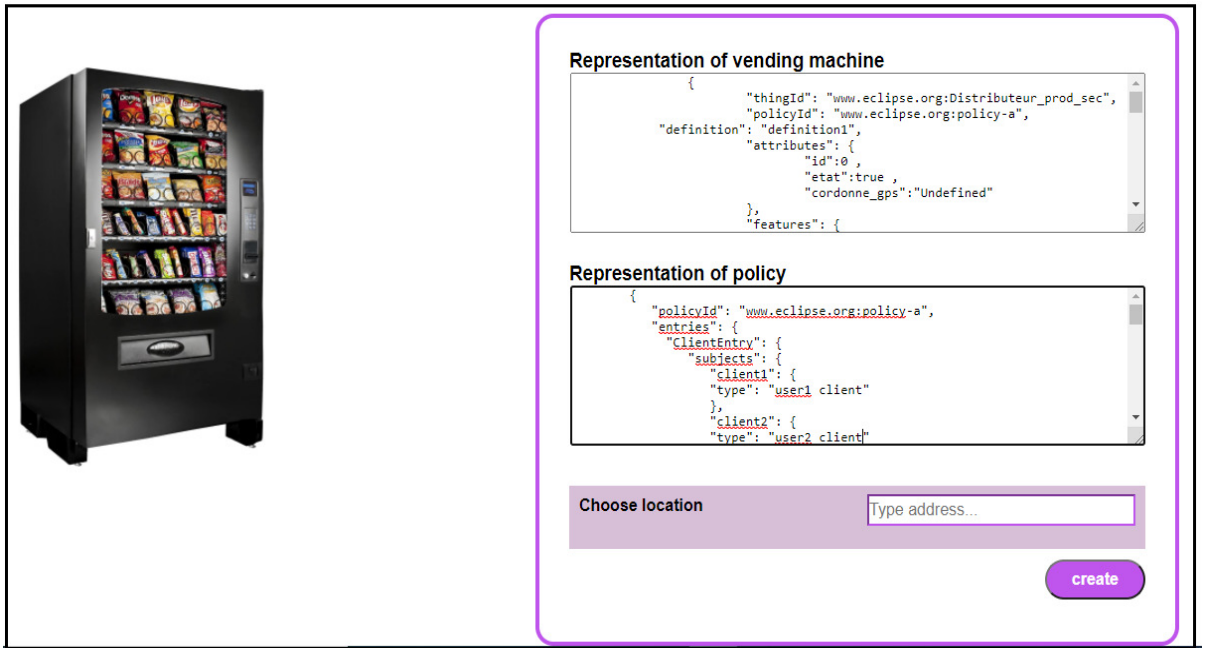


Figure IV.**27 :** Interface de la création d'un distributeur

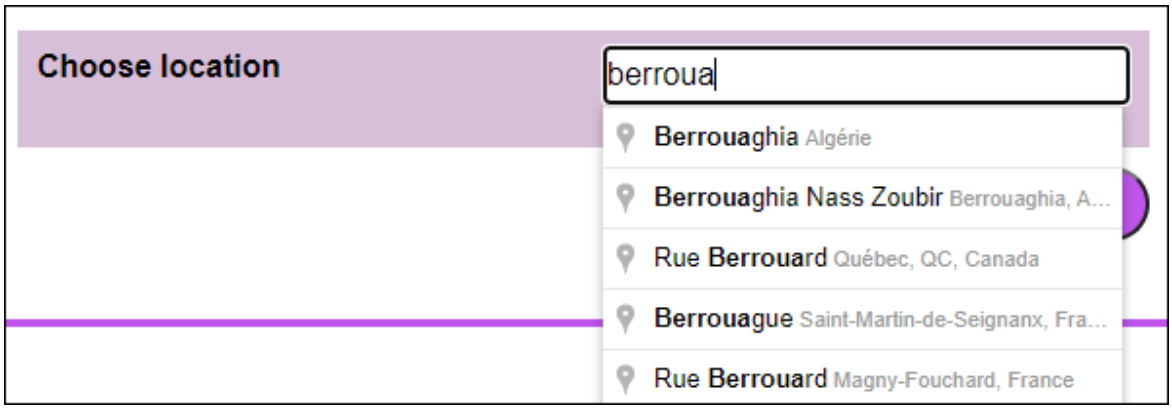


Figure IV.**28 :** Choix de localisation

IV .3.2.5. Interface de suppression des distributeurs

Dans l'interface de suppression une liste de tous les distributeurs apparait avec la possibilité de voir leurs produits, l'administrateur choisi le distributeur à supprimer et clique sur le bouton de suppression.

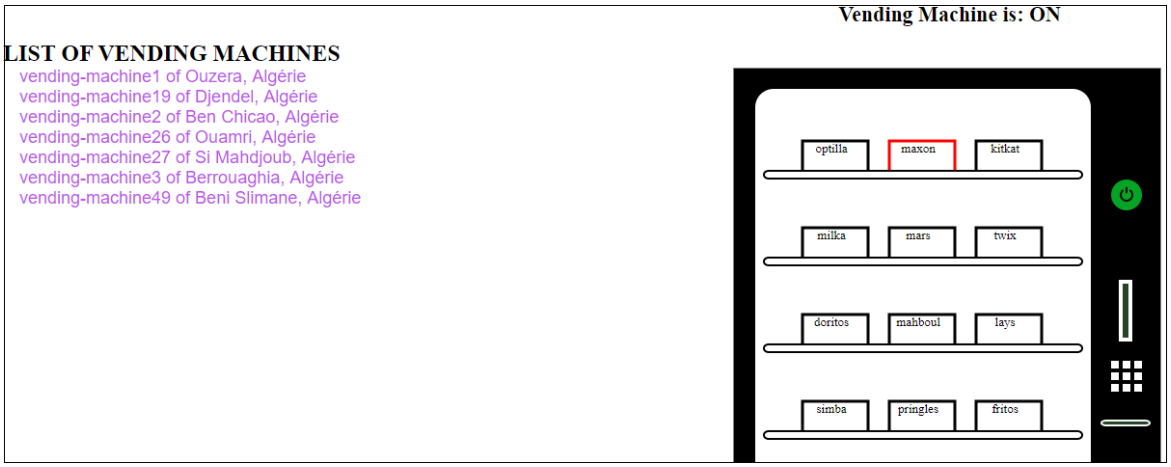


Figure IV.29 : Interface de suppression d'un distributeur 1

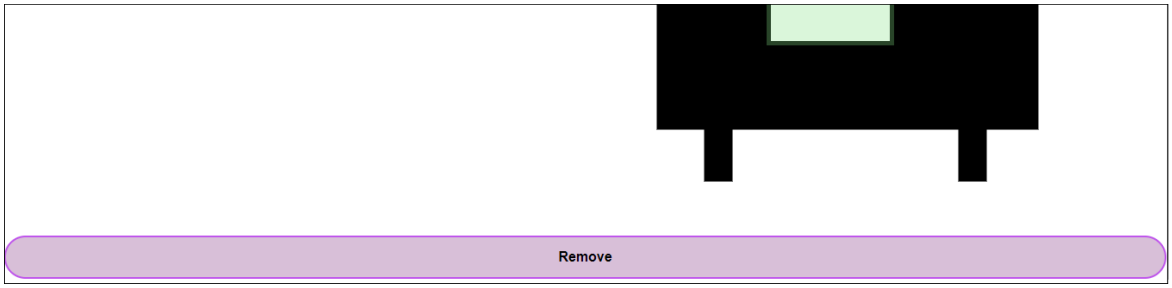


Figure IV.30 : Interface de suppression d'un distributeur 2

IV.3.2.6. Interface de changement des prix

Cette interface s'affiche lorsque le superviseur veut modifier les prix des produits. Il choisi le distributeur qu'il veut changer le prix de son produit à partir de la liste. Il peut voir tous les compartiments avec les produits et ses prix. Il clique sut le compartiment désirer et saisi le nouveau prix.

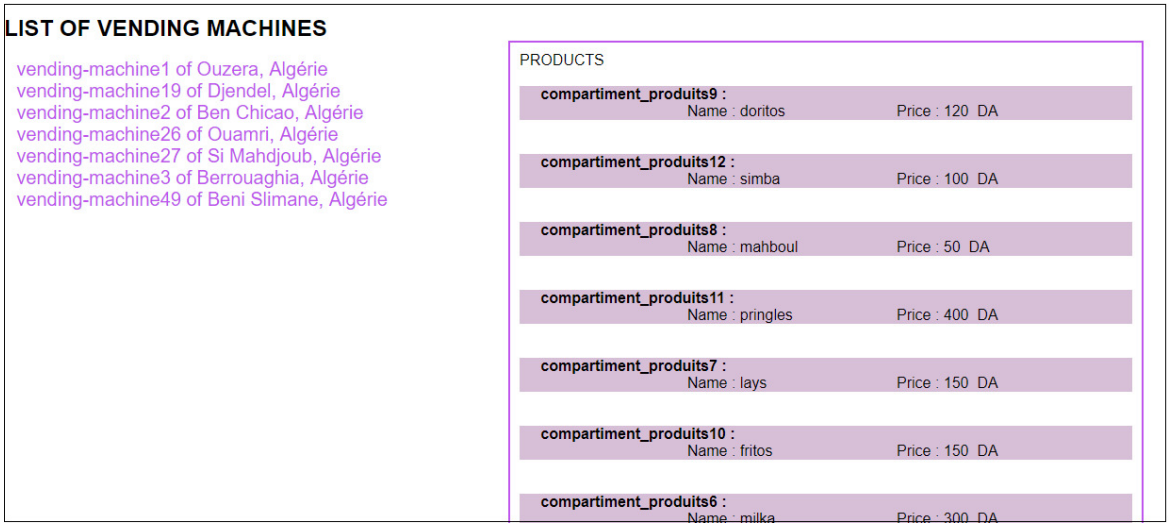


Figure IV.31 : Interface de changement de prix1

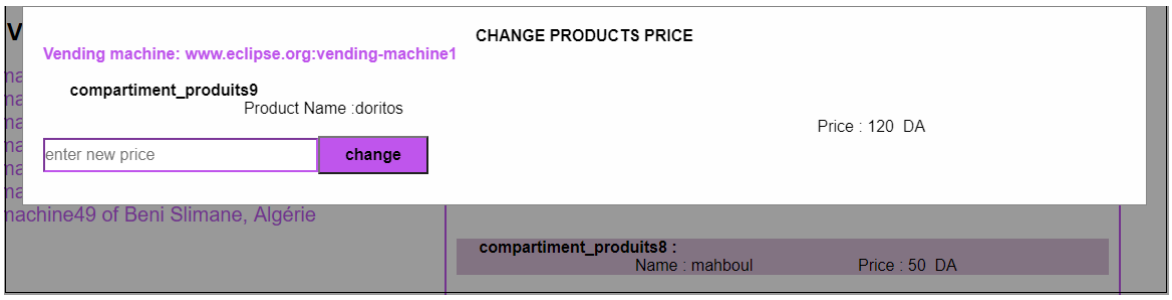


Figure IV.32 : Interface de changement de prix2

IV.4. CONCLUSION

La partie implémentation est l'étape la plus importante dans notre projet. Ce chapitre a décrit trois phases principales. La phase de modélisation et transformation ou nous avons bien expliqué la modélisation de système de distributeurs automatique de produits et les deux transformations pour obtenir le fichier JSON, ainsi les différents outils employés. La phase de simulation ou nous avons présenté la modélisation de système et ses fonctionnalités par l'outil node red. Et la phase réalisation ou nous avons décris les différentes fonctions de l'application web mises en pratique avec quelques interfaces pour bien illustrer le projet.

CONCLUSION GENERALE

CONCLUSION GENERALE

Le jumeau numérique est la réplique virtuelle d'un objet physique. Il est capable de reproduire ses fonctionnalités et de simuler ses performances dans le monde réel. Ce double numérique est continuellement nourri de données récoltées sur l'objet physique afin qu'il évolue de la même manière. Il peut être doté d'algorithmes d'intelligence artificielle lui permettant de prendre des décisions comme un être humain. Il a de nombreux avantages, il facilite l'accès aux données, permet la possibilité de collaborer à distance et à plusieurs directement autour de l'objet pour résoudre des problématiques, et permet aussi de prédire les pannes et intervenir avant les incidents. Mais malgré ses multiples avantages, Aujourd'hui, les données récoltées appartiennent aux propriétaires des équipements connectés. La plupart laissent les installateurs du jumeau numérique accéder aux données puisque cela leur apporte un bénéfice direct (analyse, prédictions et conseils).

L'étude réalisée dans ce présent mémoire qui consistait en la réalisation d'une application web permet de gérer le processus de développement des jumeaux numériques. l'objectif majeur est de faciliter le développement des jumeaux numériques des distributeurs automatiques de produits en se basant sur l'ingénierie systèmes basée sur les modèle et en utilisant le langage SysML.

Nous avons passé par plusieurs étapes dans notre travail, tous d'abord, nous avons modélisé notre système des distributeurs automatiques de produits via le diagramme de bloc. Après nous avons transformé le diagramme de bloc vers le modèle du jumeau numérique tout en suivant une démarche d'ingénierie basée sur les modèles en utilisant les deux langages de transformation ATL et ACCELEO.

Nous avons développé une application web qui lance le processus de transformation et qui permet de gérer le processus de développement des jumeaux numériques en utilisant la plate forme Eclipst Ditto. Ainsi, la gestion de localisation et la gestion des policy. Après la terminaison de développement de l'application, nous avons la simulé via l'outil node red.

Le travail accompli au cours de ce projet nous a permis de mieux comprendre le milieu de la notion de jumeau numérique et nous a aidé à avoir une vision réelle de la mise en place d'une application web pour les entreprises.

En effet cette expérience nous a été bénéfique et très utile, car elle nous a permis de savoir une nouvelles notions d'une part, et d'enrichir nos connaissances avec les outils de programmations d'autre part. Nous espérons enfin que ce travail sera bénéfique.

Comme perspective, dans l'ingénierie, il est important d'améliorer des outils d'études avancées et ajouter des applications à l'installation expérimentale pour une meilleure interprétation des résultats. Au niveau technique, il s'agit d'un modèle logiciel dynamique qui s'appuie sur les dernières technologies en collecte de données, en intelligence artificielle**,** en machine learning, en IoT (capteurs), en deep learning et autres analyses de données avancées**.**

REFERENCES BIBLIOGRAPHIQUES

[1] HUB DAY "Digital engineering" – jumeau numérique : impact sur l'industrie, la santé et la société, 28 NOVEMBRE 2019 – 28, ORGANISATEUR : Systematic PARIS-Regio.

[2] Sylvain, C, Définition Jumeau Numérique, Alliance Industrie du Futur – Janvier 2019

[3] : Patel, K. K., & Patel, S. M. (2016). Internet of things-IOT: definition, characteristics, architecture, enabling technologies, application & future challenges. International journal of engineering science and computing, 6(5).

[4] Mechalikh, C. E., Bouafia, R. (2017). IoT Based System for Greenhouses Remote Monitoring and Climate Control.

[5] Russell S. Peak , Roger M. Burkhart , Sanford A. Friedenthal, (2007). Simulation-Based Design Using SysML Part 1: A Parametrics Primer , INCOSE Intl. Symposium, San Diego, Web Release

[6] Node Red essential: (2010) Low code programming for event driven applications, website: https://nodered.org/

[7] Broker MQTT. (2019, mai 10). Consulté à l'adresse Eclipse Mosquitto website: https://mosquitto.org/

[8] https://flows.nodered.org/flow/de9ad7f94c78343416ccb1ec0d4a5b2b

[9] http://www.rgot.org/mqtt-utilisation-de-node-red/

[10] https://www.commentcamarche.net/download/telecharger-34055051-ditto

[11] https://www.eclipse.org/ditto/

[12] Westerman, G. (2015) Internet of things business implementation and opportunities,https://www.journaldunet.com/ebusiness/internet-mobile/ 1490767 -pourquoi-le-jumeau-numerique-est-devenu-tendance/

[13] C. Philipe (2010). Pourquoi le jumeau numérique est devenu "tendance" ? Web & Tech

[14] https://www.etudier.com/dissertations/Distribution-automatique/231884. html

[15]Caron, C. (2005). Introduction aux distributeurs automatiques de boissons, Doc Player.

[16] https://www.definitions-marketing.com/definition/distributeur-automatique/

[17] Centeno, M. (2019). Transparence: les documents de l'Eurogroupe , Conseil européen, Conseil de l'Union européenne.

[18] https://www.companeo.com/distribution-automatique-o-c-s/guide

[19] https://www.companeo.com/distribution-automatique-o-c-s/guide

[20] Johnson, T.A., Paredis, C. J. J., Burkhart, R. M. (2008). Integrating models and simulations of continuous dynamics into SysML. 6th International Modelica Conference, Modelica Association, Bielefeld, Germany, pp. 135-145.

[21] Roques, P. (2011). Étapes de la démarche de conception, Bulletin officiel spécial

[22] http://www.omgsysml.org/what-is-sysml.htm

[23]http://www.siloged.fr/cours/STI2D_sysml/index.html?Diagrammedebloc._html

[24] Mallouli, M. (2014). Méta-modélisation du Comportement d'un Modèle de Processus : Une Démarche de Construction d'un Moteur d'Exécution. Modélisation et simulation. Université Panthéon-Sorbonne -Paris I.

[25] http://ecariou.perso.univ-pau.fr/cours/idm/cours-transfo.pdf

[26] http://tadic.eb2a.com/chapitre%203.pdf?i=1

[27] https://en.wikipedia.org/wiki/Model-driven_engineering

[28] https://www.lembarque.com/modelisation-loutil-open-source-papyrus-prend-en-charge-le-profil-uml-rt-pour-les-systemes-temps-reel_006585

[29] https://www.eclipse.org/ditto/intro-overview.html

[30] https://www.w3schools.com/