# Modélisation des flux logistiques : vers une plateforme d'interopérabilité des objets logistiques

David R. Gnimpieba Zanfack

# Thèse de doctorat

Spécialité Informatique

Présentée à l'Ecole Doctorale en Sciences Technologie et Santé (ED 585)

par David R. GNIMPIEBA ZANFACK

Pour obtenir le grade de

Docteur de L'Université de Picardie Jules Verne

# Modélisation des flux logistiques

## *Vers une plateforme d'interopérabilité des objets logistiques*

Soutenue le 15 Mai 2017 devant le jury composé de :

| | | |
|---|---|---|
| Rapporteurs : | | |
| | Pascal LORENZ | Professeur à l'Université de Haute Alsace |
| | Lyes BENYOUCEF | Professeur à l'Université d'Aix-Marseille |
| Examinateurs : | | |
| | Imed KACEM | Professeur à l'Université de Lorraine |
| | Damien TRENTESAUX | Professeur à l'Université de Valenciennes et du Hainault-Cambresis |
| | Jaafar GABER | Maître de Conférences à l'Université de Technologiede Belfort-Montbéliard |
| Directeurs : | | |
| | Jérôme FORTIN | Professeur à l'Université de Picardie Jules Verne |
| | Ahmed NAIT-SIDI-MOH | Maître de Conférences HDR à l'Université de Picardie Jules Verne |
| | David DURAND | Maître de Conférences à l'Université de Picardie Jules Verne |

*Thèse préparée au sein des laboratoires LTI EA 3899 et MIS EA 4290*

# Remerciements

J’adresse mes sincères remerciements à mes trois directeurs de thèse qui ont accepté de diriger ces travaux, pour leurs conseils, leur disponibilité et les efforts continus qu’ils ont fournis pour la réussite de ces travaux de recherche. Monsieur Jérôme FORTIN, Professeur à l’Université de Picardie Jules Verne, Directeur de l’ESIEE-Amiens, Monsieur Ahmed NAIT-SIDI-MOH, MCF-HDR à l’Université de Picardie Jules Verne, Monsieur David DURAND, Maître de conférences à l’Université de Picardie Jules Verne.

Nous tenons également à remercier la Région Picardie pour avoir soutenu le projet COM-SLoT dans le cadre duquel ces recherches se sont effectuées.

Nous remercions humblement Monsieur Pascal LORENZ, Professeur à l’Université de Haute Alsace, et Monsieur Lyes BENYOUCEF, Professeur à l’Université d’Aix-Marseille pour avoir accepté de rapporter cette thèse. Nos remerciements à Monsieur Imed KACEM, Professeur à l’Université de Lorraine, Monsieur Damien TRENTESAUX, Professeur à l’Université de Valencienne Le Mont Houy, Monsieur Jaafar GABER, Maître de Conférences à l’Université de Technologie de Belfort-Montbéliard pour avoir accordé leur disponibilité afin d’examiner ces travaux de recherche.

J’adresse en outre mes remerciements à mes frères et sœurs et toute ma famille pour le soutien et les encouragements qu’ils m’ont donnés tout au long de cette épreuve et particulièrement ma maman Christine NGUETSE.

Mes remerciements les sincères à tous mes collègues de l’INSSET pour leurs encouragements et avec lesquels j’ai passé des moments conviviaux et agréables, nos échanges m’ont donné des idées et du courage pour accomplir ces travaux.

Je remercie enfin ma compagne Sonia pour m’avoir encouragé afin que je finalise ces travaux.

# Table des matières

# Introduction générale

## 1.1 Contexte scientifique et technologique

La "logistique globale" peut être définie d'après Courty[1] comme l'ensemble des activités internes ou externes à l'entreprise qui apportent de la valeur ajoutée aux produits et aux services au profit des clients. La chaîne logistique quant à elle est, selon Stadlter et Kilger[2], un ensemble de plusieurs organisations indépendantes, liées par des flux physiques, informationnels et financiers. Ces organisations peuvent être des entreprises produisant des composants, des produits intermédiaires ou des produits finis, des prestataires de services logistiques et même le client final lui-même. Lummus et Vokurba[3] complète cette définition en mettant un accent sur les activités et le système d'information. Selon ces auteurs, la chaîne logistique est l'ensemble des activités exercées sur un produit de la matière première jusqu'à la livraison au client final en incluant l'approvisionnement en matière et produits semi-finis, la fabrication, la distribution sur tous les canaux, la livraison au client. La chaîne logistique intègre également le système d'information permettant le suivi de toutes ces activités.

Il ressort de ces définitions que la chaîne logistique se focalise autour du concept de ***produit*** dans ses trois états, brut (matières premières), semi-fini ou fini. Plusieurs ***acteurs*** participent à l'élaboration et l'acheminement de ce produit vers sa destination finale en y apportant leur savoir-faire ou en sous-traitant leurs ***services*** divers et variés (approvisionnement, transformation, production, distribution, vente, etc.), qui forment les principales ***activités*** usuelles d'une chaîne logistique. Pour atteindre leurs objectifs stratégiques, ces acteurs sont organisés en ***réseaux*** logistiques dans lesquels s'échangent particulièrement trois types de ***flux***: les ***flux physiques***, les ***flux d'informations*** et le ***flux financier***. Le flux physique, naît du déplacement des produits à l'intérieure d'une même entreprise (flux interne), ou entre des acteurs géographiquement distribués (flux externe). Ces déplacements sont justifiés par l'ensemble des diverses transformations et des services à réaliser sur la matières première pour l'emmener au stade de produits finis et vers le client final. L'écoulement de cette matière (produit) et les différents changements d'état qu'il subit créer donc un flux physique. Afin de coordonner leurs activités sur le flux physique ainsi créer, les différents acteurs ont besoin de s'échanger des flux d'informations sous diverses formes: *mails*, *contrats*, *documents*, *téléphone*, *plannings*, etc. Le flux financier quant à lui résulte des opérations d'achat/ventes de moyens de productions, d'équipements, de biens et services, des locations d'entrepôts ou de plateforme de stockage ou tout simplement des salaires des employés.[1]

Pour optimiser les flux sus-cités, améliorer la performance, optimiser les ressources et les gains d'une organisation, la chaîne logistique doit être ***gérée*** par un acteur ou l'ensemble

---

1 Courty, 2003 : P. Courty. *Les enjeux industriels et les nouvelles problématiques scientifiques - De la logistique à la logistique globale*. Ecole d'été d'automatique. 2003, France.

2 Stadtler et Kilger, 2000 : H. Statler et C. Kilger. *Supply Chain Management and Advanced Planning : concepts, models, software and case studies*, Editions Springer Verlag, 2000.

3 Lummus et Vokurka, 2004 : R.R. Lummus, R.J. Vokurka. *Defining supply chain management: a historical perspective and practical guidelines*. Industrial Management & Data Systems, 1999.

des acteurs qui y participent. Dans ce sens certains auteurs définissent la Supply Chain Management (SCM) comme étant la gestion des approvisionnements et des marchandises depuis les fournisseurs jusqu'aux clients finaux, l'utilisation des processus et des technologies pour améliorer la compétitivité en rassemblant l'ensemble des partenaires [2]dans un but commun d'optimisation et d'efficience. Par ailleurs dans la littérature, la *SCM* c'est aussi le positionnement des produits en bonne quantité, au bon endroit, et au bon moment où le besoin se fait sentir. Ou encore allouer les ressources de production, de distribution, de transport et d'information pour atteindre le niveau de service exigé par les clients, avec un bon rapport qualité /prix.

Les activités de gestion requises par la SCM exigent d'avoir un système décisionnel qui s'appuie naturellement sur un système d'information pour piloter le flux d'informations générés par le flux physique. Pour faciliter cette prise de décision, le système d'information permet de récolter, de stocker, de transférer, de traiter, d'analyser et d'afficher les données issues des différents acteurs de la chaîne sous diverses formes (fichiers, mails, sms, etc), ou production de tableaux de bords contenant des indicateurs de performances *KPIs (Key Performance Indicator)*.

Dans un soucis d'amélioration des systèmes de gestion du flux d'information dans la chaîne logistique et d'un point de vue métier, la plupart des auteurs s'accordent sur le constat que la logistique a atteint un stade où il est plus que nécessaire de franchir un nouveau cap pour être en mesure d'adresser les nouveaux défis que ce métier doit affronter pendant les décennies à venir. Ce nouveau cap est imposé par des évolutions et transformations technologiques (EDI, Cloud computing, Internet des objets, Objets connectés, ...), économiques et sociétales marquantes dans presque tous les domaines et les secteurs clés de la logistique (gestion des stocks, transport, distribution, gestion des commandes, outsourcing, co-packaging, co-manufacturing, custom packaging, lot-sizing, service 3PL/4PL, ...). Ces évolutions technologiques et ces changements du contexte économique des affaires ont impacté autant les pratiques logistiques que les secteurs d'activités auxquels elles s'appliquent: pharmaceutique, hospitalière, logistique de production, transport de matières premières, transport de marchandises, transport de fret, logistique automobile, logistiques des biens et services, gestion des retours client, recyclage et gestion des déchets, logistique des produits dangereux, logistique militaire, etc.
Parmi cette multitude de besoins et de nécessités, nous soulignons de manière non exhaustive quelques uns.

- **La réorganisation des réseaux pour améliorer la collaboration entre les acteurs de la supply chain**

L'un des enjeux majeur de la logistique est la réorganisation des réseaux, que ce soit au niveau des PME ou des grandes entreprises. Il s'agit ici d'étendre le réseau existant, le connecter à d'autres réseaux et le rendre plus flexible afin de couvrir un périmètre d'action plus large. Les données sur les flux logistiques sont disponibles dans les bases de données et les systèmes de gestion propriétaire des entreprises. Ce mode de gestion dit *en silo* empêche

les partenaires travaillant sur la même chaîne ou le même flux d'avoir une visibilité globale sur l'ensemble du flux et anticiper les événements futurs. La nouvelle *Supply Chain* exige d'abord d'extraire ces données collectées et stockées dans les différents systèmes *(ERP, WMS, EDI, ...),* les croiser pour extraire l'information pertinente et de partager ces données à l'ensemble des partenaires agissant sur le même flux ou la même chaîne de valeurs afin de mieux piloter et coordonner les activités. L'accès aux données fiables en temps réel facilitera aussi la collaboration entre les différents acteurs logistiques impliqués dans le flux, et améliorera considérablement leur prise de décision tant au niveau stratégique, tactique que opérationnelle.

- **Le recours à de nouvelles architectures logicielles et de nouvelles technologies**

Les outils et technologies actuels de collecte, de stockage et d'analyse des données dédiés à la chaîne logistique sont pour la plupart basés sur le modèle des ERP *(Enterprise Resource Planning).* Au niveau architectural, les systèmes d'information d'entreprise sont pour la plupart constitués des portails web collaboratifs, sur des architectures orientées services, des architectures orientées composants ou des middlewares collaboratifs plus ou moins orientés événements (architectures *EDA*). Mis à part les architectures orientées services (offre *SaaS*) et les middleware orientés Événement, le point commun des architectures système existantes est de permettre une communication « *point à point »* ou d'avoir un réseau structuré sous forme de mèches entre les différentes composantes de l'entreprise ou de la firme. Il en est de même que les acteurs qui collaborent pour le pilotage du même flux dans une même région géographique ou un périmètre plus étendu (pays, continent).

Le contexte actuel nécessite de nouvelles architectures logicielles qui facilitent la collaboration entre partenaires malgré l'hétérogénéité des formats de stockage de leurs données. Ces nouvelles architectures devraient prendre en compte aussi la confidentialité et la sécurité des données collectées et partagées. L'intégration de nouvelles technologies comme l'Internet des Objets et les architectures Cloud promet d'apporter un réel changement face aux nouveaux défis et risques de cette nouvelle Supply Chain. Mais les entreprises peinent à adopter ces nouvelles technologies et s'investir dans de nouvelles infrastructures IT à cause d'une absence de visibilité sur le marché et les tendances technologiques à long terme. Ajouter à cela un manque de compétences adaptées à la réalisation et la mise en oeuvre de ces nouvelles technologies et leur utilisation.

- **La traçabilité des objets et entités logistiques**

La traçabilité des objets logistiques naît du besoin d'identifier tous les objets dans un même flux de marchandises, de savoir pour chaque objet son origine et sa destination, et aussi les différentes étapes de son parcours dans toute la chaîne, du fabricant au consommateur final. Les codes à barres ont longtemps été utilisés pour le traçage de la marchandise dans la chaîne logistique. Ces codes à barres sont des codes 1D, des codes 2D. L'avènement de la RFID a permis de mettre en place des tags RFID qui permettent le stockage de plus d'informations sur des puces électroniques.

Malgrè ces avancées, le besoin d'identification et de traçabilité des objets logistiques reste un défi. En effet, il y'a de plus en plus d'objets à identifier, et avec l'avènement des objets connectés, il y'a de plus en plus de données récoltées sur ces objets. Ces données posent un problème de stockage à cause du grand volume de données générées, des acteurs qui utilisent ces données sont différents par leur format d'échange et de stockage, par leurs protocoles de communication. À ce niveau, il faut donc de nouvelles plateformes de stockage, de nouvelles technologies de traçabilité, de nouveaux protocoles de communication pour palier à l'identification, la traçabilité et le suivi des objets logistiques dans différents domaines industriels.

- **La recherche d'une efficacité au niveau stratégique et opérationnel pour une meilleure optimisation des flux**

Le niveau stratégique est défini comme étant l'ensemble des décisions prises par l'entreprise pour le long terme (au delà de 6 mois). Ces décisions concernent l'ouverture ou la fermeture de site, la délocalisation, le choix des fournisseurs et des sous-traitants, de nouveaux partenaires, développement d'un nouveau produit, ainsi que les objectifs financiers. Au niveau tactique, il est question de définir les moyens en terme d'équipes, d'infrastructures, de quantités de matériels. Le niveau tactique assure aussi la planification des activités sur lesdites ressources afin de réaliser la stratégie défini au niveau supérieur. Le niveau opérationnel concerne l'exécution des tâches quotidiennes de l'entreprise, au niveau d'un poste de travail par exemple. A ce stade, le planning des tâches à faire est plus détaillé et plus précis.

- **La gestion des risques**

Dans un climat mondial ou les crises et catastrophes naturelles ne cessent de s'enchaîner, l'une des préoccupations majeure des entreprises est d'avoir des outils de prévention et d'analyse des risques afin de déterminer le moment où surviendra la crise, d'anticiper si possible et d'éviter des pertes énormes ou des catastrophes. Ceci est vrai tant au niveau macro-économique que micro-économique, que ce soit dans la PME ou la grande firme. En effet, les prix des matières premières et des produits semi-finis deviennent de plus en plus volatiles et sujettent à des changements que ce soit à cause des facteurs climatiques (pour ce qui est des récoltes) que ce soit des marchés financiers et les bourses de matières premières. Le partage d'informations en temps réel et une analyse intelligente de ces informations réduirait certaines catastrophes à l'échelle locale (PME, Région, ...) ou à l'échelle globale (grande firme, Pays, ...). Concrètement le problème ici est de faire la liaison entre les facteurs climatiques, les facteurs économiques et les facteurs humains pour faire de la prévision à court ou à long terme afin de réduire le risque.

## 1.2 Problématique

Ces travaux de recherche ont été réalisés dans le cadre du projet régional *Com-SLoT (Communauté de Services Logistiques sur l'Internet des Objets)* qui s'inscrit dans l'axe des *"outils numériques pour la performance industrielle" (2013-2016)*. Com-SLoT vise de

mettre en place une communauté de services pour la gestion des objets logistiques. Il s'agit précisément de récolter et de partager toutes les informations nécessaires à la planification, l'exécution et la coordination des flux logistiques.

Le problème abordé dans cette thèse est la "***Modélisation des flux logistiques, développement et implémentation d'une plateforme d'interopérabilité des objets logistiques***".

Le modèle de flux de marchandises que nous cherchons à modéliser a donc pour objectif et défi principal de faciliter le développement et la mise en place des systèmes de gestion collaboratives de flux de marchandises basés sur l'Internet des objets (*IoT : Internet of Things*) et les plateformes Cloud Computing. Ce défi scientifique et technologique peut se décliner en sous objectifs suivants:

- Proposer un modèle générique de flux logistique centré sur l'objet logistique en tant qu'entité autonome et communicante. Les nouveaux modèles de flux logistiques devraient considérer l'objet logistique comme une entité à part entière, qui interagit avec son environnement par des envoies et réceptions des événements, des messages ou des signaux (palette, container, lot de marchandises, entrepôt, moyen de transport, ...). Dans ce contexte, l'objet logistique ne peut pas être considéré comme une simple ligne de commande comme c'est le cas dans la plupart des systèmes de gestion et de pilotage de flux.

- Passer de l'échange au partage: collecter les données sur l'entité et le flux logistique, stocker et partager ces données entre l'ensemble des acteurs de la *Supply Chain* à des fins de pilotage et de coordination des activités liées au flux, de la planification stratégique à l'exécution opérationnelle des tâches. Ce partage de données facilite aussi la visibilité et la collaboration entre les différents acteurs qui interviennent sur le flux.

- Gérer l'hétérogénéité des données et les droits d'accès pour l'interopérabilité des flux: les données récoltés sur le flux de marchandises provient d'acteurs et systèmes de gestion qui diffèrent les uns des autres, que ce soit au niveau du format d'échange, au niveau des protocoles, qu'au niveau des modèles d'entités et de processus. Trouver un modèle de flux, d'architecture et de protocoles qui résolvent le problème d'hétérogénéité contribue considérablement à l'interopérabilité des systèmes, des flux et des acteurs logistiques.

- Intégration avec l'IoT et le Cloud Computing: intégrer le flux de marchandise avec les technologies de l'internet des objets et des plateformes Cloud. Les enjeux sont multiples par rapport à ce point, car la valeur ajoutée d'une Supply Chain moderne vient de sa capacité à récolter des données issues des entités logistiques en temps réel, à diffuser ces informations à travaers des plateformes Cloud, mais surtout de sa capacité d'y extraire des informations fiables et pertinentes, de les analyser et de les combiner pour générer des indicateurs afin de faciliter la prise de décision à tous les niveaux de la chaîne. Il est aussi question de coupler les technologies *RFID* et les middleware EPCIS pour

l'identification, la traçabilité des marchandises afin d'améliorer la démarche *PLM (Product Lifecycle Management)*.

- Réaliser une vrai plateforme collaborative en mode SaaS: dans ce cadre, il est plus question d'architecture et d'intégration. En effet, les architectures SaaS permettent d'avoir des systèmes de gestion collaborative consommables à la demande, et selon les besoins de l'entreprise. Elles sont basées sur les architectures *SOA (Service Oriented Architecture)* et cela permet d'avoir un système découpé en modules autonomes et interopérables. Ceci facilite l'intégration avec d'autres architectures et composants logiciels (middleware EPCIS par exemple), ainsi que d'autres systèmes existants *(portails web collaboratifs, ERP, WMS, etc)*.

## 1.3 Organisation du mémoire

Ce mémoire est organisé en cinq chapitres, une introduction générale et une conclusion générale. Dans ce qui suit, nous décrivons le contenu de chacun de ces chapitres.

**Introduction générale.** L'introduction générale définit la logistique globale et les flux logistiques. Cette section présente le contexte dans lequel ces travaux de recherches ont été mémés, et situe notre problématique dans ce contexte. Nous présentons par ailleurs les différents aspects clés de cette problématique que nous traitons dans les chapitres suivants.

**Chapitre 1: Etat de l'art**. Ce chapitre présente un état de l'art sur les modèles et outils formels de modélisation et de gestion des flux et des objets logistiques existants dans la littérature scientifique. Un panorama des projets de recherche et développement (R&D) en cours ou achevés dans le domaine de la gestion des flux logistiques et de l'Internet des Objets (IoT : *Internet of Things*) y est aussi abordé. La première partie présente une revue de littérature sur les contributions scientifiques, des projets de recherche et développement sur les modèles et plateformes collaborative en lien avec la problématique traitée. La deuxième partie traite de l'IoT et son apport pour la gestion des flux logistiques en soulignant les projets et plateformes de l'Internet des objets architecturés pour la collaboration et le partage d'informations entre les acteurs de la supply chain et les enjeux y afférents. La dernière partie de ce chapitre fait une synthèse des méthodes et outils développés dans la littérature pour la modélisation et la simulation de flux logistiques.

**Chapitre 2: Modèle statique du flux logistique**. Dans ce chapitre nous appliquons la méthode de l'ingénierie dirigée par les modèles (IDM) pour la modélisation de la statique du flux logistique. Concrètement nous proposons une extension de la spécification UML par un profil UML du niveau M2 des méta-modèles MOF. Cette extension permet de représenter le flux logistique, les entités qui le constituent ainsi que l'environnement dans lequel il évolue. Ceci en tenant compte des spécificités du cadre de nos travaux, à savoir l'intégration du flux logistique dans le Cloud à travers les technologies de l'IoT pour le partage d'informations et l'interopérabilité des flux et des acteurs logistiques. Nous complétons cette démarche par un modèle de graphe représentant le réseau de flux logistique à l'instar des réseaux sociaux.

Ceci permet de mieux appréhender les interconnexions entre les entités d'un même flux et les relations d'interdépendances entre les différents flux de marchandises, ainsi que les relations entre le flux et les événements qui sont générés par son environnement et qui sont sujets du changement de contexte auquel dépend fortement son cycle de vie.

**Chapitre 3: Modèle dynamique du flux logistique.** Dans ce chapitre, nous proposons un modèle dynamique du flux basé sur la norme ISO LOTOS, afin de spécifier le workflow. Cette modélisation dynamique du flux nous permet d'analyser ses propriétés quantitatives et qualitatives. Elle permet également de spécifier la plateforme de collaboration en tant que systèmes communicants avec des workflows et des acteurs logistiques.

**Chapitre 4: Extraction des données et modélisation des connaissances dans un réseau de flux logistique**. Dans ce chapitre, nous proposons une ontologie web pour la représentation et la classification des concepts dans le domaine du flux logistique, l'extraction de connaissance dans le réseau de flux avec des requêtes pour la gestion et la coordination des entités logistiques. Par ailleurs, cette ontologie pourra être utilisée comme format d'échange et de partage d'informations entre les différents acteurs logistiques, ce qui pourra faciliter l'interopérabilité des systèmes d'information et la gestion de flux de marchandises. Nous utilisons trois standards: le langage de définition de processus *XPDL (XML Process Definition Language)*, le *Business Process Modeling Language (BPML)*. De même, nous utilisons le standard *OWL (Web Ontology Language)* pour la spécification
L'introduction de la notion de contexte dans la gestion du flux pour mieux appréhender l'ubiquité des objets logistiques et leur changement de statut est abordée dans ce chapitre. Nous introduisons la notion d'interface de communication (gateway) entre le flux et son environnement, en nous basant sur le standard *Next Generation Service Interface (NGSI)* et l'interface *Publish/Subscribe* pour la souscription au contexte, la publication et le partage d'événements entre le les entités et les différents acteurs.

**Chapitre 5 : Plateforme collaborative basée sur l'Internet des Objets.** Dans ce chapitre nous développons l'architecture de la plateforme collaborative que nous proposons pour le suivi et le tracking des objets logistiques, le traitement et le partage des données récoltées, la gestion des droits d'accès, la gestion de l'interaction entre tous les acteurs impliqués dans le flux. La valeur ajoutée de cette architecture proposée est principalement l'intégration des différentes couches de l'IoT telles que la couche capteurs, la couche de transmission de données, la couche de stockage dans le Cloud et enfin la publication des données et événements recueillies aux utilisateurs. Cette architecture a pour objectifs de faciliter le partage d'informations sur les flux logistiques pour la traçabilité, la collaboration entre les différents acteurs de la chaîne d'approvisionnement. La plateforme est implémentée et les résultats de simulation obtenus sont rapportées et analysés dans ce chapitre.

**Conclusion générale.** Dans cette section nous présentons les principaux résultats obtenus et contributions proposées dans cette thèse. Nous y présentons également quelques perspectives et pistes de recherche de ce travail.

# Chapitre 1: Gestion des flux logistiques : état de l'art

## 1.1. Introduction

La gestion du flux logistique a toujours été au centre des organisations, des entreprises et des firmes, quelque soit leur taille et quelques soit leur domaine d'activité. Parmi ces flux, la logistique hospitalière avec les flux de médicaments et de produits pharmaceutiques, les flux de matières premières dans l'industrie de production, de produits semi-finis ou finis dans des chaînes de fabrication et de transformation de l'industrie automobile, textile, alimentaires ou cosmétiques. S'ajoute à cette liste le flux de matières généré par l'industrie tertiaire, les transports de marchandises de toute sorte et des flux des biens et services variés.

Dans ce chapitre, nous présentons un état de l'art sur les modèles et outils formels de modélisation et de gestion des flux et des objets logistiques existants dans la littérature. Un panorama de projets de recherche et développement (R&D) en cours ou achevés dans le domaine de la gestion des flux logistiques et de l'internet des objets y est abordé. La première partie présente une revue de littérature sur les contributions scientifiques, des projets de recherche et développement sur les modèles et plateformes collaborative en lien avec la problématique traitée. La deuxième partie traite de l'internet des objets, son apport vis-à-vis de la gestion des flux logistiques en soulignant les projets et plateformes de l'Internet des objets architecturés pour la collaboration et le partage d'information entre les acteurs de la supply chain et les enjeux y afférents. La dernière partie de ce chapitre fait une synthèse des méthodes et outils retrouvés dans la littérature pour la modélisation et la simulation des flux logistiques.

## 1.2. Revue de littérature et panorama des solutions développées

### 1.2.1. Contributions à la modélisation des flux logistiques

Dans ce paragraphe nous faisons état de quelques contributions scientifiques dans le domaine de la modélisation des flux logistiques. Le sujet est abordé dans chaque contribution suivant un angle bien précis, et l'objectif est fixé soit pour résoudre un problème particulier lié à la maîtrise de flux soit pour s'attaquer à la chaîne logistique dans sa globalité.

Dans [1] les auteurs proposent une modélisation de la chaîne logistique en utilisant les réseaux de Petri (RdP) continus. Ces travaux de recherche étendent le **modèle de RdP continu à vitesse variable** (RdPCV) du trafic routier à l'étude du comportement dynamique du flux physique au sein d'une chaîne logistique. Cette étude est fondée sur une similarité entre le transfert du flux de produits dans une chaîne logistique et celui du transfert de flux de véhicules sur un réseau routier. Dans cette étude, les auteurs ont utilisé les RdP continus à vitesse variable pour modéliser les différents acteurs de la chaîne logistique. Il est aussi question de définir de nouveaux éléments permettant d'étendre le modèle obtenu afin qu'il prenne en compte l'étendu de la chaîne logistique (du fournisseur des fournisseurs jusqu'aux clients finaux). Dans le même ordre d'idée, les auteurs dans [2] proposent une modélisation et une analyse de performances des systèmes logistiques à l'aide de réseaux de Petri stochastiques. Ils font appel à un nouveau modèle de RdP pour modéliser la dynamique des flux logistiques évoluant en quantités discrètes (caractéristique des systèmes à événements discrets). Le travail s'est focalisé sur les systèmes de gestion de stocks. De plus, les auteurs dans [88] proposent une modélisation de composition de services web à l'aide d'un modèle de réseau de Petri orienté Objet (Object-oriented Petri net based algebra). Ce model de réseaux de Petri appelé G-Nets permet de modéliser la composition de services web en prenant en compte l'identification des types de données qui circulent dans le réseau et en associant des conditions aux transitions. D'autres travaux ont traité la même problématique en utilisant, par exemple, les réseaux de Petri colorés pour distinguer les types de paramètres du système. Notons que cette classe des RdP n'est pas basée sur des outils formels, contrairement aux G-nets.

Par ailleurs, il est proposé dans la littérature l'utilisation des heuristiques pour la résolution du problème de localisation et de routage (LRP-Location-Routing-Problem) [3] dans la logistique. En effet, LRP est un problème de logistique du transport qui implique deux niveaux de décision : le niveau stratégique et le niveau tactique. Le niveau stratégique est celui de la localisation de dépôts et le niveau tactique est celui de l'organisation des tournées de véhicules. LRP est utilisé dans plusieurs applications telles que la distribution du courrier, la livraison de colis ou la collecte de déchets. Ces travaux de recherche visent à traiter du problème de localisation-routage avec des capacités limitées des dépôts et pour les véhicules réalisant les tournées. Ce travail de thèse propose aussi une résolution prenant en considération l'intégralité du problème en utilisant une approche heuristique et une approche exacte basée sur des modèles mathématiques. Plus précisément, la première approche proposée est basée sur un algorithme glouton renforcé par une technique

d'apprentissage et par une post-optimisation. La seconde est basée sur les heuristiques avec une approche évolutive via un algorithme de mimétique avec gestion de la population. Une dernière approche, dite coopérative, est basée sur un rapport non hiérarchique entre des phases de localisation et de constitution de tournées, ce qui permet de choisir des dépôts de façon plus simplifiée. L'auteur aborde le problème de planification tactique des activités de production dans [4]. Il analyse le cas d'une large classe de chaines logistiques et compare aussi les performances des différentes architectures de pilotage de la chaîne logistique par expérimentation en prenant en compte les incertitudes sur les données et les aléas de production, les degrés de collaboration entre les décideurs et les conflits sur l'utilisation d'une ressource partagée. Trois familles d'architectures de pilotage sont définies dans ce travail : distribuée, mixte et centralisée. Ces dernières permettent de coordonner le flux de matières premières dans les entreprises en se basant sur le flux d'informations échangées par les centres de décisions impliqués. Ici, il y'a une prise en compte de la robustesse et de la réactivité de la chaîne logistique. La robustesse étant l'aptitude de l'architecture de planification à ne pas être sensible aux incertitudes sur certaines données (capacité, demande, …) et la réactivité étant la rapidité de réaction sur l'ensemble de la chaîne face à l'apparition d'un aléa de fonctionnement ou aux événements imprévisibles de l'environnement. Le modèle proposé dans ce travail est fondé sur les techniques de la programmation dynamique (horizon glissant). Le travail présenté dans [4] mentionne deux méthodes utilisées en générale pour la planification des activités de production : la méthode MRP basée sur le concept de flux poussé et la méthode Kanban basée sur le concept de flux tiré. La **méthode MRP** (Material Requirements Planning) qui est basée sur les demandes ou les prévisions de clients ou les ordres de fabrication. La méthode **MRP** permet de proposer les commandes d'achat en fonction de la capacité de production du système, les délais de production et d'approvisionnement, les niveaux de stocks de matières premières et de produit semi-finis. La **méthode Kanban** est une généralisation du « juste à temps », elle permet de remonter des informations liées à la consommation de produits de l'aval vers l'amont. Basée sur l'idée du flux tiré, elle permet de produire en amont uniquement ce qui est demandé en aval, et qui dépend lui aussi de la demande du poste en aval. L'enjeu de ces travaux est d'améliorer le processus décisionnel dans la chaîne logistique suivant trois objectifs : la caractérisation et l'amélioration des performances de la chaîne logistique en se basant sur la collaboration inter-entreprise et en se focalisant sur des critères quantifiables de la chaîne; l'analyse comparative d'architectures décisionnelles et l'élaboration d'un modèle analytique générique pour la planification.

L'auteur aborde le problème de performance par la modélisation de flux logistiques des patients dans un service d'urgence hospitalier dans [5]. Dans cette thèse de doctorat, on s'intéresse au flux de patients dans un service hospitalier et à la minimisation du temps de parcours du patient dans les services d'urgence de l'hôpital, dans le but d'améliorer la performance globale du service. Dans ce travail, l'auteur propose l'utilisation combinée de la **simulation** (par le logiciel de simulation Witness) et des **réseaux de file d'attente** pour la modélisation et la simulation des flux de patients. Il est aussi question de l'analyse des dysfonctionnements du service d'urgence. Ces travaux ont permis aux responsables du service d'urgence de choisir les actions d'amélioration qui permettront de minimiser la durée du parcours du patient. Dans [12], l'auteur a utilisé un algorithme de la recherche opérationnelle (programmation linéaire en variables mixtes) dans le but d'optimiser les

coûts de la chaîne logistique sur un horizon lointain (plusieurs années), les variables de décision de l'algorithme sont l'ouverture, l'agrandissement des sites ou la fermeture, la gestion des flux physiques. Les heuristiques (relaxation linéaire, relaxation Lagrangienne, etc.) ont été choisies comme approche de modélisation. Une autre contribution au sujet de l'optimisation, de gestion intégrée des flux physiques dans diverses structures de la chaîne logistique est proposée dans [13]. L'auteur utilise les heuristiques et les algorithmes polynomiaux pour ramener le problème de planification intégrée à des cas particuliers du problème de dimensionnement de lots. Le but est de rechercher des plannings optimaux pour une chaîne de production. Un autre travail au sujet de la modélisation de flux et d'optimisation dans un réseau d'entreprises partenaires dans la filière Textile-habillement-Distribution est développé dans [14]. Dans ce travail, les auteurs utilisent un outil de simulation et d'optimisation dynamique pour proposer aux entreprises du textile une meilleure planification des approvisionnements à partir de prévisions des demandes. Il est à noter qu'une démarche préalable de modélisation du processus de gestion des flux a permis de mieux comprendre les politiques de gestion de flux dans le domaine du textile (structure des coûts, politique de gestion,…), afin de mieux simuler leurs comportements et d'optimiser leurs performances.

Dans le travail développé en [6], l'auteur apporte une contribution à la modélisation des systèmes industriels et à l'ordonnancement des tâches d'un atelier à machines parallèles par les méta-heuristiques. Dans cet article, l'auteur dresse un panorama d'architectures de référence pour la modélisation des entreprises : PERA (Purdue Enterprise Reference Architecture), CIMOSA (Computer Integrated Manufacturing Open System Architecture Methodology), GRAI (Integrated Methodology)/GIM (Graphe de Résultats et d'activités Inter-reliées), GERAM (Generalised Enterprise Reference Architecture and Methodology). Un panorama de cadre de modélisation (GIM, ENV 40003, etc.), et de méthodes de modélisation telles que OSSAD, SAF, MECI, PETRA, ACNOS ont été également soulignés dans ce travail. Dans notre cas, l'ensemble de ces méthodes et normes permet d'avoir plusieurs vues sur l'architecture du système à modéliser :

- Du point de vue fonctionnel : les processus de l'entreprise peuvent être modélisés en utilisant des formalismes tels que IDEF0 (SADT), IDEF1-3, CIMOSA (langage de description des processus opérationnels). Le modèle peut être fondé sur une étude de spécification à l'aide du diagramme d'activité UML et du BPMN (Business Process Modeling Notation).

- Du point de vue informationnel: il est possible de modéliser les informations portées par les objets d'entreprises à l'aide de formalismes tels que les diagrammes d'entités-associations (UML, etc), ISO DIS10303 (Langage formel de description).
- Du point de vue ressources : le système physique (moyen de transport, production, opérateurs humains, outils informatiques, etc) peut être modélisé à l'aide des réseaux de Petri, des Grafcet, des Statechart, etc.

- Du point de vue organisationnel : la modélisation de la structure organisationnelle du système en utilisant des organigrammes, des langages formels (MOSA: langage formel de description des organisations, GRAI, etc) est possible.

Dans [7], l'auteur aborde la problématique de l'interaction entre des services web composés. Il propose une approche dirigée par les modèles (MDA) pour la spécification, la vérification formelle et la mise en œuvre des services web composés. L'auteur propose un profil UML2 (appelé UML-S) pour la modélisation de la composition de services à l'aide de diagrammes de classes et d'activité, avec génération automatique de code exécutable via BPEL (Business Process Execution Language). Dans ces travaux, la vérification formelle des modèles générés est faite par le langage de processus LOTOS. Nous soulignant l'importance de ce travail dans la mesure où il nous montre la possibilité de construire une application (code exécutable) à partir d'un méta-modéle UML2 vérifié formellement à l'aide d'outils mathématiques (Algèbre de processus LOTOS). Cette démarche est d'autant intéressante qu'elle s'inscrive dans le cadre de l'ingénierie dirigée par les modèles, une piste que nous explorons pour la mise en œuvre de la plateforme COM-SLoT. Par ailleurs, le travail proposé dans [11] aborde une approche systémique pour la construction d'un méta-modèle UML des flux logistiques. Ce méta-modèle UML est ensuite instancié pour la chaîne d'approvisionnement de PSA. Le but recherché est d'évaluer la performance globale d'une chaîne logistique en proposant des indicateurs de flux pertinents. Pour la partie décisionnelle, une heuristique couplée au simulateur propre à l'entreprise est utilisée dans le but d'optimiser les stocks tout en gardant la qualité de service.

Une étude portant sur le problème du « supply contract », le contrat d'approvisionnement qui permet de mettre en œuvre une collaboration entre les différents acteurs de la chaîne logistique pour une politique d'approvisionnement optimale est proposée dans [9]. Dans ce travail, seuls les aspects technique et informationnel ayant des impacts sur le flux physique sont étudiés. Ce travail propose un modèle analytique pour une entreprise particulière. Ce modèle prend en compte les clauses de contractualisation et permet de simuler l'impact de ces clauses sur la performance de la chaîne globale de l'entreprise.

La modélisation des flux physiques et financiers est abordée dans [10]. Le but étant de générer des modèles de planification tactiques pour optimiser à la fois la fonction de coût et la fonction du profit dégagé par la chaîne logistique. La méthode mathématique utilisée est basée sur des outils de planification tels que le MSLP (Continuous Setup Lot Sizing Problem), le PLSP (Proportional Lot Sizing Problem), le GLSP (General Lot Sizing Problem). Dans ce travail, on distingue plusieurs types de flux.

- Le **flux physique** : il s'agit d'un ensemble d'unités circulant dans l'espace, sur une surface, un plan, une courbe, une droite suivant une loi précise (Tchenev, 1997). Ces flux concernent les flux de matières premières, les flux de production et de transformation, les flux de livraison de produits.
- Le **flux financier** : ce type de flux est la contrepartie monétaire des flux physiques.
- Le **flux informationnel** : il concerne les informations relatives à l'état des activités (activités de production/stockage, activité de gestion, etc), les règles de gestion et de pilotage des activités.

Un modèle de réseau de trafic routier est proposé dans [15]. L'idée étant de décrire le comportement dynamique de la circulation de véhicules dans un réseau de trafic en tenant compte de la spécificité au niveau des intersections. Ensuite d'élargir ce modèle de trafic routier à l'étude du comportement dynamique du flux physique dans la chaîne logistique par transposition des concepts du modèle de trafic routier. Dans le but de simuler le pilotage de la chaîne logistique de la filière de production et de distribution de poulets, l'auteur propose dans [18] un modèle de simulation basé sur la dynamique des systèmes de Forester pour l'amélioration du comportement de la chaîne logistique globale. Le but est de déterminer les instabilités de la chaîne qui provoquent des coûts supplémentaires.

### 1.2.2. Projets de Recherche et Développement (R&D) sur la modélisation des flux

Dans cette section, nous présentons quelques projets de recherche et développement qui ont été développés au sujet de la gestion des flux logistiques.

Le but du projet ***OPTIFLUX*** est de développer de nouveaux outils pour la construction de plans de transports ouvert sur les technologies [89]. Ce projet vise à fournir une plateforme générique pour la simulation et l'optimisation du transport afin d'améliorer la gestion de la chaîne d'approvisionnement des entreprises européennes. Les valeurs ajoutées de cette plateforme sont l'utilisation des algorithmes d'optimisation (tel que l'algorithme de génération de colonnes de fourmis), la création d'un environnement intégré pour la gestion de la logistique, la planification des tâches de transport multi-contraintes (temps, qualité de service, …), mise en place d'une plateforme web composée d'un moteur d'optimisation (algorithmes) et trois types de composants applicatifs : LP-Supply Chain pour la gestion de la chaîne logistique; LP-Hub : Système décisionnel dédié à la conception de réseaux de transport; LP-Transportation Planner pour l'organisation des tâches de transport. Les deux solutions LP-Supply Chain et LP-Transportation Planner ont été développées par la société EURODECISION pour la modélisation et l'optimisation globale de la chaîne logistique. LP-Supply Chain optimise la localisation et le dimensionnement des flux logistiques de l'entreprise. LP-Transportation Planner optimise un plan transport, l'approvisionnement, la distribution, la messagerie avec la prise en compte des contraintes de qualité de service et d'exploitation des sites (plage d'ouverture,…). Les domaines d'application de la LP-Supply Chain et la LP-Transportation Planner sont la localisation et dimensionnement des unités de production, des produits, des entrepôts et des hubs. Ainsi que l'optimisation des flux fournisseurs-usines-entrepôts-clients. Ici il est question du dimensionnement de flottes de camions, avions, wagons, navires, etc; de l'affectation des zones de chalandise aux entrepôts; de la localisation des stocks; de l'optimisation des plans de transport; gestion des flux retours.

***SYSLOG*** [90] est un projet labellisé par le pôle de compétitivité NOVALOG, porté par OSADYL Consulting et développé en partenariat avec EURODECISION. Ce projet a pour objectif de développer un logiciel innovant d'aide à la décision logistique pour les acteurs de la chaîne logistique du domaine des transports maritimes (Agents maritimes,

transitaires, manutentionnaires, transporteurs, …). Cette solution peut être accessible via le Web en mode SaaS : Software as a Service. Ce projet vise à répondre aux besoins des acteurs portuaires dans l'optimisation de leurs ressources, de leurs processus métiers et flux de marchandises afin de rationaliser les coûts et les temps de service. Les flux concernés sont les flux de conteneurs du transport par exemple, la massification des flux et la mutualisation des moyens et des ressources (conteneurs). SYSLOG est une plateforme organisant une bourse de fret multimodal pour le transport de conteneurs. Cette plateforme collaborative partage des informations de plusieurs acteurs, elle est dédiée aux échanges de données utilisateurs pour une grande communauté de transporteurs communicants ensemble. L'enjeu étant de faciliter l'identification du transport le plus adéquat, d'accéder à des informations pour définir les chemins possibles (prise en compte des coûts) et compatibles avec les contraintes de délais, tout en évitant des retours à vide.

La plateforme ***Argos transport*** a été créé dans le cadre du suivi des marchandises transportées dans le réseau routier. Elle se focalise sur la gestion en temps réel du flux d'informations concernant le vol de ces marchandises. Argos Transports est une plateforme unique d'appel muni d'un numéro vert [91]. Le système à été conçu et géré par le GIE Argos, à l'initiative de la Fédération Française des Sociétés d'Assurances (FFSA) pour enregistrer en temps réel les déclarations de vols de marchandises et l'assistance au chauffeur dans ses démarches de déclaration de vol et de dépôt de plainte. Argos gère une base de données pour enregistrer les vols de marchandises déclarées par les transporteurs, les chargeurs, les courtiers, les agents généraux, etc. Cette base de données est consultable par les entreprises, les autorités de police, des gendarmes, des douaniers, etc. dans le but de collaborer pour retrouver et restituer les biens volés.

De même, dans **Software Platforms for Internet of Things and M2M** [37], il s'agit de traiter une problématique de communication entre machines (M2M), liée à l'internet des objets. Ce travail dresse un panorama des technologies qui ont contribuées à l'essor de l'IoT. Son apport scientifique est la mise en place d'une plateforme de communication hétérogène entre objets communicants.

**Les technologies RFID** sont des technologies d'identification automatique [92]. Elles sont utilisées pour identifier automatiquement et électroniquement un objet. Les technologies RFID sont constituées de marqueurs/capteurs, de lecteurs, de logiciels de traitement des informations et sont classées suivant plusieurs paramètres : la fréquence utilisée, la portée, le prix, et la consommation d'énergie. Un exemple de mise en œuvre de cette technologie est l'ensemble des solutions proposées par **Ifm electronic** [93]. Ifm met à disposition des industriels une gamme de produits pour la commande, la mesure et le pilotage de systèmes de contrôle commande. Parmi cette gamme de produits, nous citons ceux liés aux technologies RFID :

- Tête de lecture/écriture en technologie 125 kHz: Il s'agit d'un système RFID à base de 125 kHz pour l'identification de produits et le convoyage de palettes et produits dans la chaîne de production. Dans cette gamme de produits nous avons les têtes de lecture suivantes :
  - ✓ Système DTE 100 avec Profibus DP

- ✓ Système DTE 101 avec Profinet
- ✓ Système DTE 102 avec Ethernet/IP
- ✓ Système DTS 125 avec AS-Interface

- Étiquettes électroniques à technologies 125 kHz: Ifm a développé plusieurs étiquettes électroniques (TAG ID) dont la portée dépend de la tête de lecture/écriture associée à ces tags.
- RFID 13,56 MHz: est un système de transmission rapide des données compatible avec la norme ISO. Cette solution supporte les protocoles Ethernet/IP (DTE102), Profinnet (DTE101) et Profibus DP (DTE100). Elle est en outre adaptée pour le contrôle commande dans les chaînes de production.
- RFID UHF qui est l'offre haute fréquence d'ifm constituée d'un ensemble de kits pour le contrôle de production, la gestion des biens, le contrôle du flux de matériel, suivi livraisons et gestion de la chaîne logistique. Ces tags et antennes RFID sont faits pour fonctionner dans un champ de détection proche avec une portée de lecture faible. Dans ce kit, il existe des antennes pour des applications dans un champ pouvant aller jusqu'à 10 mètre de portée, avec des angles d'ouverture de 30° et 70°.

Dans cette même logique, **EPCglobal** [92] a été crée en 2003 par un accord entre EAN International, AUto ID center et UNIFORM CODE COUNCIL. L'objectif de cet organisme est d'assurer le déploiement des technologies EPC (Electronic Product Code) dans le monde et de promouvoir les standards y afférents et les systèmes d'identification. EPCGlobal a soutenu le développement de plusieurs solutions dont Gen-2 (860-960 MHz) capable de lire 1000 puces RFID par seconde et 100 en environnement hostile. GS1 (Global Language of Business) est une structure de concertation de l'industrie, du commerce et de leurs partenaires. GS1 a mis en place des standards internationaux de communication (codes à barres, RFID et EDI) pour optimiser les processus logistiques et la traçabilité des produits. Les standards GS1 permettent de symboliser les objets logistiques, c'est-à-dire de leur donner un marquage unique à l'aide de code à barres. Les différents codes GS1 sont les suivants :

**Le GTIN** : Global Trade Item Number est un code attribué par l'industriel à un nouveau produit (carton, palette, …). Ce code permet d'identifier une unité commerciale de façon unique. La structure du code GTIN est donnée par GS1 (GTIN, GTIN13 et GTIN14). Les codes GTIN permettent au producteur de déclarer la hiérarchie des produits, d'indiquer le nombre de cartons contenus dans la palette, le nombre d'unités contenues dans le carton, etc.

**Le SSCC :** le Serial Shipping Container Code est un code à 18 chiffres qui permet d'identifier de façon unique et sur le plan international toute unité logistique. Au même point de chargement, deux palettes ont donc le même GTIN (car ayant le même contenu) et des SSCC différents. Le SSCC est symbolisé sur des unités d'expédition.

**Le GS1-128:** ce type de code à barres est constitué de codes alphanumériques et permet de donner des informations complémentaires symbolisées sur une unité logistique.
L'implémentation des codes GS1 permet d'améliorer la performance de la chaîne logistique en termes:

- de fiabilité du suivi physique de la marchandise;

- de diminution des litiges et de rapprochement entre commandes, les avis d'expédition et les factures;
- de gain de productivité et de temps; la traçabilité, la sécurité alimentaire et la gestion des rappels de produits.

**Les middlewares basés sur les normes GS1 de EPCGlobal** complètent cette démarche. Il existe plusieurs middleware basés sur GS1 [81, 82] pour la traçabilité des marchandises. A cette gamme de produits, viennent s'ajouter tout un ensemble de couches middlewares et de services qui facilitent l'intégration des standards EPC dans les applications et systèmes informatiques. Les middlewares qui intègrent les standards EPCglobal sont nombreux parmi lesquelles TIBCO RFID Interchange de l'entreprise TIBCO Software .Inc, WebLogic RFID Edge Server de Oracle, UBIMAX, LIT ALE Manager, SmartEPC, RFID Middleware U-Link, etc. Il en est de même des serveurs EPCIS (*Electronic Product Code Information Service*) certifiés par EPC global: SAP Object Event Repository, Smarttrack, Rubi IS de Samsung, RFID Middleware de Nippon telegraph & telephone Corp (NTT), SmartEPC, Synchrony Track and Trace, SRIS (*Secure RFID Integration System*), etc.

Les travaux de recherche dans [94] visent à étudier le **déploiement de la technologie RFID pour la traçabilité et l'authentification des produits emballés dans le verre**. Il s'agit de concevoir, réaliser et tester en environnement de production des technologies RFID pour la filière des emballages de verre. Les technologies développées doivent répondre à la problématique d'intégration dans les processus industriels des utilisateurs de bouteilles et flacons en verre creux. Les applications de cette technologie peuvent se situer à plusieurs niveaux (bouteille, cartons, palettes, etc.) communicants dans un environnement de production ou de stockage. Ceci permet de répondre aux nouveaux besoins des utilisateurs, vu la limite des techniques d'identification par marquage. Ces travaux se sont effectués à l'IEMN (Institut d'Electronique de Microélectronique et de Nanotechnologie) dans le cadre des réseaux de capteurs et des réseaux sans fil de proximité.

De même, **Le projet Global RFID Interoperability Forum for Standards (GRIFS)** du programme cadre FP7 Information Society Technologies (IST) de la commission européenne a reconnu l'importance de l'interopérabilité inter-entreprises au fur et à mesure que nous avançons vers des environnements de plus en plus intelligents où la plupart des appareils sont connectés à des réseaux ubiquitaires sans fils. GRIFS a été initié et financé par l'Union Européenne dans le but d'améliorer la collaboration et d'optimiser ainsi l'interopérabilité mondiale des normes RFID [99]. GRIFS est coordonné par GS1, l'ETSI et le CEN. Le projet GRIFS va lancer un forum qui va continuer à travailler de manière constructive pour garantir, après la fin du projet, un protocole d'accord entre les organismes de normalisation mondiaux actifs dans la technologie RFID. GRIF distingue cinq catégories de normes RFID : les normes pour les caractéristiques de fonctionnement des objets physiques (lecteurs, étiquettes, capteurs, etc.); les normes d'infrastructure pour définir les communications, la résolution et les structures; les normes d'échange de données; les normes relatives aux aspects techniques de l'attribution par les régulateurs de spectre approprié pour l'utilisation de la RFID; et les normes relatives aux questions de confidentialité et de sécurité concernant l'utilisation de la RFID et de réglementation.

Le Projet **ASPIRE** [95] **(Advanced Sensors and lightweight Programmable middleware for Innovative Rfid Enterprise applications)** vise à changer le paradigme actuel de déploiement de la technologie RFID par l'introduction de middleware RFID libre de droit, offert gratuitement aux utilisateurs et PME. Cette tendance facilitera l'intégration du middleware RFID avec du matériel peu coûteux pour les entreprises et avec des infrastructures réseaux de télécommunication. Le middleware ASPIRE est une solution légère qui facilitera le développement et le déploiement de solutions RFID entièrement automatiques, innovantes, programmables, libre de droits, tout en respectant la confidentialité. Ce nouveau paradigme de middleware sera notamment bénéfique pour les PME européennes qui connaissent de nos jours des coûts de déploiement énormes de la RFID.

De même, le premier objectif du projet **HyDRA** [96] est de mettre en place un middleware basé sur une architecture orientée services dont la couche de communication sous-jacente est transparente. Le middleware inclura un support pour des architectures distribuées et centralisées, ainsi que des architectures orientées modèles (MDA). Hydra est déployé sur des infrastructures réseaux nouvelles et existantes, avec une sécurité fiable, une interopérabilité pour des données et des informations, ainsi que des services web pour des périphériques connectés sur le réseau. Le second objectif de Hydra est de développer un SDK (Software Development Kit) qui sera utilisé par les développeurs pour implémenter des applications suivant une approche MDA.

Pour le middleware **MIDFLEX (Un middleware flexible et réflectif pour systèmes embarqués hétérogènes mobiles à basse consommation connectés à l'Internet)** [97], le constat est qu'un nombre croissant de systèmes électroniques font partie de notre environnement et façonneront de plus en plus notre façon de vivre en offrant des moyens de mesurer et contrôler certains paramètres du monde qui nous entoure. Cette augmentation est couplée à une large diversification des systèmes électroniques que ce soit au niveau des technologies de communication (Bluetooth, Wi-Fi, ZigBee, etc.), de leur taille, de la richesse de leurs interfaces, de leur puissance de calcul, de leur capacité mémoire et de stockage d'énergie et surtout de leurs fonctionnalités. La forte variété de ces systèmes, incluant notamment des « réseaux de capteurs », les rend particulièrement complexes à mettre en œuvre et requiert actuellement une expertise très pointue et dépendante du contexte.

La conception d'une couche logicielle intermédiaire entre l'application utilisateur, le matériel et le système d'exploitation, appelée « middleware », facilite la programmation de ces réseaux de capteurs hétérogènes. Actuellement, les middlewares existants y répondent que de manière limitée. **TRASER** est un projet financé par l'UE dans le cadre du 6ème PCRD (FP6) IST (Information Society Technologies [98]. L'objectif du projet est de promouvoir et soutenir des entreprises sur la toile (entreprise dont le modèle économique est basé sur le net). L'objectif scientifique de TRASER est de développer des actions itératives et de comprendre comment les partenaires du réseau pourraient être incités à participer à des services d'information au niveau du réseau déployé par les PME. Il permet également de fournir les meilleures pratiques commerciales et des solutions technologiques qui facilitent les services au niveau du réseau. L'objectif technologique de TRASER est de comprendre comment intégrer des solutions innovantes de

produits centrées sur des solutions existantes de traitement des transactions, et comment une plateforme de développement d'applications en code source ouvert (open source) peut être utilisée efficacement par les différentes PME pour développer des services au niveau du réseau.

### 1.2.3. Architecture dirigée par les modèles (MDA)

Ce demi siècle a été marqué par la quête frénétique vers des langages de modélisation, de notations et de programmation. Nous constatons que la plupart de ces langages bien que robustes sont destinés à des applications et des domaines bien spécifiques et peinent dans l'ensemble à faire l'unanimité et encore moins l'universalité. C'est dans cette optique que *Object Management Group* (*OMG*) proposa d'unifier les langages de modélisation dans un seul toolkit, une seule syntaxe très riche, une seule sémantique consistante, non redondante, flexible et générique qui aboutit à UML 2.0 [78] vers les années 2000. Ainsi l'UML naît de la volonté des trois principaux standards de modélisation OMT [79], Booch [75] et OOSE [76] d'unifier leurs pratiques de modélisation et de conception logicielle.

Le L'architecture dirrigée par les modèles *(MDA)* [77] est une approche de modélisation proposée par OMG comme démarche de réalisation d'une architecture logicielle. MDA est fondé sur le principe que tout est modèle, et donc qu'il serait intéressant de développer des modèles indépendants de toute plateforme, de produire des méta-modèles d'architectures qui sont indépendant des technologies d'implémentation (J2EE, .Net, C++, Python, ...) afin de mieux assurer l'évolutivité et l'indépendance de modèles vis à vis des technologies en pleine expansion, la réutilisabilité des modèles ou des composants dans de nouvelles architectures, leurs robustesse et leur flexibilité. MDA permet aussi de mieux se concentrer sur la logique métier et la découplée des technologies et plateformes d'implémentation et ainsi mieux assurer la pérennité des systèmes d'information des entreprises en perpétuelle remise en cause, entraînant des coûts d'investissement assez conséquent. Pour atteindre cet objectif, OMG met en place tout un ensemble d'outils, de méthodes et de langages. Parmi les multiples outils d'OMG pour le MDA nous pouvons citer les standards de modélisation tels que le MOF, l'UML, les profils UML, le *Common Warehouse Metamodel* (*CWM*), le *Human Usable Textual Notation* (*HUTN*), les standards de sérialisation et de partage de modèle tels que XML, *XML Metadata Interchange* (*XMI*), les standards de transformation de modèles comme *Query-View-Transform(QVT)*, QVT-O (*QVT-Operational*), *Atlas-Transform-Load (ATL)*, d'expression de contraintes sur le modèles comme *Object-Constraint-Language (OCL)*, et tout un ensemble de profils UML pour des domaines spécifiques, extensibles selon les besoins du concepteur. Il est important de souligner l'apport des plugins Eclipse plugins Eclipse comme *EMF (Eclipse Modeling Framework)* et *GMF (Graphical Modeling Framework)*, les projets tels que Equinox, Syrius, AndroMDA et MagicDraw qui sont des environnements de travail qui facilitent la réalisation de modèles, l'écriture de code de transformation, l'importation et l'exportation de modèles au format divers (XML, XMI, etc).

## 1.3. Internet des objets (IoT)

Comme définit dans [53], " *L'Internet des Objets est un réseau de réseau qui permet, via un système d'identification électronique normalisé et unifié, et des dispositifs mobiles sans fil, d'identifier directement et sans ambiguïté des entités numériques et des objets physiques et ainsi de pouvoir récupérer, stocker, transférer et traiter, sans discontinuité entre les mondes physiques et virtuels, les données s'y rattachant.*"**.**

L'Internet des Objets est un concept en pleine expansion ces dernières années. Il permet de relever certains défis technologiques auxquels la communauté fait face dans la vie de tous les jours. Dans cette revue de littérature, nous avons pour objectif de présenter les développements récents des technologies de l'IoT et leur utilisation dans divers domaines en générale, et dans la gestion de processus et des pratiques logistiques en particuliers. Nous accordons une importance particulière à la collecte et le traitement des données issues des capteurs et des puces RFID, et à leur utilisation pour une chaîne logistique communicante, intégrative, flexible et collaborative [123, 124, 125, 126].
Cette revue de littérature présente l'état de l'art des technologies de l'IoT, les protocoles de communication et les plateformes dédiés à ce concept, leur application à la gestion et l'exploitation de la chaîne logistique dans son ensemble. Plusieurs travaux de recherche ont abordé et développé ce sujet d'actualité avec des études détaillées de synthèse au sujet des middlewares, des protocoles de communication et des plateformes de service et de tests [39, 40, 53, 42]. La première section présente quelques concepts de base au sujet de l'IoT, suivi par une revue de littérature sur ses technologies, les protocoles de communications et les réseaux propres à l'IoT. La section 3 est consacrée aux architectures des plateformes existantes. Dans la section 4, nous présentons les protocoles et réseaux de communication dédiés à l'IoT. Cette revue se termine par l'utilisation des technologies de l'IoT pour les problèmes de gestion, de pilotage et de coordination de flux et pratiques logistiques (production, distribution, transport, maintenance, vente, marketing, management), à partir des informations collectées par des capteurs sur des objets physiques. Et enfin un dernier aspect important est abordé au dernier paragraphe, celui de la sécurité et la gestion des droits d'accès aux données privées.

### 1.3.1. Concepts et technologies de l'IoT

L'internet des objets est une révolution technologique dans le domaine de l'informatique et des télécommunications [39]. L'IoT fait référence à une variété d'équipements et de systèmes d'informations de détection tels que les réseaux de capteurs, des dispositifs de lecture (RFID, code à barres), de systèmes de localisation et de communication courte portée basés sur la communication machine à machine (M2M), à travers le réseau Internet pour former un réseau plus grand et plus intelligent [53]. Cette révolution est basée sur une évolution constante de l'Internet, des technologies et des logiciels, des protocoles de communication, des capteurs embarqués qui ne cessent d'être améliorés, des objets physiques de plus en plus intelligents et capables de fournir des

informations et de percevoir en temps réel leur environnement [40]. L'IoT peut être vu sous deux angles, soit centré sur l'Internet ou centrer sur l'objet. Quand elle est centrée sur Internet, les services sont le point principal de son architecture et les objets y contribuent en alimentant par des données. Lorsqu'elle est centrée sur l'objet, le centre de l'architecture devient l'objet et on parle de Cloud des objets. Le Cloud des objets apparaît donc comme une plateforme d'objets permettant un usage intelligent des infrastructures, des applications et de l'information à un coût réduit.

L'IoT repose sur un large panel de technologies, de protocoles, de réseaux et de concepts avec des infrastructures réseaux, de nouvelles plateformes logicielles, matérielles et de services. L'IoT est en particulier associé à l'identification et la traçabilité via l'intégration des puces RFID (Radio Frequency Identification Systems), le web sémantique, les nanotechnologies, la mobilité, l'ubiquité et le crowdfunding [41]. Cette mutation constante et évolutive des technologies et l'avènement de nouvelles plateformes, de nouveaux services et de nouvelles architectures entraînent de nouvelles perspectives, de nouveaux marchés avec des enjeux économiques, sociaux, politiques, éthiques, sécuritaires et réglementaires larges et variés .

Nous citons comme défis à relever l'intégration et le partage de données sur des plateformes Cloud, la sécurisation des données personnelles des utilisateurs (liberté et confidentialité), la bonne gouvernance (transparente et démocratique), l'harmonisation des standards, des réseaux et les aléas de la compétition économique [53]. La gestion de la chaîne logistique s'inscrit dans cette perspective et occupe une place prépondérante parmi les champs d'application de ces nouvelles technologies et concepts liés à l'IoT. Nous pouvons citer à titre d'exemple des problématiques traitées dans le domaine de la logistique, comme abordé dans [42], telles que l'aide à la gestion et à la prise de décision, l'optimisation des stocks, l'amélioration de la qualité de service, l'identification par radiofréquence, le suivi temps-réel des produits et des processus.

L'internet des objets vise à connecter des objets entre eux via les protocoles d'Internet. L'objet représente ici tout ce qui nous entoure (machines, téléphones mobiles, ordinateurs, capteurs) [44]. Pour atteindre cet objectif, il est impératif de pouvoir identifier les objets, leur attribuer une interface virtuelle afin qu'ils puissent communiquer avec leur environnement. Il est important de noter que plus d'une dizaine de milliards d'objets seront connectés d'ici 2020 [43]. Les domaines technologiques couverts par l'IoT sont larges et variés. Dans ce qui suit, nous présentons une liste non exhaustive des différents concepts et technologies de l'IoT et leurs applications dans le domaine de la logistique.

Plusieurs technologies sont utilisées pour faire communiquer un objet avec l'Internet, parmi lesquelles RFID (Identification par Radio Frequence), NFC (Near Fields Communication), le protocole de communication Zigbee [51]. D'autres solutions sont en cours de développement telles que les systèmes d'identification acoustiques, les micro-ondes, les systèmes optiques, l'utilisation de l'ADN, le marquage logiciel ou l'intégration des puces dans la conception des objets [51, 52]. La RFID est constituée d'un couple lecteur/étiquette. Le lecteur envoie une onde radio, l'étiquette envoie à son tour une trame d'identification. Une fois la puce alimentée, l'étiquette et le tag communiquent suivant le protocole TTF (Tag Talk First) ou ITF (Interrogator Talk First). Dans le mode TTF

l'étiquette transmet en premier les informations contenues dans la puce à l'interrogateur. En mode ITF, l'interrogateur envoie une requête à l'étiquette, et ce dernier répond par la suite.

Il existe trois types d'étiquettes, les étiquettes passives, actives et semi-actives. Les premières n'ont pas leur propre source d'énergie : une petite quantité d'énergie leur est fournie par le champ magnétique induit par le lecteur au moment de l'identification. Les tags actifs sont quant à eux alimentés par piles, ils sont capables d'envoyer eux-mêmes des informations d'identification sans sollicitation d'un lecteur. Les semi-actifs utilisent un mécanisme hybride: auto-alimentés, ils ne s'activent qu'à la demande du lecteur, permettant une plus faible consommation d'énergie que les tags actifs. La distance de lecture des puces RFID varie de quelques cm à quelques mètres (10 m), et peut aller au-delà (200 m) avec des technologies de communication longue portée [53]. La technologie NFC est le résultat de plusieurs évolutions des microcontrôleurs, des cartes à puce, et des communications à courte portée [54, 55]. NFC est basée sur le même principe que la RFID, c'est-à-dire l'identification par radiofréquence. Elle permet l'échange d'informations à courte distance entre deux objets (un lecteur et une carte) sans contact et fonctionne suivant deux modes, le mode passif et le mode actif. En mode passif, le terminal de l'utilisateur émule une carte à puce et acquiert de l'énergie des radiations du lecteur (téléphone mobile par exemple). En mode actif, le terminal se comporte comme un lecteur d'étiquettes électroniques (code à barres, étiquettes 2D) et possède sa propre source d'énergie (une batterie embarquée par exemple). NFC permet à l'utilisateur d'échanger des informations avec son environnement, notamment dans le domaine des transports, des loisirs, des achats, ou la lecture d'informations sur des panneaux d'affichage. L'utilisation de la NFC facilite la gestion des données de ventes dans la chaîne logistique, la gestion et la validation de tickets de bus dans le transport urbain [55]. Lorsqu'un utilisateur scanne un tag NFC, il peut en outre avoir accès à des informations sur le produit (origine, fabricant, contenu/ingrédients, procédé de fabrication) [56].

Zigbee est un protocole de communication sans fil à bas coût qui permet des échanges à courte distance entre les nœuds d'un réseau WPAN (Wireless Personal Area Networks). Ce protocole est basé sur la norme IEEE 802.15.4 qui spécifie les protocoles de communication entre les couches physiques et liaison de données du modèle OSI, en définissant trois types d'équipements: les FFD (Full Function Devices) qui sont des équipements à fonctionnalité complète, les RFD (Reduced Function Devices) équipements à fonctionnalité réduite, et les coordinateurs de réseau. Les FFD coordonnent l'ensemble du réseau, ce sont des coordinateurs PAN (Personal Area Network), routeur ou dispositif relié à un capteur. Les RFD (Reduced Function Device) sont des équipements à fonctionnalité réduite, conçue pour des applications simples comme l'allumage d'une lampe. Les RFD ne peuvent communiquer qu'avec un FFD [57]. Parmi les avantages que procure ce protocole de communication, nous pouvons citer la faible consommation d'énergie, l'utilisation optimale de la bande passante, et son faible coût de mise en œuvre. Ces avantages permettent d'adopter le protocole Zigbee dans les environnements embarqués et les réseaux industriels, ou le développement de nouveaux produits basés sur ce protocole [57].

**Les Technologies de marquage et de détection dans l'IoT**

Plusieurs techniques sont utilisées dans le marquage et l'identification dans l'IoT. Les marqueurs ADN (enchaînement de base azoté: guanine, cytosine, thymine, adénine), utilisent un mélange de produits (liquide ou poudre) pour mettre au point un code unique d'identification et d'authentification du produit marqué, typiquement les produits pétroliers et pharmaceutiques, et les pièces techniques de l'industrie. Basés sur l'ADN synthétique, les marqueurs ADN sont en théorie impossibles à contrefaire, au vu du nombre de combinaisons sur les segments de base (G-C-T-A) [70]. La RFID permet, à l'aide d'un identifiant contenu dans une puce, d'adresser beaucoup plus d'objets que les codes à barre et les codes OCR (Reconnaissance Optique de Caractères). Les tags RFID sont maintenant largement incorporés dans des objets logistiques (containers, wagons, palettes, cartons, chariots, bacs plastiques, vêtements) à des fins de traçabilité, de suivi temps réel et de coordination des flux de produits. La numérisation 3D de la zone de fabrication du produit permet de mettre en place un code dit hybride qui permet d'identifier l'objet de façon unique. D'autres techniques telles que le code matriciel (Datamatrix, QR code, Maxicode) [74] sont utilisées pour le tri des colis dans le transport du courrier. Des techniques de marquage comme le code numérique, les nanotraceurs (électroluminescents, nanoparticules), la biométrie (reconnaissance faciale, empreintes, voix), les hologrammes, sont également utilisés pour l'identification, la traçabilité, et la coordination des objets logistiques.

### 1.3.2. Architectures et protocoles de communication dans l'Internet des Objets

**Architectures**

De nouveaux modèles d'architecture permettent d'intégrer des capteurs et le réseau Internet. Cette communication entre un capteur et le Cloud se fait par une couche virtuelle qui implémente le fonctionnement des capteurs réels. Une telle couche donne naissance à de nouvelles architectures appelées réseaux de capteurs (Sensor Cloud) ou Cloud des capteurs virtuels [45]. Avec une telle architecture, il est possible de créer des services basés sur des capteurs virtuels, c'est-à-dire des environnements de capteurs distribués géographiquement et pouvant être utilisés à la demande par plusieurs utilisateurs.

Dans le domaine de la logistique, et en particulier dans la gestion des parcs éoliens, de telles architectures sont utilisées pour la transmission de données à des fins de maintenance préventive. Dans ce domaine, beaucoup d'autres applications sont en cours afin de coupler des algorithmes d'optimisation mathématiques à des services web pour mieux exploiter les données récoltées par les capteurs afin de fournir de meilleurs outils et services de maintenance [46]. Avec l'IoT, des milliards d'objets seront connectés à l'Internet, chaque objet possédant éventuellement ses propres capteurs. Cet écosystème pose des problèmes d'identification et de localisation d'objets et de services (capteur, actionneurs, palettes, containers, services, etc) de façon unique dans l'Internet. Ceci rend plus difficile encore la gestion de l'énorme volume d'informations générées par ces réseaux de capteurs. Pour répondre à cette problématique, plusieurs protocoles et algorithmes ont vu le jour, parmi

lesquels CASSARAM, un middleware proposant des modèles de découverte basés sur la sémantique; l'ONS (*Objects Naming Service*) [47] pour adresser de façon unique des objets sur Internet afin de faciliter la recherche et l'identification des objets, améliorer la communication entre les objets et les plateformes Cloud. L'ONS est un service de nommage qui permet de diffuser des informations sur la source d'un produit, depuis le fabricant jusqu'au consommateur. Il est basé sur le même principe que le DNS (*Domain Name System*). Pour y accéder il suffit de connaître l'identifiant du produit EPC (*Electronic Product Code*), ou le GTIN (*Global Trade Item Number*) [47, 48].

Plusieurs architectures (2-Tiers, 3-Tiers) couplant la chaîne de production à l'IdO ont été proposées dans le but de mettre en place des plateformes de fabrication basées sur le Cloud Computing [48, 49]. Il en est de même pour les outils permettant de piloter la chaîne de production à partir d'étiquettes RFID, à l'instar de la gamme de produits IFM utilisé dans la chaîne de production. Une application de ces nouvelles architectures basées sur l'IoT en logistique est la chaîne de production où il est question de partager les ressources (machines, robots) et les capacités de production de façon optimale, et faire de l'allocation de ressource à la demande [50]. Une autre fonctionnalité de cette application est la possibilité d'utiliser ces protocoles pour renseigner le client sur l'origine et le contenu du produit. En intégrant les données des capteurs à des plateformes Cloud, les architectures de l'IoT donnent des possibilités d'utilisation plus larges qui dépassent la sphère de la logistique.

**Protocoles de communication et réseaux**

Avec l'avènement de l'IoT, un autre défi est celui de la mise à disposition des réseaux de communication fiables, tant au niveau des infrastructures qu'au niveau des protocoles de communication. Ce défi est lié entre autres à la mobilité des objets, à l'hétérogénéité des données et des plateformes, à l'accès à l'information depuis n'importe quel lieu, à n'importe quel moment et à travers n'importe quel dispositif (PDA, Smartphone, tablette), ce qui rend plus ardue la standardisation des protocoles et des algorithmes. De plus, le volume de données à transmettre par les capteurs pose un grand souci sur la disponibilité de la bande passante, d'où la nécessité de mettre en œuvre des réseaux adaptés à ces nouvelles contraintes. Dans la littérature, il est mentionné deux approches, l'utilisation de réseaux sans fil courte portée (Zigbee, Wifi) qui permettent de connecter les objets à l'Internet via une passerelle ; et les réseaux cellulaires classiques large bande (4G, 3G).

L'utilisation de réseaux cellulaires ultra bas débit (*UNB : Ultra Narrow Band*) est en plein essor du fait que la plupart d'objets n'ont pas besoin de la bande passante mise à leur disposition dans les réseaux haut-débit, mais plutôt de réseaux à très faibles coûts et à très faible consommation d'énergie. Parmi les acteurs des réseaux ultra bas-débit, nous pouvons citer Sigfox, Neul et On-Ramp. Les réseaux Sigfox sont caractérisés par leur structure hiérarchique; des serveurs qui vérifient l'intégrité des données et routent les messages vers des systèmes d'information, des modems UNB qui communiquent avec des stations de base ou cellules pour couvrir des zones larges, et des stations de base qui routent les messages vers les serveurs.

D'autres solutions existent pour résoudre des problèmes de communications dans l'IoT, à savoir l'intégration de réseaux ubiquitaires à communication sans fil pour gérer la

connectivité des objets [58, 122], la virtualisation des ressources réseaux (physiques et virtuelles) pour faciliter le partage et la disponibilité des ressources, les réseaux Xbee [57], l'utilisation des capteurs hertziens pour des applications domotiques, la surveillance de l'environnement, ou la sécurité des portails. Ces technologies sont utilisées dans la logistique de transport urbain pour éviter la collision entre véhicules [59], la localisation des personnes dans les zones à risques par l'utilisation de capteurs sans fils, la prédiction du temps de transport pour les marchandises, ou dans la logistique médicale pour collecter des informations lors d'un transport d'organe [60].

**Géopositionnement et tracking des objets dans l'Internet des Objets**

La puce GPS (*Global Positioning System*) est actuellement le système de repérage le plus utilisé dans le monde. En effet, elle s'intègre facilement dans les dispositifs mobiles, et permet de transmettre la position du mobile en temps réel aux applications dans divers domaines: le transport, les services d'urgence, la météo [72, 121].
Parmi les techniques de localisation nous citons deux principaux algorithmes: La Triangulation et la Trilatération. La trilatération est basée sur La distance entre le mobile et la station de base. La position du mobile est déterminée à partir des distances estimées depuis trois stations de bases distinctes minima. La triangulation utilise quant à elle la direction du signal provenant du mobile. La localisation dans ce cas se fait en interprétant les angles que fait la direction du signal et les antennes des stations de base. Avec l'arrivée des tags RFID et des périphériques à faible consommation d'énergie, d'autres solutions de localisation émergent, comme RSN (*Radar Sensor Network*) qui utilisent des réseaux à faible puissance et des radars Doppler (5,8 GHz) pour estimer la position et la vitesse de la cible dans les WSN (*Wireless Sensor Networks*), en utilisant un filtre de Kalman étendu. Cette technique permet aussi de localiser des cibles non coopératives utilisant des capteurs actifs [73]. Rappelons que les cibles non coopératives sont des objets qui ne renvoient pas de signaux de localisation pour que l'on puisse déterminer leur position en temps réel. Une des applications de cette technique est la plateforme *iRobot* qui permet de connecter un robot à un PC Linux, et qui fournit une librairie pour déterminer, contrôler ou programmer la trajectoire, la vitesse ou la direction d'un robot et de récupérer aussi beaucoup d'autres informations sur le Robot (distance parcourue, vitesse). D'autres frameworks s'intéressent particulièrement à la localisation *Indoor* pour l'optimisation des tournées ou la sécurité des personnes. C'est le cas avec ETICOM-FRAMEWORK de Etineo [61]. L'IETF (*Internet Engineering Task Force*), un organisme proche du W3C chargé du développement et de la promotion des standards d'Internet a développé le protocole HIP (*Host Identity Protocol*) dont le but est de séparer la partie localisation de la partie identification, celle chargée de l'identification de l'hôte sur un réseau. Ce protocole est basé sur une infrastructure à clé publiques (PKI) et permet entre autres le multi-homing et la mobilité IP [62].

### 1.3.3. Evaluation de performances et enjeux de l'Internet des Objets

**Evaluation des performances**

Mesurer les performances d'un système dans l'IoT pose encore beaucoup de problèmes, du fait que les objets doivent être testés dans des conditions réelles d'utilisation. Une autre difficulté est due au fait que les systèmes dans l'IoT sont la plupart du temps basés sur des capteurs/actionneurs et requièrent une interaction plus ou moins forte avec les opérateurs humains. Ils peuvent intégrer plusieurs technologies et plusieurs disciplines, ce qui complexifie considérablement les processus de tests et d'évaluation des performances de tels systèmes. Il existe dans ce domaine plusieurs solutions de tests à différents degrés de réalisme, classées selon leur architecture (2-Tiers ou 3-tiers) et les domaines couverts. Les bancs d'essai peuvent proposer des fonctionnalités génériques (MoteLab, WISBED, SenseLab) ou spécifiques à un domaine d'application (CitySense, Friedrichshafen, Oulu Smart City). En fonction de leurs architectures, les environnements d'évaluation peuvent être classés en deux catégories, selon que l'architecture soit 2- tiers (Capteur- Serveur) ou 3-tiers (Capteur-Serveur-Internet). Pour les architectures 2-tiers nous pouvons citer MIRAGE, Vinelab, City Sense, FrONTS, dont la limite est leur incapacité à prendre en charge la couche réseau. Les architectures 3-tiers telles que TWIST, INDRIYA, prennent en compte la couche réseau pour faciliter la communication entre les objets et les serveurs de test, en offrant plus de flexibilité et des gains en performance. Ces solutions de test peuvent intégrer un ou plusieurs systèmes d'exploitation utilisé généralement dans le domaine de l'embarqué (TinyOS, iSense, MoteRunner, Contiki, Sunspot) [71].

L'un des critères de performance d'un système est la sécurité. Dans l'IoT la sécurité est l'un des défis majeurs que nous abordons dans le paragraphe suivant.

**Sécurité dans l'Internet des Objets**

La sécurité est un des problèmes majeurs de l'internet en général, et de l'Internet des Objets en particulier. Plus de 70% d'objets connectés sont vulnérables aux attaques d'après le cabinet d'analyse VDC. Le programme EagleEye, mis au point par Dan Tentler, permettrait de prendre le contrôle de près d'un million de webcams via le moteur d'objets connectés Showdan. La question de la sécurité tourne autour de quelques points centraux parmi lesquels la protection des données personnelles et confidentielles contre des intrusions de toute sortes (lecture non autorisée, falsification, usurpation d'identité, espionnage). Il y a aussi la protection des canaux de routage de l'information contre des attaques passives (écoute du réseau), la protection de l'intégrité des capteurs. Il faut donc porter une attention particulière sur la protection des périphériques connectés contre les différentes formes de piratage: détournement de capteurs, accès aux données de vidéo-surveillance et autres données confidentielles, l'authentification. Pour gérer tous ces problèmes, des protocoles, des algorithmes et des architectures ont été proposés, parmi lesquels APHA (*Aggregate-Proof based Hierarchical Authentication scheme*) pour la transmission sécurisée de données dans les réseaux ubiquitaires et en couches [67]. Nous citons également les algorithmes

cryptographiques utilisant des opérations arithmétiques qui proposent des architectures assez légères en terme de calcul et de ressources, adaptés à l'environnement de l'embarqué et de l'IoT [68]. D'autres protocoles proposent des solutions basées sur la structure des ressources web, en couplant des informations liées au contexte de l'objet aux informations d'identification usuelles pour résoudre le problème des permissions et de contrôle d'accès décentralisé aux ressources et informations publiées par les objets dans le Cloud [69].

**Enjeux de l'Internet des Objets**

Avec l'avènement de la montre **Galaxy Gear** de Samsung et les lunettes connectées **GooGle Glass** de Google**,** l'internet des objets devient un écosystème dont l'industrialisation n'est plus à prouver. Dans les années à venir, il ne sera plus question de gérer seulement la connexion des objets, mais de gérer aussi le volume d'objets connectés, estimé à 25 voire 50 milliards d'ici 2020 pour le marché de l'IoT, selon la commission européenne. Les technologies de l'internet des objets sont classées sur 5 principaux axes [53], **l'identification de l'ensemble des objets** connectés à l'aide du protocole IPv6, la **gestion du stockage** de l'ensemble des données collectées sur les objets via les **Big Data**, donner la possibilité aux utilisateurs tiers d'accéder aux données via l'**open Data , le Crowdsourcing** qui consiste à partager les données entre les consommateurs et de créer de la valeur ajoutée, et la mise en place des infrastructures et des moyens de transmission pour acheminer les données collectées via les réseaux sans fils (wifi, NFC, Bluetooth, …).
L'internet des objets est une révolution dont les enjeux sont multiples, tant politiques, économiques que sociaux. Parmi ces enjeux nous pouvons citer les **enjeux réglementaires** qui se préoccupent de la politique d'accessibilité et de protection des données collectées et de l'harmonisation des standards, les enjeux scientifiques et les enjeux économiques [53] que nous détaillons dans les paragraphes suivants.

**Les enjeux scientifiques et technologiques liés à l'IoT**

- L'interaction machine/machine et homme/machine

L'une des finalités de l'IoT est de fiabiliser, améliorer et faciliter la communication entre l'humain et la machine d'une part, et entre les machines d'autre part. Il s'agit concrètement d'intégrer des capteurs capables de transmettre des informations en temps réel dans plusieurs domaines dont la logistique, le transport, l'automobile, le secteur hospitalier, etc.

Au niveau de l'interaction homme/machine [31, 32, 33] il s'agit de réduire la complexité des technologies, tout en améliorant le design, la gestion de l'autonomie, la résistance des objets et l'intégration de l'environnement de l'utilisateur. Les exemples concrets de cette mouvance sont : le **jawBone** pour le poignet de main et le **Fliike** pour le logement, et les montres connectées sans boutons de commande. Pour ce qui est de la communication machine à machine (M2M), on imagine un système intelligent se connecter à un serveur ou une plateforme ubiquitaire pour récupérer des informations ou envoyer des données relatives à son environnement, comme dans le domaine de l'automobile où un automobiliste peut se connecter pour avoir des informations sur les parkings qui se trouvent

dans son périmètre et vérifier leur disponibilité, ou dans le domaine de la santé où un objet connecté peut envoyer des alertes pour un meilleur suivi du patient.

- L'interopérabilité des objets

L'interopérabilité des objets connectés consiste ici à déclencher un service ou un ensemble de services suite à une action [32, 33]. Un accident sur la route pourrait déclencher automatiquement un appel aux services d'urgence [32], ou une notification automatique des services d'assurances concernés. Cette nouvelle façon de procéder va créer des opportunités de marchés sur les objets connectés. Un exemple concret sur l'interopérabilité des objets connectés sont la plateforme IFTTT (If This, Then That) qui permet de relier 71 services sur le Cloud (Twitter, Facebook, Evernote, etc.) et de créer un ensemble de services suite à une seule action. D'autres applications ont vu le jour pour des usages personnels (animaux familiers, santé, etc.), la gestion de production et des flux logistiques (récupération des données des capteurs à distance, traçabilité et géolocalisation des marchandises, etc.).

- Les middlewares

Les middlewares permettent de gérer les interfaces entre différents systèmes et jouent un rôle critique dans l'IoT. Ils permettent aux utilisateurs de connecter par exemple tout type d'appareils (actionneurs, téléphones mobiles, serveurs, capteurs, automobile, etc.). Coupler au Cloud Computing, les middlewares permettent de collecter, de traiter, de stocker et de consulter des informations en temps réel de façon illimitée. Les middlewares contribuent à l'amélioration des transports publics, la gestion de la sécurité (webcam connectées, systèmes d'alarme, etc.), la gestion des urgences, etc.
Les middlewares jouent aussi un rôle critique dans l'extraction et le filtrage de données RFID depuis des lecteurs, et de leur transmission aux systèmes d'information et aux serveurs (ERP, SCM, CRM, etc.).

- L'informatique ubiquitaire

L'informatique ubiquitaire regroupe les technologies pour la détection et le changement de contexte d'application. Il s'agit de donner accès à un objet virtuel représentant un objet réel en fonction du contexte de l'utilisateur, d'adapter un service à un contexte particulier, pour une personne donnée. Cette nouvelle façon de consommer les applications sur internet est renforcée par l'ONS (Object Naming Service), un protocole lié à l'IoT, qui permet d'identifier un objet virtuel de façon unique sur internet, et par la découverte de services (Discovery Services) [17]. L'informatique ubiquitaire recoupe aussi les notions de web sémantique, de solutions RFID, de Service Oriented Architecture (SOA), etc.

**Les enjeux liés à la réglementation de l'IoT**

L'objectif ici d'éviter le vide juridique sur les retombées que pourraient avoir l'usage de l'internet des objets. La première étape consiste à rassurer les utilisateurs, les organismes de protection des droits d'utilisateurs et les services d'état qu'il est possible d'identifier un objet de façon unique et donc d'identifier son créateur et ses utilisateurs. Le protocole ONS (Object Naming Service), basé sur le même principe que le DNS (Domain Naming Service), contribue fortement à cette identification.
Il est aussi question de gérer les problématiques liées à la confidentialité des données des utilisateurs sur les serveurs, de la protection et la sécurité des informations qui transitent par le net (plus de 70% d'objets connectés sont vulnérables aux attaques selon le cabinet d'analyse VDC) [32, 33, 34]. Le 7$^{eme}$ PCRD de l'Union Européenne se penche sur ces questions et vise à répondre à quelques questions troubles sur l'usage de l'Internet des Objets. Le programme FP7 cherche des solutions qui peuvent favoriser la croissance économique, guider l'innovation, concevoir des réseaux ubiquitaires sans qu'ils ne soient intrusifs, concevoir de nouveaux réseaux de services et d'infrastructures du futur (sans fil, haut débit, mobile, etc.), tout en respectant la vie privée des utilisateurs, les accords de confidentialité et la sécurité des données partagées.

**Les enjeux économiques**

Selon l'Union Européenne, le marché des objets connectés devrait atteindre plus de 10 milliards de dollars en 2016. Aujourd'hui la moyenne est de 3 à 6 objets connectés par famille, et devrait atteindre 15 à 20 objets par personne en 2020 (smartphone, tablettes, électroménager, montre, bracelet, etc.). Dans ce contexte, les entreprises doivent mettre en place des stratégies industrielles, de commercialisation, de marketing et de partenariat pour définir leur offre de service et de produit dans ce nouveau marché en plein essor [31, 32, 53]. Les avantages économiques de l'IoT seront les suivantes :

- Optimisation de la chaîne d'approvisionnement
- Efficacité des coûts
- Amélioration de l'expérience consommateur

Par ailleurs, il existe tout de même des défis à relever pour une mise en place d'une solution basée sur l'IoT. L'intégration de systèmes et le partage de données sont les premiers obstacles à franchir, vu l'hétérogénéité des systèmes et des technologies participant à l'architecture d'une solution IoT. Vient ensuite l'identification du secteur d'activité où le retour sur investissement sera le plus rapide. En effet, les entreprises préfèrent agir sur un segment d'activité où le retour sur investissement est immédiat plutôt que d'agir sur la chaîne d'approvisionnement dans son ensemble. L'IoT permettra aussi de réduire le gaspillage industriel (44 millions de vaccins par an en raison du non respect des températures requises), d'améliorer la gestion des retours d'articles et de commandes via des numéros de lots, d'envoyer des alertes automatiques aux consommateurs sur des produits critiques.

**Les enjeux liés à la maîtrise des flux logistiques**

- **Problème d'optimisation des coûts dans la chaîne logistique globale**

Le caractère distribué de la chaîne logistique engendre de nombreux problèmes d'optimisation. Les mathématiques, la recherche opérationnelle (RO) et l'Intelligence artificielle (IA) donnent un ensemble de méthodes et outils qui permettent de traiter quelques-uns de ces problèmes à partir de la modélisation des flux logistiques [35, 36, 37]. L'intelligence artificielle distribuée et les systèmes multi-agents permettent de modéliser et doter les systèmes logistiques d'intelligence, ce qui permet de prendre des décisions au niveau local (avec la notion d'agent) et de rendre ainsi les entités du système plus autonomes et plus interactives. Les problèmes d'optimisation sont généralement regroupés en trois catégories, stratégique, tactique et opérationnelle et les fonctions d'optimisation ont généralement les mêmes objectifs, la minimisation des coûts (coût de production, coût de transport, coût de stockage, etc) tout en maximisant la qualité du produit, et de service (QoS).

- **Amélioration des outils d'aide à la décision**

Les tableaux de bord d'aide à la décision dans le domaine de la logistique sont jusqu'ici orientés vers des données et indicateurs financiers pour la plupart (valeurs des stocks, coût de production des articles, valeur ajouté économique, prix des actions à la bourse, coût de transport, etc.). Le nouveau challenge dans ce domaine est de doter les entreprises, et en particuliers les acteurs de la chaîne logistique, de nouveaux outils d'aide à la décision qui prennent en compte, outre les facteurs économiques, l'environnement de travail, le climat social, l'impact des politiques publiques et régionales, et des indicateurs sur le comportement des autres acteurs de la chaîne dans la fonction d'optimisation globale. Dans cette optique, une nouvelle catégorie de systèmes a vu le jour au nom du SIMPeG (Système Intégré de Mesure de la Performance Globale), et beaucoup de concepts tels que TBP (Tableau de Bord prospectif), le « Stakeholder Approach », le « Balanced Scorecard » qui misent sur les fonctions principales d'un système d'aide à la décision, à savoir :

- La fonction de coordination qui sert à mesurer la performance des objectifs primaires et secondaires de l'entreprise.
- La fonction de suivi permettant de comparer les résultats aux conditions des uns et des autres.
- La fonction de diagnostic permettant de comprendre comment la performance des processus affecte la performance organisationnelle.

- **L'Échange de Données Informatisées (EDI)**

Le flux d'information est l'un des principaux flux gérés par les entreprises de la chaîne logistique. Il permet de gérer et coordonner le flux physique de matières premières aux produits finis, ou les flux de matière dans la chaîne de production. Une importance capitale est donc accordée aux flux d'informations dans la chaîne logistique comme dans tout système informatisé, tant sur la qualité et la fiabilité de l'information que sur sa transmission et sa mise à disposition pour les acteurs concernés. Le défi actuel est de passer des ERP classiques vers des modèles de systèmes plus collaboratifs, plus intégratifs, qui permettront de mieux coordonner les actions entre différents acteurs dans la chaîne

globalisée, de partager l'information pour assurer une gestion partagée des approvisionnements. L'idée étant d'atteindre un niveau de collaboration où la demande, la planification des activités, les approvisionnements se font conjointement entre tous les acteurs impliqués. Cette démarche est connue sous le nom CPFR (Collaborative Planning Forecasting and Replenishment).

**Les enjeux économiques de la maîtrise des flux logistiques**

Les enjeux économiques de la maîtrise des flux logistiques sont nombreux, et ont fait l'objet de plusieurs travaux. Ces enjeux sont fonction de la taille (petite, moyenne ou grande entreprise), de sa structure fonctionnelle et organisationnelle, de ses activités et surtout de la conjoncture politico-économique. Sans faire une liste exhaustive, nous citons quelques défis auxquels les entreprises de la chaîne logistique font face actuellement.

- ❑ La réduction du gaspillage

Le gaspillage au sens large désigne toute activité sans valeur ajoutée, inutile au regard du processus de fabrication du produit fini. Il est important de pouvoir déterminer et inventorier les postes ou les activités de gaspillage, afin de pouvoir les supprimer, ou à défaut les réduire dans la chaîne de production. Les activités de gaspillage peuvent être classées en plusieurs catégories. La surproduction; les délais d'attente; les activités de manutention et de transport; les traitements inadéquats; les stock inutiles; les mouvements inutiles; les défauts de fabrication.

Une bonne modélisation des flux permet de palier aux activités de gaspillage afin d'atteindre des objectifs tels que l'identification, l'analyse et la diminution de mauvaise utilisation des ressources dans le processus d'approvisionnement, de fabrication ou de distribution; la séparation des activités à valeur ajoutée et des activités sans valeur ajoutée.

Les activités sans valeur ajoutée sont supprimées du processus du fait qu'elles présentent une source de pertes. La réduction du gaspillage procure à l'entreprise des avantages concurrentiels tels que:

- ✓ Localisation et dimensionnement des unités de production
- ✓ Localisation des produits, des entrepôts et des hubs
- ✓ Optimisation des flux: fournisseurs-usines-entrepôts-clients
- ✓ Dimensionnement de flottes de camions, voitures, avions, wagons, navires (fleet management)
- ✓ Affectation des zones de chalandise aux entrepôts
- ✓ Localisation des stocks
- ✓ Optimisation des plans de transport
- ✓ Gestion des flux retours.

- ❑ Fiabiliser les clients et réduire les délais de livraison

L'un des enjeux majeurs de la maîtrise des flux est de permettre aux entreprises de fidéliser leur clientèle et de réduire les délais de livraison. Pour ce qui est de la fiabilisation

de la relation client, il est question d'utiliser le flux d'informations lié aux clients pour faire des analyses de comportement, des campagnes de marketing, de transformer ces informations en connaissances et en actions. Du côté fournisseur, l'anticipation des attentes du client permet de réagir promptement et de donner une réponse adaptée à son besoin, ce qui constitue un véritable avantage concurrentiel.

Dans ce domaine, les solutions dite CRM (Customer Relationship Management) existe déjà, mais restent insuffisantes dans une logique de gestion collaborative du flux de marchandises. L'optimisation des délais de livraison entraîne des bénéfices pour le consommateur (disponibilité des produits), la diminution des niveaux de stocks et donc la diminution des coûts liés à la maintenance des produits stockés. Au niveau du fournisseur, les gains sont liés à l'optimisation du processus de fabrication (flux tendu minimisant les flux tampons), meilleur gestion de la planification et des expéditions, et optimisation des stocks de produits finis.

### 1.3.4. Application de l'IoT à la gestion de flux logistiques

Cette partie du document traite des applications des technologies de l'IoT à la chaîne logistique. Nous résumons ces applications en quatre points: l'extraction et le traitement de données à des fins de pilotages de la chaîne logistique, la sécurité et la confidentialité de ces données qui peuvent être privées, les technologies de marquage et de détection d'objets, et enfin l'évaluation des performances d'un système dans l'Internet des Objets.

- **Extraction et traitement des données dans l'IoT**

Les « Big Data » remplacent progressivement et de façon très invasive les bases de données relationnelles existantes depuis les années 1990. En effet, ces bases de données géantes permettent de gérer un plus grand volume de données, leur disparité, leur hétérogénéité et facilitent l'accès en temps réel aux informations, qu'elles soient distribuées ou centralisées. Avec la montée en puissance de l'Internet des objets, les capteurs embarqués sur des dispositifs mobiles permettent de remonter de plus en plus de données (température, pression, position GPS, vitesse, luminosité, rythme cardiaque, etc.). Cette situation complique de plus en plus la tâche des analystes, et de la fouille de données, qui se trouvent face au problème de proposer de nouvelles solutions adaptées à cette forte volumétrie en constante progression, en remplacement aux démarches de fouille traditionnelles qui se basent en général sur des modèles statistiques, la régression linéaire ou logistique [63]. Aussi, il faut noter que la collecte, le formatage et la transmission de données issues de capteurs vers des plateformes Cloud requiert beaucoup d'énergie, ce qui peut s'avérer très contraignant, si bien qu'il faut maintenant penser à déplacer les tâches de traitement vers des périphériques dont la contrainte en énergie est moins forte que celle du capteur (Smartphone, PDA, ordinateur). Il existe des middlewares qui ont été développés dans ce sens (*MOSDEN : Mobile Sensor Data Processing Engine*) [64].

Côté protocoles, ceux utilisés maintenant depuis quelques années pour les applications web (*SOAP : Simple Object Access Protocol*, *REST : Representational State Transfert*)

servent également pour les échanges entre objets. Les langages de fouille de données (*SPARC QL, ETL : Extract Transform and Load)* facilitent l'extraction et le traitement de données provenant de sources diverses, de bases de données distribuées ou non, relationnelles ou NoSQL. Ces langages, ces standards et ces protocoles visent aussi à intégrer l'internet des Objets avec le Cloud Computing pour donner naissance au Cloud des objets (Cloud of Things) [65]. L'objectif est de créer des applications plus intelligentes, de rendre les données collectées sur les objets plus accessibles, et surtout plus pertinentes et significatives.

Les applications sont diverses et variées dans le domaine de la logistique, de la production à la gestion des relations entre tous les acteurs de la chaîne logistique y compris le client. Au niveau de la production, des capteurs (tags RFID) sont intégrés sur des produits (palettes, sacs, bacs) pour stocker des informations sur leurs contenus, des informations de traçabilité (opérations effectuées sur les produits, origine du produit et de ses composants, etc). Des applications de pilotage intelligent de la chaîne de production ont vu le jour, via des lecteurs RFID montés à bord des convoyeurs pour aiguiller automatiquement des articles ou des palettes en fonction des opérations à effectuer ou de leur contenu. D'autre part, il est possible d'intégrer la chaîne de production et les systèmes de gestion de l'entreprise (*MES : Manufacturing Execution System, EMI : Enterprise Manufacturing Intelligence, ERP : Enterprise Ressource Planning*), de sorte que les données transmises par des capteurs intelligents au niveau de la chaîne de production puissent être utilisées pour la prise de décision au niveau local et global pour la fabrication de produits [66]. Cette intégration permet aussi le management stratégique et tactique de l'entreprise au niveau global (planification, management, ordonnancement de la production, management des ressources), et sert à optimiser la maintenance des équipements dans tous les domaines (véhicule électrique, éolien, solaire, aéronautique, transport, etc.). Un exemple d'application est celui de BMW qui utilise la technologie *Win River* pour connecter son ordinateur de bord à des opérateurs de télécommunication pour dialoguer en temps réel avec des services Cloud (météo, trafic) à des fins de maintenance du véhicule en cas de panne. Ce service permet aussi de remonter des informations sur l'état des pièces importantes du véhicule à l'aide de capteurs intégrés, ainsi qu'à la remontée des statistiques (informations sur l'état de la route, fluidité de la circulation) et des diagnostics plus précis.

Il est fort de constater que ce panorama de solutions développées est basé sur une multitude d'approches de modélisation parmi lesquelles les approches formelles et semi formelles. Les approches formelles basées sur des outils mathématiques de modélisation et simulation de flux tels que les réseaux de Petri, les algèbres de processus, les graphes. Des méthodes semi formelles plus ou moins graphiques quant à elle telles que UML. D'outils et de méthodes de conception, des middlewares orientés services ou orientés événements, des plateformes Cloud en mode *aaS pour la gestion et le pilotage de données issues du flux.

Des outils et des technologies d'identification orientés PLM (Product Lifecycle Management) pour la traçabilité et la gestion du cycle de vie des entités et des objets logistiques. L'internet des Objets apporte des nouveautés dans la littératures en ce sens que les technologies y afférentes sont vastes et couvrent des domaines variés. Elles peuvent intégrer diverses composantes, des algorithmes, des protocoles de communication, des middlewares, des données et des services de nature différente. Les données récoltées et

échangées dans l'IoT entre capteurs, actionneurs, serveurs, et l'Internet proviennent de sources diverses, sont fortement hétérogènes, disparates et représentent une masse de plus en plus volumineuse. Il est clair que l'exploitation intelligente de ces données représente un atout concurrentiel pour les entreprises, à des fins d'optimisation de la chaîne logistique, de la production à la livraison au client final, en passant par le transport, la logistique de maintenance, le marketing et la gestion de la relation client.

## 1.4. Panorama des méthodes et outils développées dans la littérature

### 1.4.1. Méthodes et approches de modélisation

**Méthodes mathématiques:** Dans l'approche mathématique de résolution de problèmes, l'importance est accordée à l'aspect formel des outils utilisés [18]. Il s'agit ici de pouvoir prouver via une démarche mathématique que le modèle obtenu répond à la problématique posée. Dans une démarche mathématique, les outils sont choisis et utilisés selon le problème à résoudre et les objectifs à atteindre. A titre d'exemple, si nous voulons modéliser la variation de la température ou de la pression d'un fluide, nous le faisons à l'aide des équations différentielles; ce qui n'est pas le cas si nous voulons représenter une carte routière avec des itinéraires et des carrefours. Pour le problème de la carte routière, nous pouvons utiliser les graphs orientés (ou non) où les arcs représentent les tronçons du réseau routier, et les sommets représentent les points d'intersection entre ces tronçons. Les graphes permettent aussi de modéliser et résoudre d'autres problèmes comme celui du plus court chemin entre deux villes, le problème de tournées de véhicules etc.

Les graphes et les autres outils graphiques développés dans le cadre de la théorie des SED (Systèmes à Evénements Discrets) tels que les automates et les réseaux de Petri, sont utilisés pour la modélisation et l'étude de la dynamique des flux logistiques. Ces outils ont prouvé leur puissance et leur efficacité dans la modélisation et analyse des autres classes des systèmes décisionnels dont les systèmes de production et les systèmes de transport. La littérature mentionne aussi l'utilisation des équations aux dérivées partielles (EDP), l'algèbre des dioïdes pour la résolution des problèmes du « juste à temps », l'algèbre de processus (LOTOS, Pi-Calcul, etc.) pour la modélisation des systèmes concurrents ou distribués, et de l'intelligence artificielle distribuée (IAD).

**Approche Systémique:** Un système est un ensemble cohérent d'éléments en interaction dynamique, organisés en fonction d'un but bien défini (finalité du système). On distingue des systèmes fermés ou ouverts, artificiels ou naturels, hiérarchiques ou en réseau.
L'approche systémique [19, 20] ou la systémique est une discipline qui regroupe les démarches théoriques, pratiques et méthodologiques relatives à l'étude de ce qui est complexe et pose des problèmes de frontières, de relations internes et externes, de structure, de lois ou de propriétés émergentes caractérisant le système comme tel. Elle s'applique également sur des problèmes de mode d'observation, de représentation, de modélisation ou de simulation d'une totalité complexe.

L'approche systémique peut jouer un rôle prépondérant dans la démarche de modélisation. Elle est plus basée sur les outils graphiques de modélisation (diagrammes UML, SADT, BPML, etc.) et des standards (NAF, TOGAF, etc.), qui peuvent aller de l'élaboration de modèles qualitatifs à la construction de modèles dynamiques et quantifiés, opérables sur ordinateur et débouchant sur la simulation ou la génération de code.
L'approche systémique permet d'apporter des éléments de réponse à des questions telles que: pourquoi modéliser les flux logistiques? Quelle est la finalité visée dans un processus de maîtrise des flux? Pour quels usages et quels besoins pour les utilisateurs finaux? Elle permet de définir et de valider les fonctionnalités attendues par les utilisateurs potentiels du système en cours de construction, ce que ne peut faire les réseaux de Petri ou d'autres graphes du même type.

**Approche orienté événement (EDA):** Dans l'approche EDA (Event Driven Approach) [21], l'évènement est au centre de la modélisation du système. Un événement est une occurrence identifiable d'une action qui a une signification pour le système à modéliser (par exemple: émission d'un ordre de fabrication). Les acteurs du système (opérateur, transporteur, etc.) sont à l'origine des événements qui modifient le fonctionnement du système et l'astreint à changer d'états. Un événement est un stimulus qui peut être causé par une interruption matérielle ou logicielle, ou par un acteur humain ou non, ou tout simplement par l'environnement du système.

La programmation événementielle sépare la logique de traitement des événements du reste du système. En tant qu'approche de modélisation, EDA est indépendante de la plateforme d'implémentation et vise à améliorer le traitement événementiel, la réactivité et la robustesse du système étudié. L'approche événementiel complète l'approche orientée service, de sorte que les services du système s'abonnent aux événements qu'ils souhaitent traiter. Ainsi, dans les architectures EDA, le bus de service joue un rôle essentiel (supervision et contrôle de SLA: Service Level Agreement, composition de plusieurs services, etc.) dans la médiation entre les services émetteurs et les potentiels consommateurs.

**Approche orientée modèle (MDA) :** Les approches MDA s'inscrivent dans la logique de l'ingénierie dirigée par les modèles [22]. MDA (Model Driven Architecture) est un concept du groupe OMG (Object Management Group) lancé en 2001, qui vise à promouvoir un ensemble de nouvelles pratiques dans la conception d'architectures de systèmes logiciels, où les spécifications sont transformées en modèles. L'objectif des approches MDA est de séparer les architectures applicatives des technologies d'implémentation, afin d'assurer que l'architecture du système puisse survivre aux évolutions technologiques.

Dans une approche MDA, les exigences du client sont spécifiées par le modèle d'exigences CIM (Computation Independent Model), qui définit les services offerts par l'application, et les entités avec lesquelles elle interagit. Contrairement au diagramme de cas d'utilisation UML qui fournit les fonctionnalités de l'application et les acteurs, le CIM définit les processus métier, les exigences, les cas d'utilisation et une vue systémique de la

plateforme. Une fois le modèle d'exigences obtenu, la prochaine étape consiste à concevoir un modèle d'analyse et de conception abstraite indépendant de la plateforme que l'on appelle le PIM (Platform Independent Model). Le PIM structure l'application en module et sous-modules à l'aide des patrons de conception, sans influencer les contraintes techniques de réalisation et les technologies à utiliser. A partir d'un modèle d'analyse et de conception, on peut générer un modèle de code spécifique à la plateforme PSM (Platform Specific Model). Un PSM contient toutes les informations nécessaires à l'exploitation d'une plateforme d'exécution (manipulation de fichier, authentification, etc.). L'approche MDA propose d'utiliser des profils UML pour l'élaboration de modèles de code. Un profil UML est une adaptation du langage UML à un domaine bien spécifique, il permet d'automatiser la génération du code vers une plateforme d'exécution. Pour ce qui est de la transformation de modèles MDA, c'est à dire le passage d'un CIM vers un PIM et d'un PIM vers un PSM, elle est considérée comme une application qu'il faut analyser et concevoir, modéliser ses exigences afin d'assurer la génération automatique de code.

Une norme spécifique a été définie pour la transformation des modèles, le QVT (Query/View/Transform), cette norme est à l'origine du langage ATL (ATLAS Transformation Language) développé au laboratoire LINA à Nantes dans le cadre du projet Eclipse M2M (Model to Model). Un formalisme définit les concepts et les relations entre concepts nécessaires à l'expression de modèles. Le méta formalisme est l'action qui consiste à décrire un formalisme, et les modèles qu'il permet d'exprimer sont appelés méta-modèles. MDA définit un seul méta formalisme, le MOF (Meta Object facility). Il est utile de mentionner le langage OCL qui permet de décrire les contraintes sur le méta modèle, c'est à dire d'exprimer des exigences informelles qui ne peuvent pas être représentées par un diagramme UML.

Le tableau 1 récapitule l'ensemble de ces approches, les outils associés, ainsi que leurs domaines d'application et leurs limites.

**Tableau 1: Synthèse sur les approches de modélisation**

| Approches | Outils | Description | Domaines d'application | Limites |
|---|---|---|---|---|
| Mathématique | ● EDP<br>● Algèbre des dioïdes<br>● Algèbre des processus<br>● CSP<br>● LOTOS<br>● Graphes<br>● RdP<br>● Automates<br>● IA<br>● ... | Utilisation d'outils mathématiques pour une modélisation plus fidèle et plus réactive du comportement des systèmes réels | ● Problèmes d'optimisation<br>● Problèmes de tournées de véhicules<br>● Logistique du juste à temps<br>● Coordination dans les chaînes de production | Ne donne pas une vue globale du système à modéliser, ses fonctionnalités, ses acteurs/utilisateurs et sa finalité. Il est parfois indispensable de combiner plusieurs outils pour une étude complète du système |
| Systémique | ● MODAF<br>● DODAF<br>● UML<br>● MERISE<br>● SART<br>● SysML<br>● LOTOS<br>● ... | Utilisation de modèles graphiques, des composants, des annotations et des standards pour la représentation de l'aspect statique, fonctionnel et dynamique d'un système complexe et communicant. | ● Modélisation des systèmes de production (atelier flexibles, automates, …)<br>● Modélisation de systèmes temps réel<br>● Modélisation de la dynamique des systèmes complexes communicants (LOTOS) | Manque de formalisme pour les outils graphique (UML, MERISE, ...). à compléter par des outils plus formels (LOTOS, CADP, ...) |
| EDA | ● SART<br>● Drools<br>● CEP<br>● ... | Modélisation d'un système en se focalisant sur les événements et les règles qui le stimule dans son environnement, et les événements qu'il émet en réponse à ces stimuli | ● Modélisation des automates, des systèmes à basés sur des événements, des règles et des conditions de déclenchement de ces événements (RDBMS, ...)<br>● Modélisation de systèmes temps réel<br>● Modélisation de la dynamique des systèmes complexes | Manque de vue fonctionnelle du système, peinent à représenter l'aspect statique du système à modéliser. |
| MDA | ● UML<br>● Profil UML<br>● DSML<br>● EMF<br>● MOF<br>● OCL<br>● QVT<br>● ... | Les approches MDA permettent spécifier et de modéliser un système à partir de méta modèles, d'annotations et de contraintes sur les objets manipulés. Elles permettent aussi de définir des systèmes indépendamment des technologies et des plateformes d'implémentation. | ● Ingénierie dirigée par les modèles<br>● Méta-modélisation<br>● Réalisation des DSML et des profils UML<br>● Transformation de modèles. | Manque d'outils accomplis, manque de formalisme comparativement aux méthodes mathématique (RdP, CSP, (Max, +), EDP...) |

### 1.4.2. Outils de modélisation

## Outils formels

- **Les graphes**

Formellement, un graphe [23] est la donnée d'un couple $G = (S, A)$ où $S$ est un ensemble fini de sommets et $A$ est un ensemble de couples de sommets $(s_i, s_j)$.
Dans un graphe orienté [24] les couples de sommets sont ordonnés et appelés arcs, et représentés par un sommet initial et un sommet terminal. Les graphes sont utilisés dans la modélisation de plusieurs problèmes de la vie courante, à l'instar du réseau routier d'un pays qui peut être représenté par un graphe dont les sommets sont les villes et les arcs représentent les itinéraires entre les villes. Si l'on considère que toutes les routes sont à double sens, on utilisera un graphe non orienté et on reliera par une arête tout couple de sommets correspondant à deux villes reliées par une route. Si l'on considère en revanche que certaines routes sont à sens unique, on utilisera un graphe orienté. Les arêtes pourront être valuées par la longueur des routes correspondantes. Étant donné un tel graphe, on pourra s'intéresser, par exemple, à la résolution des problèmes suivants:
- Quel est le plus court chemin, en nombre de kilomètres, passant par un certain nombre de villes données ?
- Quel est le chemin traversant le moins de villes pour aller d'une ville à une autre ?
- Est-il possible de passer par toutes les villes sans passer deux fois par une même route?
- Comment relier un ensemble de villes de façon optimale afin d'assurer des opérations logistiques (gestion des flux logistiques, optimisation des transports, etc).
- Comment organiser une tournée de véhicules (réseau de bus, distribution de marchandises).
Les graphes peuvent être utiles dans la démarche de représentation du réseau de transport (trafic routier, ferroviaire, maritimes, aérien) et faciliter une meilleure appréhension et la modélisation des flux logistiques qui circulent dans ces réseaux.

- **Les automates**

Les automates [25] représentent un moyen ancien très utilisé pour modéliser le comportement dynamique d'un système à événements discrets. Un automate est un quadruplet composé d'un ensemble d'états $S$, d'un alphabet $E$, d'une fonction de transition $T$, et d'un état initial $s_0$. La fonction de transition exprime comment le système change d'état lorsqu'il reçoit un événement, c'est-à-dire le passage d'un état du système vers un autre.

$$\square = (S, E, T, s_0)$$

Il existe plusieurs façons de représenter un automate, soit par son diagramme d'états-transitions, ou sa table de transitions, ou des notations formelles. Il existe plusieurs classes d'automates, nous nous intéressons à deux d'entre elles :

- Les automates finis. Un automate fini est une machine abstraite qui effectue des traitements en utilisant une mémoire bornée. Le concept d'automates finis est lié au

fait que le nombre d'états par lesquels passe l'automate est fini, bien qu'il y'ait la possibilité de passer par le même état plusieurs fois au cours de la reconnaissance d'un mot. L'automate à état fini peut être déterministe ou non, cela dépend de la fonction de transition. Si à partir d'un état donné on peut passer à deux états différents par la même étiquette, l'automate est dit non déterministe, et sera dit déterministe dans le cas contraire. Un mot X est accepté ou reconnu par un automate fini s'il existe un chemin de l'état initial vers un état terminal dont la suite des étiquettes (transitions) vaut X.

- Les automates temporisés. Un automate temporisé est un outil qui permet de représenter des systèmes réactifs à temps continu. Il s'agit des automates avec des horloges, c'est-à-dire qu'en plus des variables d'états il a des variables réelles positives qui représentent le temps écoulé depuis une certaine action. Il peut y avoir plusieurs horloges dans chaque place, ayant des chronomètres différents, de sorte que les transitions entre les places soient conditionnées par des contraintes sur ces horloges. Formellement, un automate temporisé est un 7-uplet qui comprend un ensemble fini d'états S, un alphabet E, un ensemble d'états initiaux $S_I$, un ensemble d'états terminaux $S_F$, un ensemble fini d'horloges $X$, une fonction de transition $T$, et une condition de garde $g$ sur les horloges.

- **Les Réseaux de Petri (RdP)**

Les réseaux de Petri [26] sont un langage formel pour la modélisation des processus gérés entre autres par des phénomènes de synchronisation, de concurrence et de parallélisme. Un RdP ordinaire est un graphe orienté comprenant deux types de nœuds, les places et les transitions, qui sont reliées par des arcs orientés. Les places correspondent aux états du système et les transitions à des événements ou à des actions qui vont être accomplies par le système. Un RdP est connu comme étant un outil de modélisation graphique et mathématique permettant de vérifier les caractéristiques d'un système du point de vue structurel et son comportement dynamique. La puissance des RdP réside dans leur capacité de modélisation, d'évaluation et de validation des systèmes dynamiques en se basant sur des concepts mathématiques et graphiques. Cet outil permet d'analyser les propriétés quantitatives et qualitatives du système modélisé en étudiant les matrices issues du modèle graphique obtenu. Plusieurs classes des RdP ont vu le jour pour modéliser les comportements des systèmes dynamiques et répondre aux diverses problématiques. Nous citons à titre d'exemple les graphes d'événements temporisés (GET), les RdP continus, les RdP hybrides, les RdP colories, les RdP stochastiques, etc. Une classe ou une combinaison de ces classes des RdP peut être utilisée selon la problématique traitée dans les systèmes logistiques. A titre d'exemple, pour la modélisation et l'analyse des objets logistiques en s'intéressant à l'évolution de leurs états au sein d'une chaîne logistique, on peut utiliser les GET; et si on s'intéresse à la modélisation des flux d'informations qui circulent dans le système on peut utiliser les RdP continus, ou encore les RdP hybrides dans le cas de l'étude des composantes discrète et continue d'un système logistique (flux d'objets, et flux d'information).

Les réseaux de Petri ont pour rôle essentiel la modélisation de l'aspect dynamique d'un système, la détermination des points de blocages ainsi que d'autres propriétés qualitatives du système à partir des matrices obtenus.

- **LOTOS**

LOTOS (Language Of Temporal Ordering Specification) [28] est un langage de spécification qui a été conçu pour la spécification formelle de systèmes concurrents en général, et de systèmes distribués en particulier. La notion de système ici fait référence à un ensemble de processus qui interagissent et échangent des données entre eux et avec leur environnement. LOTOS est devenu une norme internationale de spécification depuis 1988.
LOTOS est basé sur les notions de processus et d'événements. Le système est vu comme un ensemble de processus, un processus est un ensemble de sous processus qui sont à leur tour vu comme des processus. Le processus est une entité capable d'effectuer un ensemble d'actions et d'interagir avec son environnement, c'est-à-dire d'autres processus par interception d'événements où action atomique. Les événements entraînent la synchronisation de processus avec échange ou non de données.

- **Spécification à événements discrets**

La spécification à événements discrets (DEVS - Discrete Event Specification) est un formalisme qui a été proposé par B.P. Zeigler en 1976. L'avantage de cet outil réside dans son indépendance des technologies d'implémentation [29], et l'intégration de l'aspect temporelle permettant de représenter les événements et les états. Ce formalisme permet de modéliser des systèmes à événements discrets, des systèmes industriels de production, des systèmes complexes, de simuler et d'étudier leur comportement. DEVS est utilisé dans plusieurs domaines tels que le domaine médical, la défense, et le domaine industriel.

## Outils semi-formels

- **UML et UML-S:** UML [30] est un langage de modélisation orienté objet développé par OMG (Object Management Group) dans le but d'unifier les méthodes de modélisation orienté objet OMT, BOCH et OOSE (Object Oriented Software Engineering). UML propose un **standard de notation** pour la spécification et la conception de systèmes et de logiciels. UML est un formalisme qui décrit plusieurs modèles permettant d'aborder le système à modéliser sous plusieurs aspects: l'**aspect statique** qui décrit les objets composant le système et les relations de composition et d'héritage entre ces objets. L'**aspect dynamique** qui décrit le comportement des objets du système étudié, les états par lesquels peuvent passer ces objets, et les événements qui permettent ces changements d'états. L'**aspect fonctionnel** décrit les fonctions réalisées par les objets du système. Parmi les modèles décrits par UML nous citons:

- **Modèle de classes**: pour la description de la structure statique du système à étudier. Le modèle de classes est représenté par un diagramme de classes où on fait ressortir

les dénominations des classes, les noms d'attributs et les méthodes appropriées au domaine étudié. Le diagramme de classe représente l'aspect statique du système à l'aide de classes et de relations.

- **Modèle des cas d'utilisation** : décrit les fonctionnalités du système, les besoins des utilisateurs du futur système. Ce modèle est représenté par le diagramme des cas d'utilisation.
- **Modèle d'interaction**: décrit les interactions entre les objets du système. Il est représenté par le diagramme de séquence et le diagramme de collaboration. Le diagramme de séquence décrit les appels de méthodes par les objets (avec leur signature) dans le but de réaliser une fonction précise du système. Ici l'importance est accordée à la dimension temporelle des échanges, c'est-à-dire la représentation des échanges dans le temps. Le diagramme de collaboration représente les appels de méthodes entre objets, les liens et les interactions. Ici l'importance est accordée à la représentation spatiale des objets et des interactions.
- **Modèle de réalisation:** représente les différents scénarios et les échanges de messages entre les objets du système.
- **Modèle de déploiement:** ce modèle décrit la répartition des processus, des composants sur une plateforme matérielle.

En plus de ces modèles, UML défini la notion de méta modèle qui permet de décrire un modèle de modèles, utilisé dans les approches MDA, et faire ainsi de l'ingénierie dirigée par les modèles. Les profils UML sont des extensions d'UML pour faciliter la conception dans un domaine métier spécifique, à l'instar d'UML-S pour la spécification, la conception et la composition de services web.

- **BPML / BPEL/ BPMN:** La modélisation des flux logistiques et du workflow remonte loin dans les années 1990 sur la modélisation des entreprises. Les travaux de van der Aalst [84] sur le workflow management ont beaucoup contribués à établir une syntaxe et une sémantique assez riche pour la représentation et la planification des workflow via des gateways. Le *Workflow Management Coalition* (*WFMC*) s'inscrit dans cette même dynamique en proposant en 2005 un standard d'interopérabilité entre workflows, le *XPDL* (*XML Process Definition Language*) [85] qui sert d'outil d'échange de modèle de processus entre différents systèmes interopérant. Par opposition au *Workflow Management System*, le *Business Process Management (BPM)* a pour ambition de donner des outils de modélisation, de simulation et d'analyse post-exécution du *Business Process*, il est question de lier les processus métier ayant une forte abstraction aux architectures IT qui implémentent et exécutent ces processus. Dans cette direction, la proposition d'*OMG (Object Management Group)* d'unifier et standardiser les annotations relatives aux business process pour donner naissance en 2005 au *Business Process Model and Notations* (*BPMN)* [86] pour ce qui est des annotations, au *BPEL* (*Business Process Execution Language*) et *BPEL For Web Services (BPEL-4WS)* pour ce qui est de l'exécution des processus et services web y apportant une réelle contribution tant au niveau syntaxique qu'au niveau sémantique du PBM. Le BPML (Business Process Modeling Language) est un métalangage de modélisation de processus métiers. Le BPML a été développé par le

BPMI (Business Process Model Initiative) comme support au langage d'exécution de processus métiers, le BPEL (Business Process Execution Language). Le BPML permet de modéliser le système comme un ensemble de processus métiers et d'événements à base d'annotations BPMN (Business Process Modeling Notation), le BPML permet aussi de modéliser la synchronisation de ces processus. Le processus est vu ici comme un ensemble d'activités, une activité est vue comme un ensemble de sous activités qui peuvent être simples ou complexes.

- **DDSN: La DDSN (*Demand Driven Supply Network*)** [83 ,125] s'inscrit dans la logique Kanban et promeut l'utilisation des technologies pour partager des informations sur toute la chaîne afin de permettre à tous les fournisseurs et les opérateurs d'avoir une meilleure visibilité sur l'ensemble de la demande. Ces informations permettent de mieux s'organiser pour répondre à cette demande globale, que de répondre à des demandes locales par fournisseur. Ceci permet d'optimiser au mieux les coûts de production et de stocks à l'échelle globale et locale, évitera les surproductions et diminue les délais de livraison. En résumé, il est question de déployer dans la nouvelle supply chain une meilleure stratégie qui allie plusieurs objectifs dans le même schéma stratégique, tactique et opérationnel. Il s'agit de la baisse des coûts de production, de transport et de livraison, la réduction des délais de livraison, la réduction des stocks voire atteindre l'objectif "zéro stock" dans l'idéal. Il s'agit également de l'amélioration de la satisfaction des clients afin de les fiabiliser, la réduction de la pollution environnementale pour la durabilité de la chaîne et le climat, ainsi que l'efficacité énergétique dans la chaîne globale.

## Outils de simulation de flux

La gestion des flux se fait également par simulation. Dans ce qui suit nous présentons quelques uns de ces outils.

- **ADONIS:** est une méthode de modélisation de processus métiers crée par BOC (Business Objects Consulting). Cette méthode permet de modéliser et de simuler les processus métiers avec trois types de modèles de base: la **carte des processus**, le **modèle opérationnel de processus** et le **modèle d'environnement de travail**. Il est possible de modéliser aussi les **ressources matérielles et logicielles utilisées** par les processus et de les rattacher à une unité organisationnelle ou un rôle dans l'entreprise.

- **OSSAD:** est une méthode de modélisation de flux logistiques qui propose des modèles pour la représentation de processus. Le **modèle de procédures** détaille les processus définis dans le modèle abstrait. Le **modèle d'opération** détaille un processus en fonction des rôles joués par des utilisateurs dans le processus et du temps de réalisation des activités composant le processus. Le **modèle de rôles** représente les différents rôles internes et externes au système, ainsi que les échanges entre les différents acteurs qui jouent ces rôles. Le modèle **d'unités organisationnelles** spécifie la structure hiérarchique du système.

- **ARENA:** est un outil de simulation qui permet de représenter le modèle des flux dans une chaîne de production. Le modèle ARENA est une maquette qui représente l'ensemble des composants physiques qui interagissent pour le bon fonctionnement du système, et les opérateurs qui commandent ces machines. ARENA est basée sur une représentation graphique où le modèle se présente sous forme d'un ensemble d'objets ou icônes graphiques. Chacune de ces icônes correspond à une fonctionnalité précise (avec des options) du système réel. La représentation graphique ainsi obtenue peut être lue et comprise même par un non spécialiste du domaine. Les paramètres qui sont traités par les différentes icônes sont décrits par l'utilisateur avec son propre vocabulaire. Ainsi, des concepts comme "type de palette", "temps de fraisage" ou "nombre de caristes" peuvent être employés pour définir les caractéristiques d'un atelier. L'un des atouts majeurs d'ARENA est la notion générique d'îlot qui permet de décrire simplement des systèmes de plusieurs centaines de machines. Une seule description constitue, en quelques icônes ou objets, la modélisation d'îlots de structures similaires mais de fonctionnement ou de paramètres spécifiques. Ainsi, un atelier typique composé de files d'attente devant chaque poste de travail, chacun étant relié au suivant par un système de manutention, sera modélisé par un seul îlot, même si les temps opératoires, les temps de convoyage ou la taille des zones d'attente sont spécifiques à chaque machine. La spécification de la gamme de chaque type de pièce sur l'atelier et du planning de disponibilité de chacune des ressources s'effectue dans l'instance.

- **WITNESS:** est un outil de simulation qui fournit un environnement professionnel pour modéliser et simuler tous les processus d'activité, quelle que soit leur complexité. Que ce soit des flux de production dans une industrie manufacturière, ou des stratégies d'amélioration continue pour un investissement, la plateforme est assez flexible pour tester et vérifier des scénarios de changement sans risque. Les principales caractéristiques de WITNESS sont l'aide à la réalisation du modèle de processus, une interface graphique de bonne qualité, un outil d'aide à l'analyse des résultats, et une interface d'intégration avec les langages de programmation classiques. WITNESS est utilisé pour la simulation de système de production, la simulation de systèmes dynamiques et la vérification de circuits électroniques. Et puisqu'on parle de système de production, on parle de flux d'objets logistiques dans la chaîne de production, qui est un segment important dans la chaîne logistique globale.

Le tabelau 2 présente un récapitulatif et ne classification de l'ensemble de ces outils de modélisation. Pour chaque outil, nous présentons ses domaines d'utilisation.

**Tableau 2: Synthèse sur les outils de modélisation**

| Méthodes /Outils | UML | UML-S | SysML | BPMN /BPEL /BPML | Automates | RdP | Graphes | LOTOS/ E-LOTOS | pi-Calcul |
|---|---|---|---|---|---|---|---|---|---|
| **Description** | Unified Modeling Language | Profil UML pour la composition de service | System Modeling Language/ Profil UML pour la les systemes | Business process Model Notation | AUtomate fini déterministe ou non déterministe | Réseau de Petri | Graphes orientés, non orientés, labelisés | Language Of temporal Ordering Specification | Un modèle de calcul algébrique |
| **Formel** | Non | Non | Non | Non | Oui | Oui | Oui | Oui | Oui |
| **Modélisation de la statique du système** | Oui | Oui | Oui | Oui | Non | Non | Non | Non | Non |
| **Modèle de la dynamique du système** | Non | Non | +/- | +/- | Oui | Oui | Oui | Oui | Oui |
| **Modélisation de l'aspect Communicationnel du système** | Oui | Oui | Oui | Oui | Oui | Non | Non | Oui | Non |
| **Modélisation des règles et processus métiers** | Oui | Oui | Oui | Oui | +/- | Non | Non | Oui | Non |
| **Vérification et test du système** | Non | Non | Oui | +/- | Oui | Oui | Oui | Oui | Oui |
| **Orienté service** | Non | Oui | Oui | +/- (BPEL-WS) | Non | Non | Non | +/- (orienté process) | Non |
| **Génération de modèle et de code** | Oui | Oui | Oui | Oui | Non | Non | Non | Non | Non |

## 1.5. Conclusion

Ce chapitre a pour but de faire un tour d'horizon sur les méthodes et les outils de modélisation des flux logistiques existant dans la littérature scientifique, de dresser un panorama des travaux de recherche et développement sur l'internet des objets, et de mentionner les enjeux liés à ces nouvelles technologies.

Quelques approches ont été présentées, parmi lesquelles les approches mathématiques permettant de représenter le système par des équations mathématiques (EDP, algèbre tropicale, etc.) ou d'autres outils avec des fondements mathématiques (graphes, réseaux de Petri, etc.). L'approche systémique permettant de représenter le système à modéliser suivant plusieurs vues (vue fonctionnelle, vue statique, vue dynamique, vue organisationnelle, etc.) ; et l'ingénierie dirigée par les modèles (MDA) qui considère le système à modéliser comme étant un ensemble de modèles mise en relation, et qui prône la modélisation du système par des modèles génériques afin de séparer le domaine métier des technologies d'implémentation. Il faut noter que ces approches ne sont pas mutuellement exclusives et peuvent se chevaucher. Plus précisément, il est possible de faire une représentation de la vue statique et fonctionnelle du système par des diagrammes (UML, SART, etc.) et modéliser sa dynamique par des réseaux de Petri, ou bien intégrer un algorithme d'optimisation dans un PSM ou dans une architecture MDA. Pour chacune de ces approches, nous avons mentionné quelques outils pour la logique mathématique (graphes, automates, réseaux de Petri, algèbres de processus), et quelques outils pour l'ingénierie dirigée par les modèles (UML, SADT, BPML, etc.).

Ce qui ressort des travaux de recherche est la large utilisation des réseaux de Petri pour la modélisation de la dynamique des flux et leur coordination au sein de la chaîne logistique, l'utilisation des diagrammes UML, SADT, BPML, etc. pour la modélisation des vues fonctionnelles et statiques des systèmes de pilotages de ces flux ou la combinaison de plusieurs outils de modélisation pour des études complémentaires. Nous remarquons aussi l'utilisation d'algorithmes mathématiques pour des problèmes d'optimisation dans la gestion des flux logistiques (tournées de véhicule, plus court chemin) et de décision tant au niveau stratégique, tactique ou organisationnel.

Les travaux sur l'internet des objets mettent en évidence la possibilité d'utiliser les technologies RFID, NFC, etc. pour le suivi en temps réel et la traçabilité des objets logistiques de la matière première à la livraison au client final. Ces travaux montrent aussi qu'il est possible de coupler ces technologies à des plateformes Cloud pour la gestion coopératives des flux et le partage d'informations sur les objets échangés entre les acteurs de la chaîne.

Les enjeux liés à la modélisation des flux et l'internet des objets sont nombreux, tant sur l'aspect économique (réduction des gaspillages au niveau des stocks, optimisation des coûts de production, de transport, etc.) politique, sociale, que sur le plan réglementaire (sécurité des données personnelles, contrôle d'accès et authentification).

# Chapitre 2: Modèle statique du flux logistique

## 2.1. Introduction

La modélisation du flux logistique pose deux problèmes fondamentaux. D'une part, la modélisation des concepts et des entités qui forment le flux, leurs propriétés et leurs interrelations (ce que nous appelons "la statique du flux"). D'autre part, la modélisation de l'évolution du flux et des changements d'états des entités qui le constituent dans le temps et dans l'espace (ce qui fait référence à la dynamique du flux). Dans ce chapitre, nous appliquons la méthode de l'ingénierie dirigée par les modèles pour la modélisation de la statique du flux logistique d'une part, et nous utilisons les graphes orientés et labellisés (Directed-Labelled-Graph) pour la modélisation du flux en tant que réseau d'entités logistiques, leur interconnexions et leur interopérabilité dans l'ensemble des flux de marchandises.

La première partie du chapitre propose une extension de la spécification UML [100] par un profil UML du niveau M2 des méta-modèle MOF, pour la représentation des flux logistiques, des entités qui le constitue et de l'environnement dans lequel il évolue. Ceci en tenant compte des spécificités du cadre de nos travaux, à savoir l'intégration du flux logistique dans le Cloud à travers les technologies de l'internet des Objets pour le partage d'informations et l'interopérabilité des flux et des acteurs logistiques. La deuxième partie complète cette démarche par un modèle de graphe représentant le réseau de flux logistiques à l'instar des réseaux sociaux. Ceci permet de mieux appréhender les interconnexions entre les entités d'un même flux et les relations d'interdépendances entre les différents flux de marchandises, ainsi que les relations entre le flux et les événements qui sont générés par son environnement et qui sont sujets du changement de contexte auquel dépend fortement son cycle de vie.

## 2.2. Généralités sur l'approche MDA

### 2.2.1. Généralités et technologies liées à l'approche MDA

**MDA (Model Driven Architecture)** est née vers les années 2000, de la volonté de OMG (Object Management Group) de standardiser la conception logicielle en se centrant sur le modèle. L'idée ici est de faire une séparation entre le modèle de l'application à développer et le code source implémentant cette dernière. En découplant la logique métier du code source, MDA promet de réduire la dépendance entre l'application et les technologies qui peuvent être propriétaires et ne cessent d'évoluer, et dont les coûts de mise à jour sont chiffrés à 15% du coût total du projet initial.

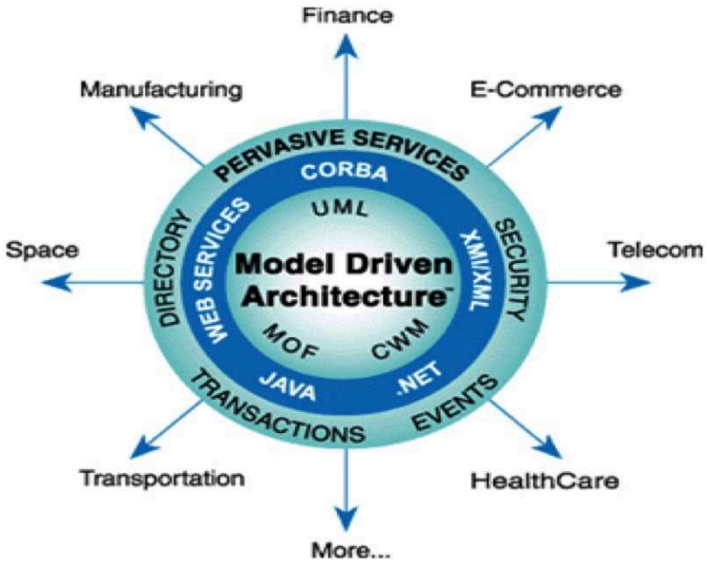


Figure 1. Architecture générale du modèle MDA

La figure 1 présente l'architecture générale du modèle MDA. Cette architecture comporte les technologies telles que **UML**, MOF, et **WWM** qui sont des standards de OMG pour la modélisation (UML), la transformation de modèles (MOF), et le datamining (**CWM**). MDA est indépendant des technologies d'implémentation (JAVA, .NET, CORBA,…) et peut être appliqué à n'importe quelle domaine métier (transport, santé, espace, télécom, finance, E-commerce).

**Modèle:** Un modèle est une représentation d'un objet, d'un concept ou d'un système permettant de faciliter ou de simplifier la compréhension de ce système. L'Ingénierie Dirigée par les Modèles (IDM) ou Model-Driven-Engineering (MDE) est une approche de modélisation de systèmes complexes dont la philosophie est de considérer le système comme étant un ensemble de modèles abstraits, liés les uns aux autres par des relations de transformation. C'est dans cet ordre d'idée que OMG (Object Management Group) préconise en 2000 l'utilisation des méta-modèles dans tout le cycle de développement d'applications, en proposant la démarche MDA (Model Driven Architecture).

**MOF**: Le MOF défini trois méta-modèles pour la représentation des systèmes, le PIM, le PSM et le modèle du code relié par des étapes de transformation successives comme illustré dans le schéma de la figure 2. MDA est fondée sur la nécessité de définir des modèles de systèmes productifs, flexibles, modulables, interchangeables, et pérennes. Ainsi, MDA propose un formalisme de modélisation dont l'architecture repose sur quatre niveaux et est basé sur le MOF (Meta Object Facility). La démarche MDA est structurée selon quatre niveaux: M0, M1, M2 et M3. Le niveau M0 est le domaine du monde réel; le niveau M1 est celui du modèle, de la représentation des objets et des concepts du monde réel (du niveau M0). Au niveau M1 nous retrouvons les modèles UML. Le niveau M2 est celui des méta-modèles. Un méta-modèle permet de définir la spécification d'un ensemble de modèles, et un modèle doit être conforme à sa spécification selon le MOF, c'est à dire à son méta-modèle. Au niveau M2 nous pouvons citer la spécification UML ou le méta-modèle UML, les profils UML tel que SysML pour la modélisation des systèmes, SOAML pour la

modélisation des systèmes orientés services, Uml Profile for DoDAF/MODAF (UPDM) pour la spécification des architectures de systèmes du domaine de la défense, MARTE (Modeling and Analysis of Real-time Embedded systems) pour la spécification des systèmes temps réel et d'autres langages de modélisation tel que BPML, BPEL. Le niveau M3 est celui du méta-méta-modèle, c'est à dire le MOF lui-même. Le MOF définit le méta-méta-modèle comme un modèle récursif, auto-descriptif c'est à dire qui est sa propre description, qui peut s'auto-générer. Il faut noter que UML n'est pas imposé par OMG comme formalisme, mais plutôt recommandé en tant que outil le mieux adapté pour la modélisation des systèmes d'information.

### 2.2.2. Architecture d'un modèle MDA

Un modèle dans MDA est architecturé suivant deux couches, le PIM (Platform Independent Model) et un ou plusieurs PSM (Platform Specific Model).
Le PIM est le point de départ de la modélisation, il spécifie les fonctionnalités de l'application et ses contraintes. Le PIM peut être réalisé à l'aide de diagramme de classe UML pour représenter les processus métiers et les règles d'organisation. Ce modèle est indépendant de toutes technologies (JAVA, .Net, CORBA, etc.) et peut être instancié par un PSM pour une plateforme technologique donnée en se basant sur les spécificités de la plateforme. Le PSM représente le modèle du code de l'application et est généré automatiquement à l'aide des règles de transformation de modèles. Une fois le PSM obtenu, il peut être utilisé pour générer le code implémentant l'application.

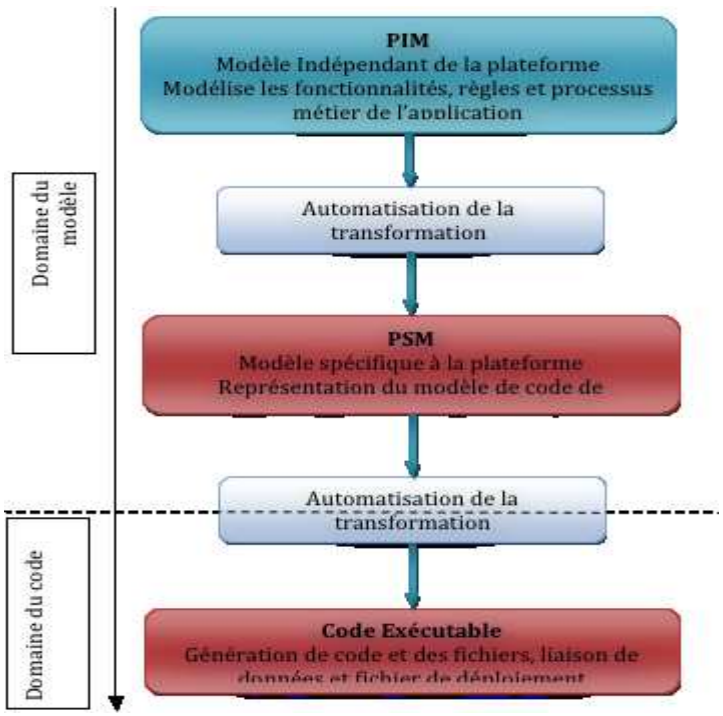


Figure 2. Architecture des modèles dans MDA

La figure 3 représente les différents niveaux que nous venons de citer et précise le passage d'un modèle à un autre.

### 2.2.3. La méta-modélisation

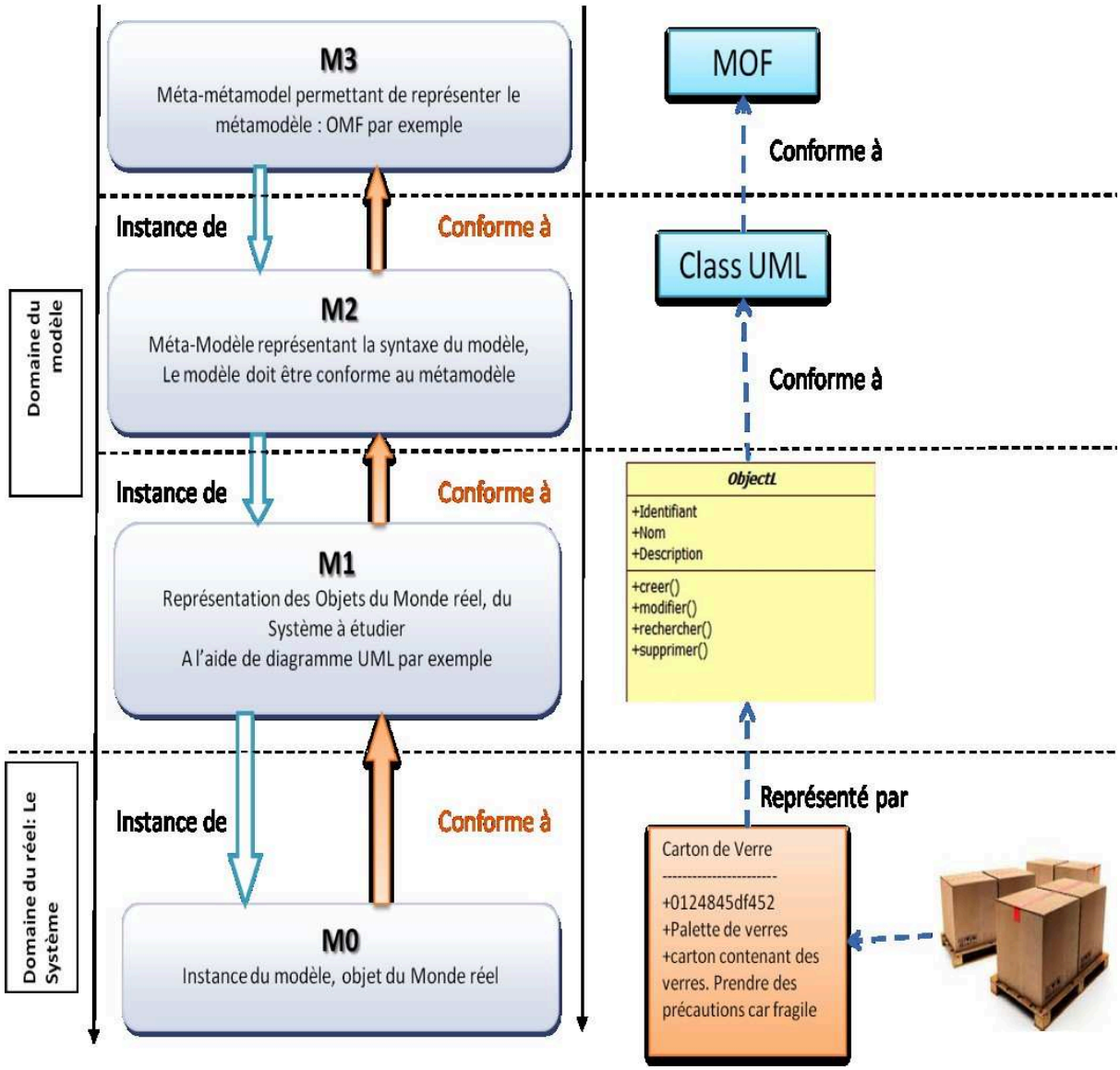


Figure 3. Les différents niveaux de méta-modélisation dans MDA

Selon OMG, le méta-modèle est un modèle spécial qui définit la syntaxe abstraite d'un langage de modélisation. Ici, il s'agit de la représentation des classes de tous les modèles exprimés dans ce langage. OMG exprime les méta-modèles MDA en utilisant MOF (Méta Object Facility) [101]. Selon d'autres sources (Melor, Scott, Clark, Sammut et Williams), le méta-modèle est un modèle du langage de modélisation, ceci inclut la représentation des concepts du langage, sa syntaxe graphique et sa sémantique : c'est-à-dire la signification des modèles et programmes écrits dans ce langage. La figure 3 représente les quatre niveaux d'abstraction du méta-modèle dans une démarche MDA. Le niveau M0 représente le domaine métier, le système à étudier. A ce niveau nous trouvons les objets du monde réel (palettes, documents, articles, véhicules, personnes, organisations, etc.).

Figure 4. Objet du monde réel

Au niveau M1, nous avons des modèles qui représentent un aspect de ce système, qui représente ces objets du monde réel (figure 4). Il s'agit ici des diagrammes de classes UML par exemple.

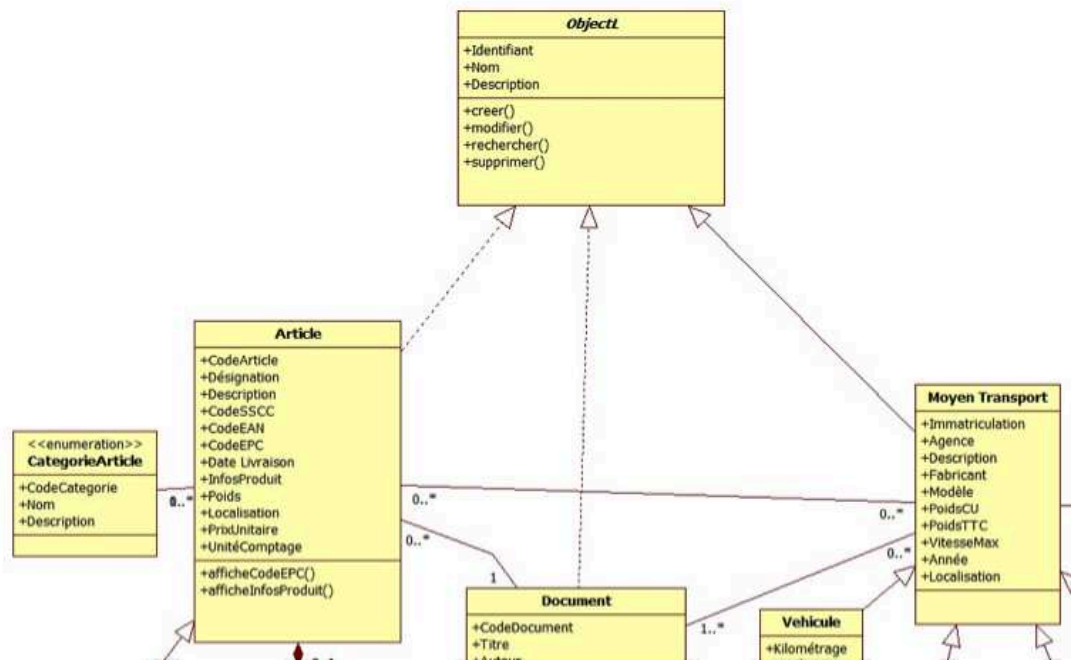


Figure 5. Exemple de diagramme UML

Le modèle peut être écrit par des classes UML (figure 5). Il doit donc être conforme à la spécification UML en tant que langage de modélisation. Les classes UML forment donc un méta-modèle qui permet de spécifier les diagrammes de classes, qui en sont une instance. On parle de méta-modèle UML, appartenant au niveau **M2**. Le méta-modèle UML doit à son tour être spécifié par un modèle, le MOF qui s'auto définit. Au niveau **M3** nous avons donc les méta-méta-modèles qui permettent de définir les méta-modèles qui serviront de décrire les modèles.

### 2.2.4. Les technologies liées à MDA

L'apport des diagrammes UML pour l'approche MDA réside dans l'aspect graphique et l'aspect orienté objet. UML 2.0 est la version actuelle d'UML, divisée en trois grandes parties : UML 2.0 Infrastructure, UML 2.0 OCL, et UML 2.0 Diagram Interchange. Ce langage a été spécifié pour palier aux exigences de l'approche MDA. Il est à noter que l'utilisation d'UML n'est qu'une recommandation. Les profiles UML sont une extension d'UML à un domaine d'application précis. Le MOF (Meta-Object Facility) est un langage d'expression de modèles utilisé pour décrire les modèles MDA. Le MOF assure la neutralité du modèle en définissant :

- Les méta-objets du MOF,
- Les relations entre les différents objets du méta-modèle,
- Les paquetages rendant les classes des modèles modulaires en classe et associations
- Un ensemble de règles pour exprimer le méta-modèle à l'aide d'interface IDL (Interface Definition Language)

OCL (Object Constraint language) est un langage qui permet d'exprimer des contraintes sur les modèles et les méta-modèles UML. OCL [102] est utilisé dans la démarche MDA pour faire aussi de la vérification des modèles et méta-modèles (model checking). Il existe d'autres outils qui facilitent la mise en place de modèle et leur transformation dans une démarche MDA. Parmi lesquels QVT [103] pour la transformation de modèle, les plugins Eclipse comme EMF (Eclipse Modeling Framework) et GMF (Graphical Modeling Framework), Equinox, Syrius, qui sont des environnements de travail qui facilitent l'écriture de code de transformation, l'importation et l'export de modèles au format divers.

## 2.3. Application de la démarche MDA à la mise en place d'un DSML pour le flux logistiques

Dans ce paragraphe, nous proposons de modéliser la statique du flux logistique par la mise en place d'un langage spécifique au domaine du workflow (DSML), en suivant la démarche MDA. Pour ce faire, nous commençons par représenter les différents concepts du workflow par un méta-modèle qui sert de PIM, le modèle de flux indépendant de toutes plateforme d'implémentation. Le PIM que nous proposons est subdivisé en plusieurs packages afin de résorber toutes les préoccupations liées à la gestion du flux logistique dans un contexte de partage d'information pour une meilleure interopérabilité des flux et des acteurs logistiques. Cette subdivision permet également l'intégration du flux dans le Cloud et les architectures de l'Internet des objets.

Le PIM est ensuite enrichi par un profil UML qui étend la spécification d'UML 2.0 en vue d'une meilleure personnalisation; ceci afin de mieux répondre à nos attentes. Nous avons rajouté des règles de validation OCL sur le profil dans le but de vérifier les contraintes liés au modèle et sa consistance avant une éventuelle transformation.

Afin d'intégrer les spécificités liées aux différentes plateformes de déploiement Cloud et l'architecture de l'IoT, le PIM ainsi construit a été ensuite décliné en plusieurs modèles spécifiques aux plateformes de déploiement (PSM). Pour ce faire, nous proposons un PSM pour la plateforme Cloud GoogleAppEngine de Google, un PSM pour FIWare, de même que Hadoop et Firebase ; et enfin un PSM pour l'intégration du flux dans les architectures de l'IoT ainsi que le Cloud des capteurs (Sensor Cloud)

### 2.3.1. Un méta-modèle pour la modélisation d'un PIM du flux logistique

Comme illustré par la figure 6, nous organisons le PIM en sept packages contenus dans un seul package (COM_SLOT_Meta_Models) par des relations de *containment*. Le premier package est celui qui gère les entités logistiques dans la globalité (*Business_Entity_Management*). L'Item étant une de nos préoccupations principales, nous dédions la gestion de l'ensemble de ses concepts au package *Items_management*. La gestion des acteurs logistiques ainsi que les organisations auxquelles ils sont affiliés est du ressort du package *Organization_Management.* Le contexte de l'entité business et les événements déclenchés par son environnement sont modélisés dans le package *Context-Event_Management.* Le Workflow quant à lui est subdivisé en deux packages, les ressources engagées dans le processus *(Ressources)* et le cycle de vie du processus lui-même *(WorkflowProcess_Lifecycle).* Un dernier package s'occupe de la gestion des contraintes et des règles métier liées au processus *(Business_Rules_Management).*

**Diagramme de package du PIM**

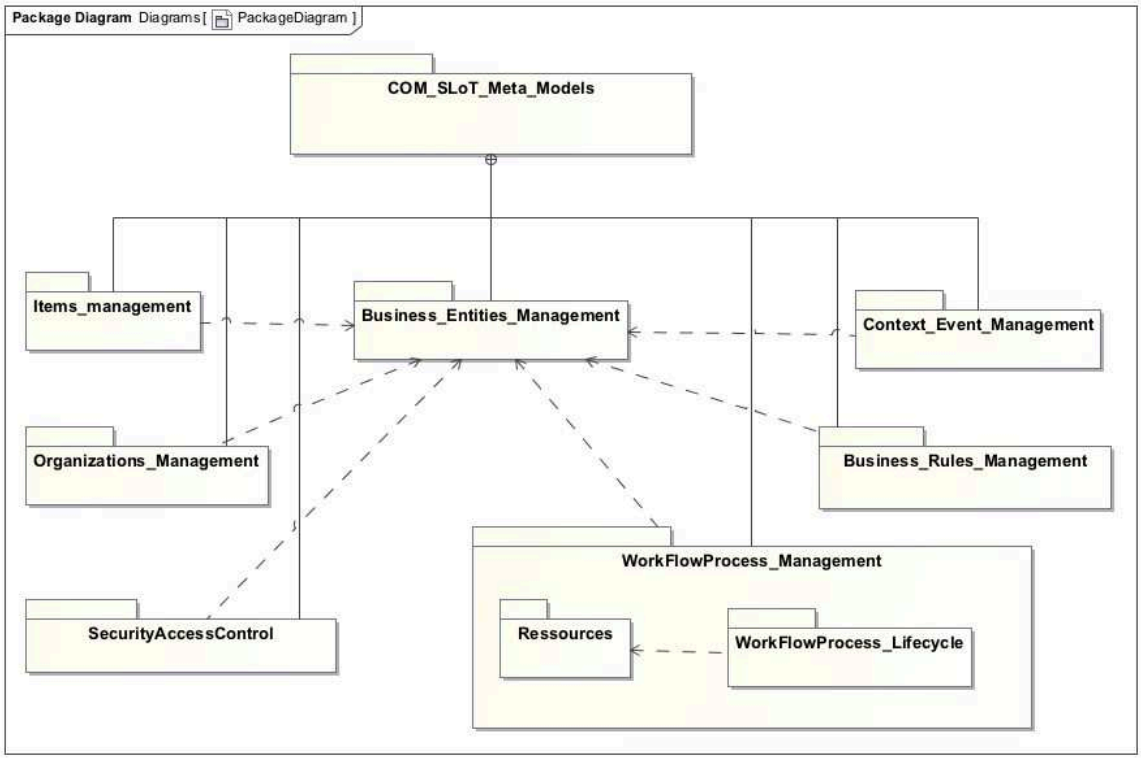


Figure 6. Diagramme de Packages et leurs interdépendances

**Modélisation des entités logistiques**

Ce package précise la spécification des entités logistiques ainsi que les propriétés qui permettent leurs descriptions (figure 7). Une entité logistique (Entity) est décrite par ses caractéristiques (Property). Les entités logistiques peuvent être agrégées pour former des unités logistiques plus grandes (container, camion, cargo). Dans ce cas, il est intéressant de retrouver les entités contenues dans l'entité agrégante (*inner_entities).* Les caractéristiques d'une entité logistique peuvent être décrites à leur tour par ajout d'informations complémentaires *(Property_metadata)* via l'association *hasMetaData.*

Les relations inter-entités sont multiples : des relations de contenu-contenant, des relations de composé-composant, des relations d'usage lorsqu'une ressource utilise une autre ressource, les relations d'affiliation comme celle entre un acteur logistique et son organisation ou son équipe. D'autres types de relations existent comme celle qui lie les éléments d'un même flux. Sur une entité, on peut réaliser des opérations qui peuvent avoir des paramètres ou pas. Ces opérations peuvent concerner la gestion des données de l'entité comme la création d'une instance de l'entité, la mise à jour de ses caractéristiques, sa suppression; ou des opérations liées aux tâches du workflow process (chargement / déchargement de palettes, emballage des Items, étiquetage des cartons, scan du code RFID, etc.) que nous aborderons dans le cycle de vie du processus.

La clusterisation des entités logistiques permet d'optimiser leur gestion par région, selon l'organisation propriétaire du flux, ou selon que ces entités appartiennent à un même flux de marchandises. Ainsi, dans la plateforme de partage on peut optimiser la recherche d'entités ayant les mêmes caractéristiques dans un même cluster région (ville, pays, continent, ou combinaison de ces trois), ou des entités satisfaisant certaines propriétés dans une même organisation logistique, ou se trouvant dans un même workflow.

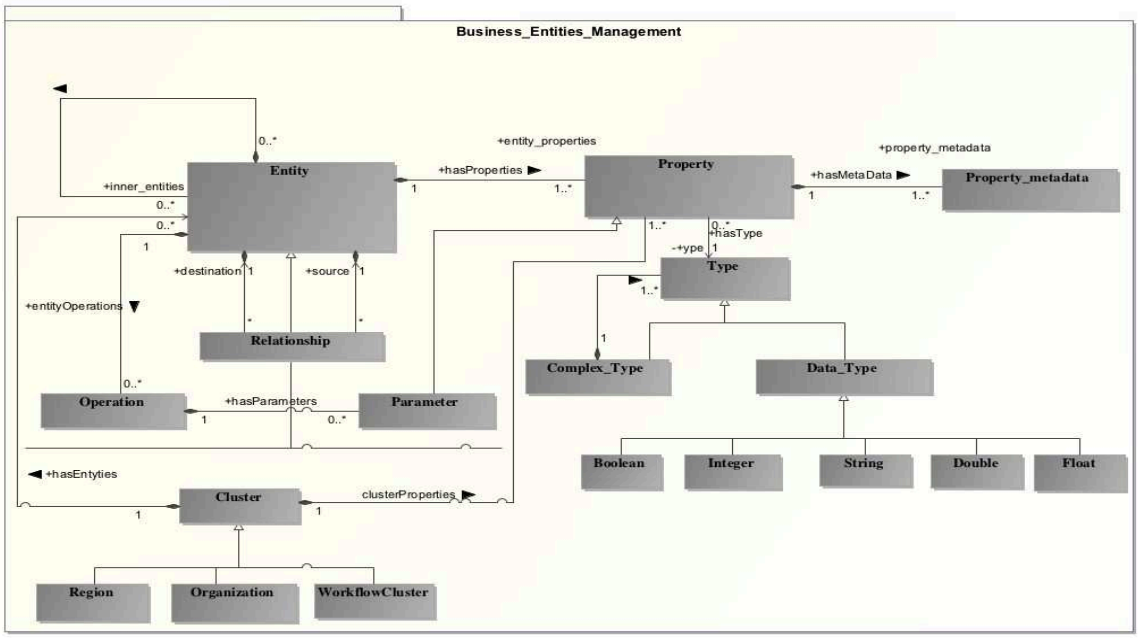


Figure 7. Méta-modèle des entités logistiques et leurs dépendances

**Modélisation des marchandises (Items)**

Un Item est une spécialisation de l'entité (Entity). Un Item désigne de manière générique une marchandise, un livrable, un article au sens de la commande dans le processus de livraison. Comme illustré par la figure 8, les Items peuvent être agrégés pour former des ensembles (Item_Assembly) à l'instar des batch et des lots. Un article peut avoir des versions, dans ce cas les versions d'un même Item auront des relations comme celle de précédence entre les versions. Les instances d'un même Item peuvent avoir des relations quantifiées ou non. Une instance est en fait la concrétisation physique de l'Item.

Une instance possède des propriétés physiques liées au monde réel comme son code électronique (EPC - Electronic Product Code), qui est unique. Les instances d'un Item peuvent s'inscrire dans le cadre d'un ou de plusieurs processus (manufacturing, delivering, custom_packaging, transport, ...) où ils subissent des transformations successives pour donner naissance à de nouveaux Items, ou tout simplement verront leurs caractéristiques modifiées (position, forme, quantité, état de la matière). Nous reviendrons sur ce point dans le package dédié au cycle de vie du processus. Le suivi de l'Item se fait en suivant l'ensemble des positions qu'il a occupé pendant son cycle de vie. À chacune de ces positions, l'Item peut subir des opérations qui peuvent modifier sa structure interne ou changer ses informations contextuelles.

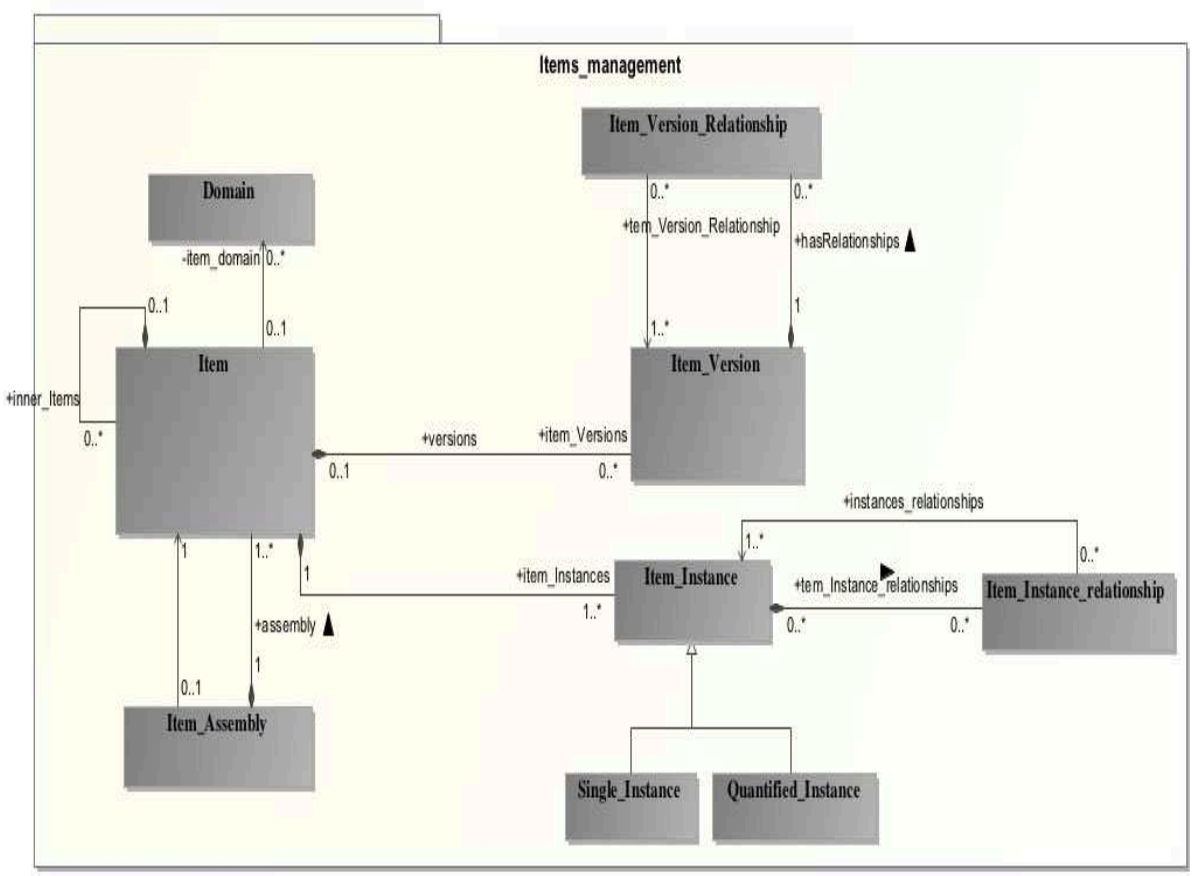


Figure 8. Diagramme du méta-modèle des Items

**Gestion des organisations et des acteurs logistiques**

Une organisation est une entreprise, une firme ou un opérateur 3PL/4PL qui participe au flux de façon active. Elle agit sur les Items du flux ou met à disposition du flux des ressources ou moyens nécessaires à son accomplissement, ou participe de façon passive en consultant par exemple des informations liées au flux (figure 9). Un acteur logistique (*Business_Actor*), ou agent, joue un rôle (*Team_Role*) dans une organisation à travers son équipe (*Team*). Ce rôle peut être différent du rôle qu'il joue dans le cadre du flux de marchandise, dans lequel il peut se voir attribuer plusieurs rôles (*WorkFlow_Role*) à des périodes différentes. Les acteurs logistiques peuvent être liés les uns aux autres par des relations d'interdépendance multiples.

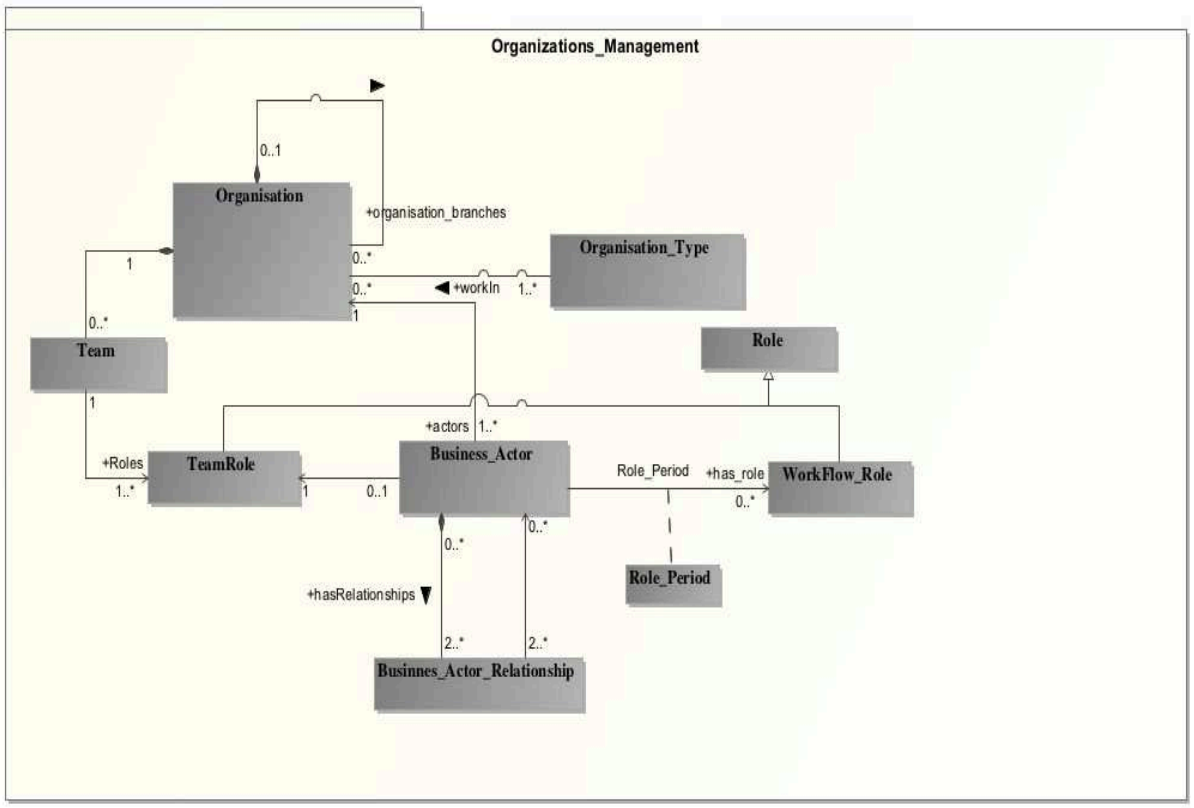


Figure 9. Méta-modèle des organisations logistiques et des acteurs impliqués dans le flux

**Méta-modèle des Workflow Process et leurs interrelations**

Nous subdivisons ce package en deux sous packages. Le premier décrit le cycle de vie du processus, et le second modélise les ressources engagées dans le processus pour qu'il réalise l'ensemble de ses traitements sur les Items. Le cycle de vie d'un processus (*WorkFlow_Process*) décrit l'ensemble des étapes (*Stage*) par lesquelles le processus évolue, de sa création jusqu'à sa terminaison comme illustré par la figure 10. Nous utilisons la terminologie du framework Guard-Stage-Framework (GSM), qui stipule qu'un processus est la donnée d'un ou de plusieurs stages. Les stages à leur tour étant des ensembles de

tâches dont l'exécution est spécifiée par le stage. Une tâche est indivisible car considérée comme une action atomique, du point de vue du Workflow process.

Selon SPARK, un Workflow n'a de raison d'être que l'accomplissement d'un ou de plusieurs objectifs business (Goals). Ces objectifs peuvent être déclinés en sous objectifs ou en objectifs à plus petite échelle des stages qui constituent le workflow process (milestones). Pour atteindre ces objectifs le Workflow consomme des ressources ou se voit affecter des moyens pour son accomplissement. Ces ressources constituent les entrées du processus du workflow, en tant que système.

Un processus peut être subdivisé en sous processus (*sub_processes*), selon sa taille. Les relations d'interdépendances entre processus sont diverses et variées, les relations d'usage (uses) qui dénote qu'un workflow utilise un autre workflow pour atteindre son objectif, les relations de convergences (*FlowComposition*) ou de divergence de flux (*FlowDesagregation*), les relations de chaînage de flux (*WorkFlowChain*), de complémentarité dans le cadre du parallélisme de flux (*ParrallelWorkFlow*).

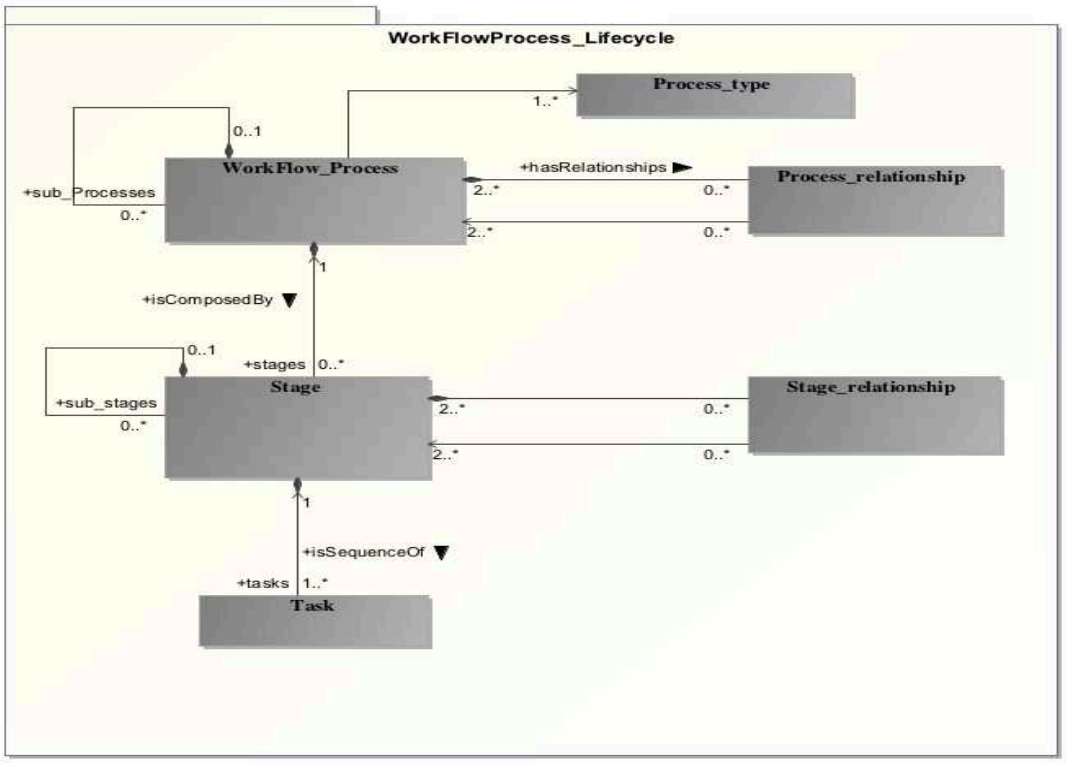


Figure 10. Méta-modèle du cycle de vie du workflow process

Un processus ne peut fonctionner tout seul, il lui faut des ressources. Une ressource désigne tout objet matériel, ou toute personne ou équipe qui participe au bon déroulement du Workflow. Dans un processus de livraison, une équipe peut par exemple être chargé par le chargement des palettes, une autre équipe par le transport et une dernière équipe par la distribution de ces palettes. Les camions qui servent de moyen de transport sont aussi des ressources, sans oublier les emballages nécessaires au processus ainsi que les infrastructures (route, pont, voies ferrées, fluviales, Hub logistiques, plateformes, etc.) qui ont servi pour l'acheminement de la marchandise. Tout ceci représente l'ensemble des ressources que le flux met à sa disposition pour accomplir ses objectifs. Une ressource peut avoir plusieurs

instances (*Resource_Instance*), voir la figure 11. Les instances de ressources sont liées entre elles par des relations qui peuvent prendre plusieurs formes. Il s'agit des relations d'inclusion (*Include_In*) qui dénote qu'une ressource est incluse dans une autre, des relations d'usage (*uses*) qui spécifie qu'une ressource utilise une autre ressource (un camion emprunte une route) dans ce cas on précise la quantité utilisée par exemple. De même que les instances, les ressources sont elles aussi reliées par les même types de relations, en plus des relations qui les lient au flux dans lequel elles participent.
Faire le suivi d'une ressource revient à déterminer les différentes positions géographiques occupées par ses instances au fur et à mesure de l'évolution du workflow. De ce fait, nous définissons la meta-class *Location* pour désigner les différentes coordonnées GPS occupées par l'instance ou les différentes adresses par lesquelles l'Item est passé tout au long de son cycle de vie.

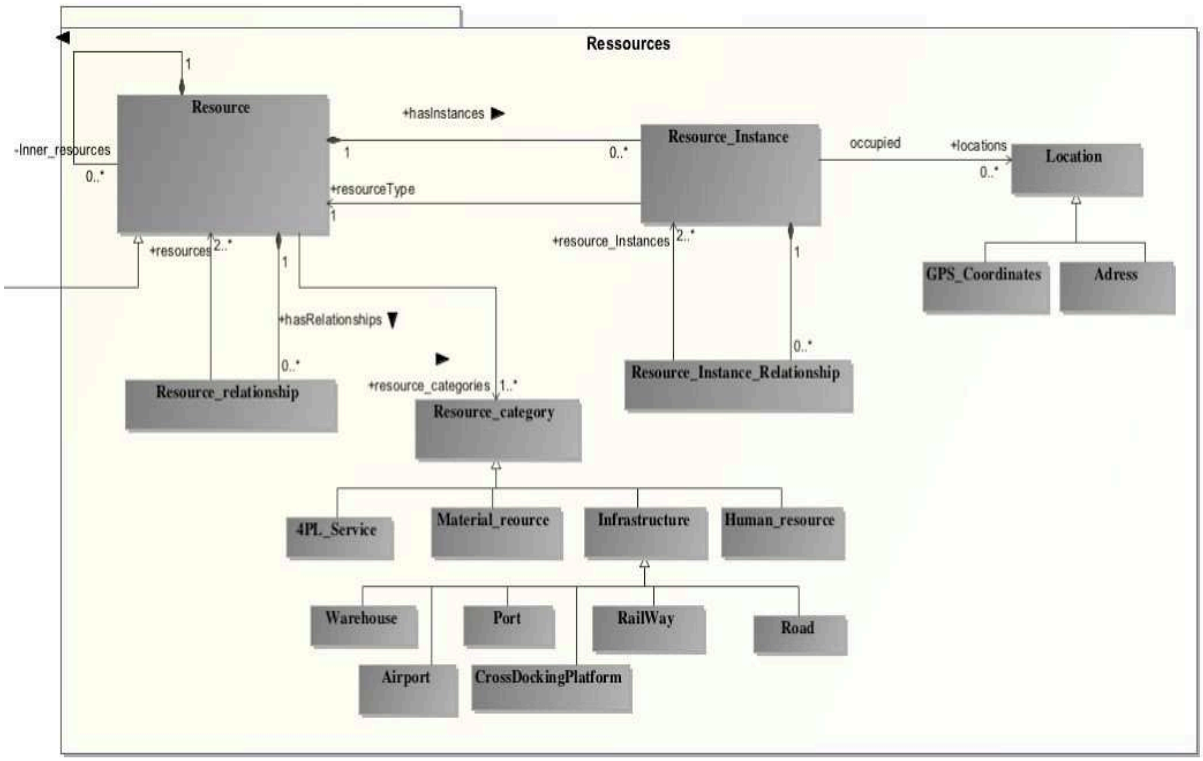


Figure 11. Méta-modèles des ressources engagées dans le processus

**Modélisation des contraintes et des règles métier**

La spécification SBVR (Semantics of Business Vocabulary and Business Rules) [101] de OMG (Object Management Group) est assez formelle et riche pour la définition de la sémantique des règles métier. Ceci au niveau organisationnel, opérationnel ou au niveau de la sécurité, ou des règles de régulation du business process (figure 12). L'un des avantages de SBVR est sa compatibilité avec la démarche MDA en étant partie intégrante. Ce package reprend une toute petite partie de cette spécification pour ce qui est de la classification des règles de gestion du business process en général, et du Workflow en particulier. Nous faisons ensuite une extension pour ajouter des concepts qui nous sont utiles, en nous focalisant sur une catégorie particulière de règles métier que sont les règles de réaction

(*Reaction_Rules*). Selon la spécification SBVR, les reaction_Rule sont une sous catégorie de règle de contrôle de flux. Nous faisons une extension de cette catégorie de règle pour y rajouter des concepts propres au traitement d'événements complexes. Ces événements sont issus de la philosophie CEP (Complex-Event-Processing) qui consiste à déclencher des actions dès lors qu'un événement ou un ensemble d'événements (event-pattern) survient, et que certaines conditions sont satisfaites. Ainsi, une règle sur des événements (*Event_Rule*) est un ensemble d'événements (*Event*) qui déclenche une ou plusieurs actions (*Action*) si certaines conditions sont satisfaites (*Condition*). Cette façon de modéliser nous permet de rester conforme au paradigme ECA (Event Condition Action) largement recommandé et utilisé pour la coordination et le suivi des workflows. Elle s'inscrit également dans la logique de programmation déclarative (Declarative Programming) avec tous les avantages qui en découlent. Pour faciliter l'écriture de ces règles, nous associons à chacune d'elle la possibilité de définir la formule (booléenne ou non) qui permet de calculer sa satisfiabilité. Une formule booléenne est donc une fonction logique à plusieurs variables qui retourne *true* ou *false* selon les valeurs de ses variables. Elle permet ainsi à la condition d'être satisfaite ou pas selon les circonstances (valeurs de ses paramètres). Plusieurs formules peuvent être associées à la même condition. Dans ce travail, nous ne rentrons pas dans les détails de combinaison de ces formules. Ceci est du ressort du calcul de fonctions booléennes à plusieurs variables, donc certaines variables peuvent être aussi des fonctions booléennes, ce qui s'éloigne du cadre de cette étude.

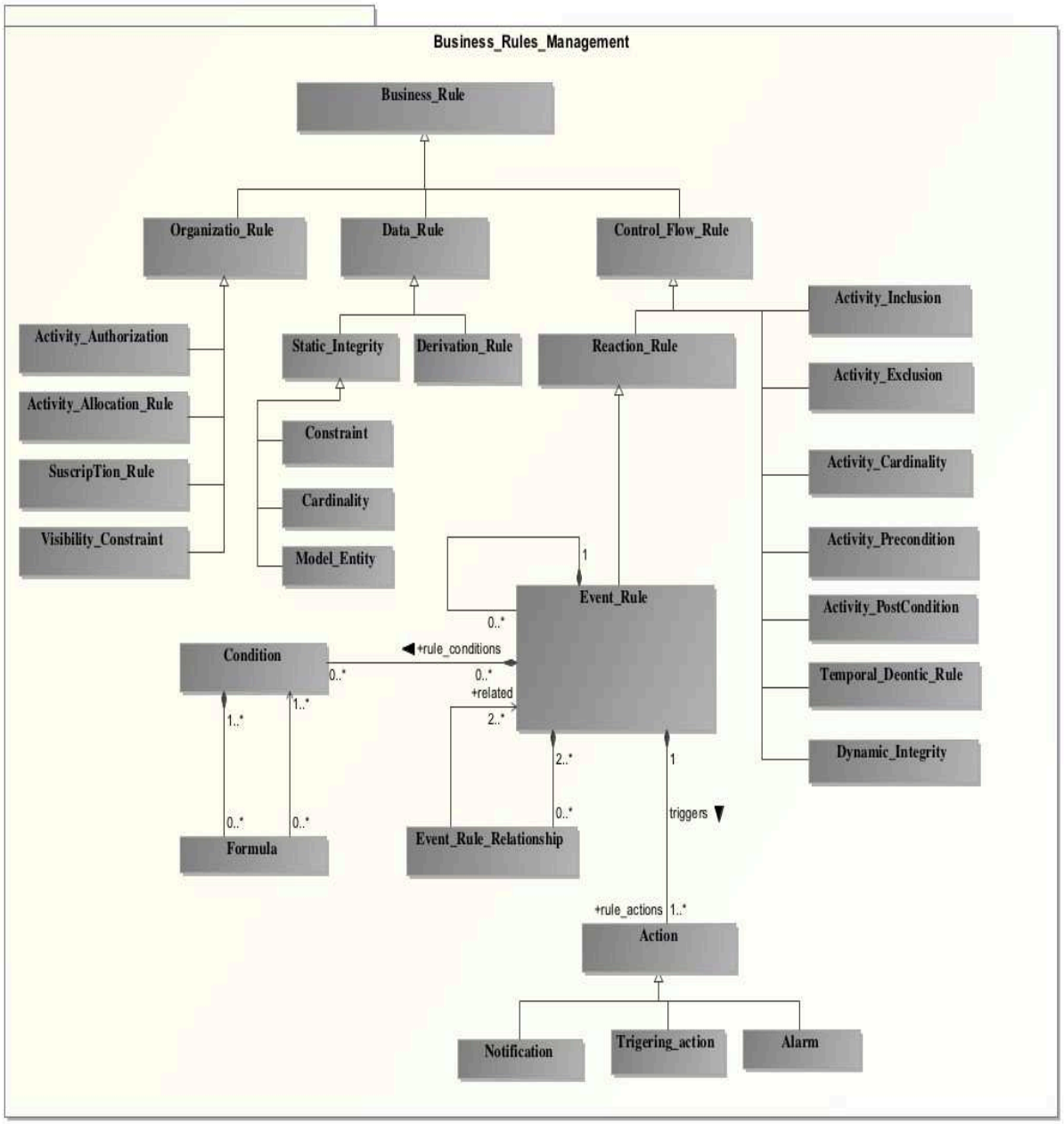


Figure 12. Méta-modèle des contraintes et des règles métier

**Gestion des événements liés au Workflow**

Le flux est vu comme un système à événements discrets, c'est à dire qu'il émet et reçoit en permanence des événements qui proviennent de son environnement. L'ordre d'occurrences de ces événements n'est pas préétablie, et ne suit aucune fonction prédéfinie. Ces événements peuvent perturber le contexte du flux, soit en modifiant la valeurs des variables de contexte de l'Item ou des Items qui le constituent. Le Workflow est de ce fait assujettie à changer d'état en fonction des événements internes au flux (*Internal_Event*) ou externe au flux (*External_Event*). Ceci inclut le passage d'une opération (ou tâche) à la prochaine opération dans le cas d'un chaînage de processus (*process chaining*), ou la notification d'un processus à d'autres processus de l'occurrence d'un événement dans le cadre d'une synchronisation parallèle via des événements. Les sources d'événements sont diverses et variées, ceci est lié à la diversité des opérateurs ou acteurs logistiques. Nous pouvons citer les opérateur de manutention (caristes, conducteurs de camions, ...), les services logistiques des entreprise partenaires du flux (service d'achat-vente, service de commande, de transport, ... ), les services informatiques (backend des plateformes Cloud ou des systèmes propriétaires) et des capteurs qui remonte aussi des données pour le suivi et la géolocalisation du flux.

Nous définissons le contexte du flux comme l'ensemble des variables qui changent au fur et à mesure que le workflow évolue. Il s'agit de la position du flux (*Location*) ou de l'Item qui le constitue selon le niveau de granularité, de l'étape business (task/Stage/process) dans lequel se trouve l'Item (*Bstep*), du propriétaire du flux (*Owner*) au moment où il se trouve dans cette position, et surtout de la date (*Time*) à laquelle l'Item se retrouve dans cette étape du workflow. D'autres variables de contexte peuvent être définies selon le flux étudié. Des opérateurs logistiques qui collaborent à la coordination du flux peuvent donc souscrire au contexte d'un Item ou d'un Workflow afin d'être notifié des événements qui surviennent dans ce contexte, via une interface *PUB_SUB* de souscription et de publication d'événements. De même, les sources d'événements peuvent publier des événements pour mettre à jour les variables du contexte d'un Business entity (Item). Le méta-modèle des événements liés au contexte du workflow process est illustré par la figure 13.

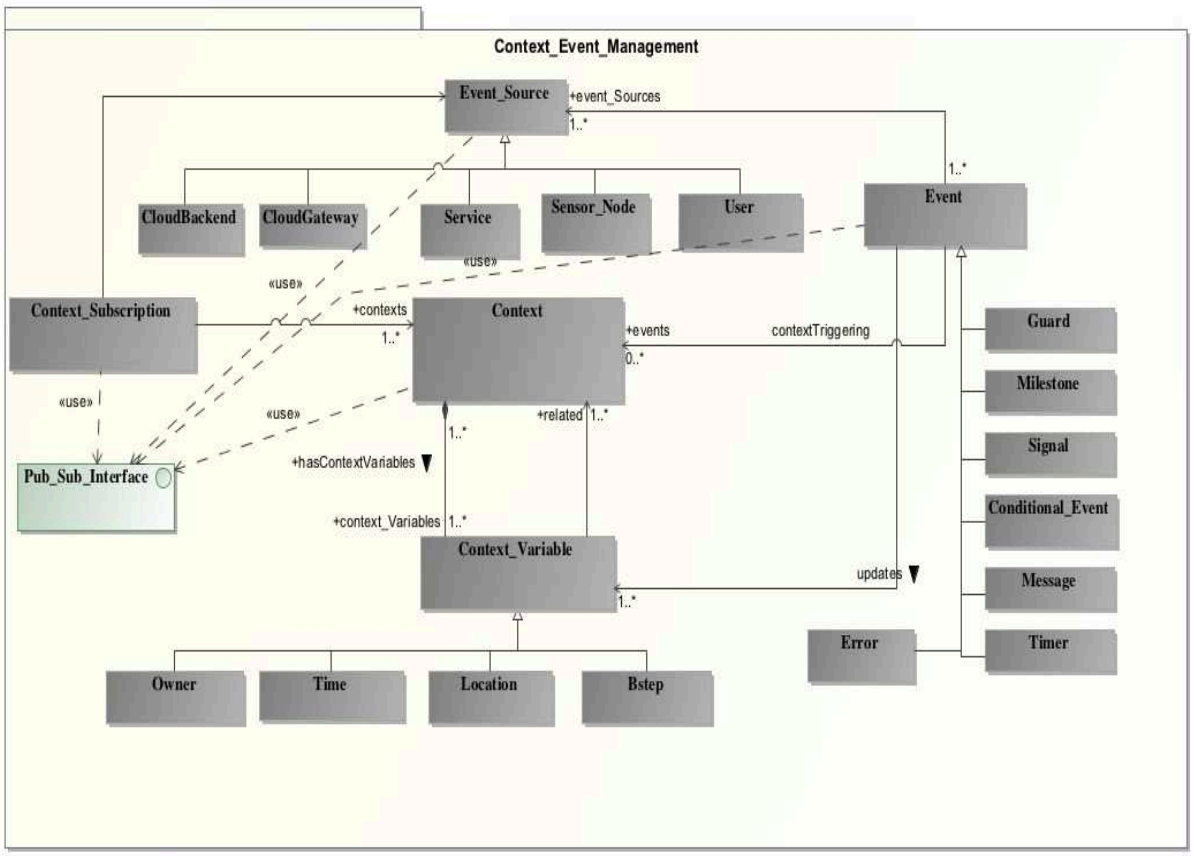


Figure 13. Méta-modèle des événements liés au contexte du workflow process

### 2.3.2. Réalisation d'un profil UML pour le flux logistique

Il existe trois manières de construire un méta-modèle. Soit par définition d'un nouveau modèle à partir de concepts complètement nouveaux héritant de rien, soit par modification d'un méta-modèle existant par ajout, suppression ou modification de ses éléments et des contraintes afférentes, soit par spécialisation d'un méta-modèle existant par ajout d'éléments et de contraintes. Les profils UML s'inscrivent dans la troisième catégorie. Un profil UML est une spécialisation du méta-modèle UML par ajout de nouveaux concepts, de nouvelles relations et de nouvelles contraintes sur des concepts existant du méta-modèle UML. Un profil UML est une extension du méta-modèle UML afin de l'adapter à un domaine métier, à une problématique bien déterminée et particulière. Concrètement, Un *méta-modèle* est un ensemble de *méta-classes* reliées entre elles par des *méta-relations*. Une instance de méta-modèle est un *méta-objet* constitué des instances de méta-classes. Une méta-classe a des *méta-attributs* et des *méta-opérations*. Le méta-attribut est une propriété des éléments du modèle. La méta-opération quant à elle représente un traitement applicable à un élément du modèle. Un méta-attribut tout comme les paramètres d'une méta-opération peuvent être typés par des types de données (*dataType*). La méta-association est une relation binaire entre deux méta-classes, portant un nom et si possible un nom de rôle et une multiplicité à chacune de ses extrémités. Le méta-package est un espace de nommage permettant de regrouper plusieurs méta-classes, méta-associations et d'autres éléments du méta-modèle.

Après avoir présenté les concepts du méta-modèle et les différents packages qui le constituent, nous réalisons, dans ce qui suit, un profil UML pour la gestion des flux logistiques. Le profil que nous présentons est constitué de plusieurs méta-modèles et subdivisés en méta-packages. Chacun représente un aspect spécifique de la modélisation des flux de marchandises.

**Méthode de réalisation et les différentes étapes de mise en oeuvre du DSML**

Cette méthode de réalisation du profil UML est inspirée de la démarche proposée par Darius Silingas [104]. Elle est axée sur quatre grandes étapes: les requirements, le design, l'implémentation et les tests. Chaque étape est subdivisée en sous- tâches comme l'indique le schéma de la figure 14.

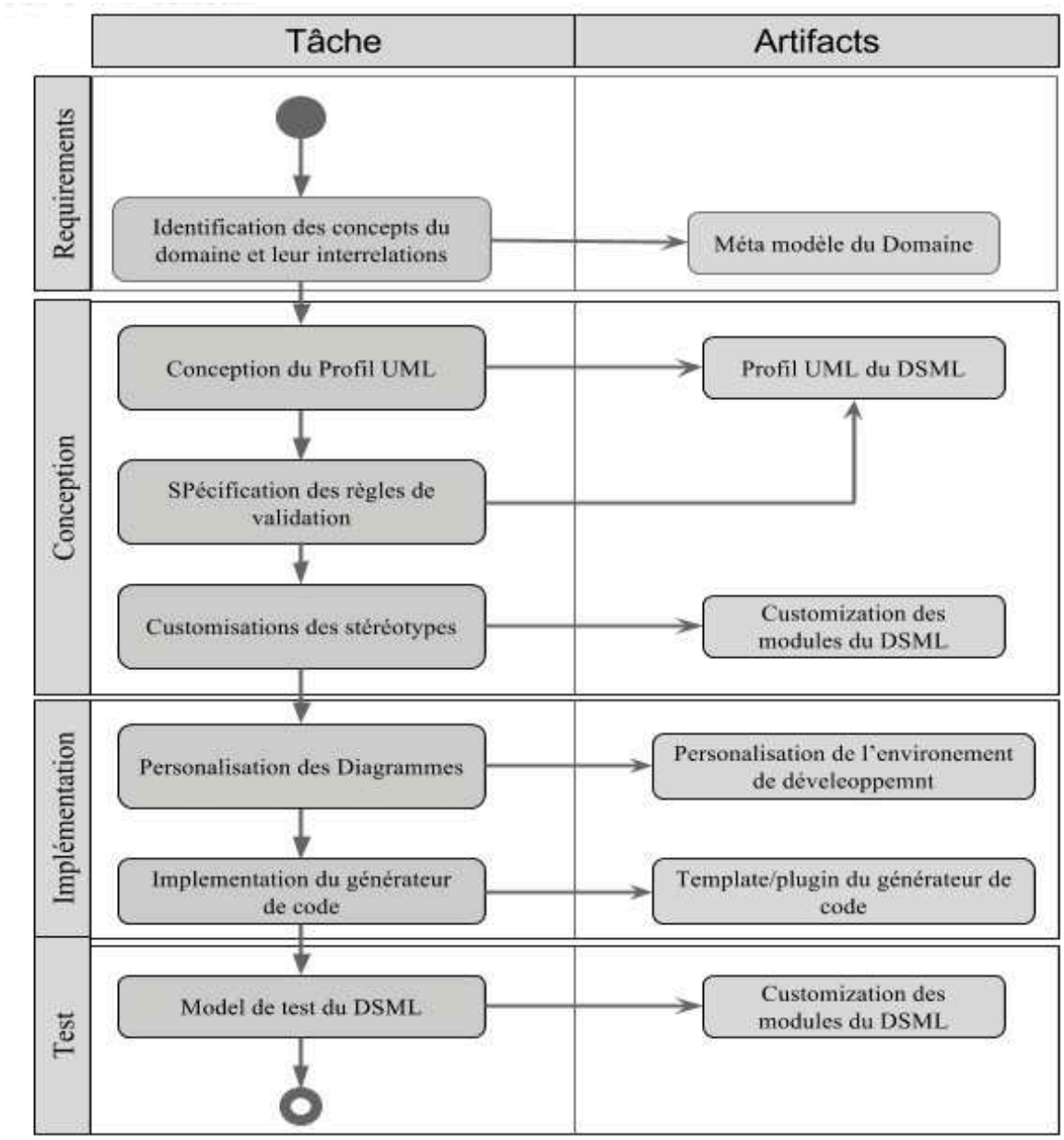


Figure 14. Les étapes de mise en oeuvre du profil UML du DSML

L'étape d'identification des concepts du domaine a été faite dans la phase de mise en oeuvre du méta-modèle. Ce qui suit est la conception du profil UML, la spécification des règles de validation et la customisation des stéréotypes. Nous proposons aussi par la suite des diagrammes personnalisés pour le *DSML* (*Domain-Specific Modeling Language*) ainsi construit. L'implémentation du générateur de code sera traitée dans le paragraphe dédié à la transformation du PSM vers le code.

**Profil UML des entités logistiques**

Le profil UML de la figure 15 étend le méta-modèle présenté précédemment pour la gestion des entités logistiques. Nous avons transformé les méta-classes en stéréotype UML et ajouté des méta-attributs afin de customiser le diagramme de profil. Un deuxième niveau de personnalisation est le rajout des icônes correspondant à chaque concept du domaine étudié qui est celui du flux logistique. L'entité a un nom et un identifiant, et caractérisé par des propriétés qui ont des noms et des valeurs. La valeur d'une propriété a un type qui peut être primitif (*Data_Type*) ou complexe (*Complex_Type*), c'est à dire une agrégation de type primitif. Les propriétés de l'entité sont des caractéristiques telles que le code électronique EPC, le volume, le poids. Ces caractéristiques peuvent avoir des méta-données qui permettent d'affiner la description de l'entité logistique. Ces méta-données auront donc un nom et un type. En plus du nom, nous associons un poids à la relation afin de déterminer la force du lien entre les deux entités. Nous pouvons dire, par exemple, qu'un container contient des palettes de sucre. Le lien entre l'entité conteneur et l'entité palette aura pour poids le nombre de palettes de sucres. De même, si nous rajoutons des cartons de chocolat dans le container, nous aurons le nombre de cartons comme poids de la relation. Nous détaillons cette notion de relation plus précisément dans le chapitre dédié à la modélisation de connaissances et à l'extraction de données par l'approche ontologique.

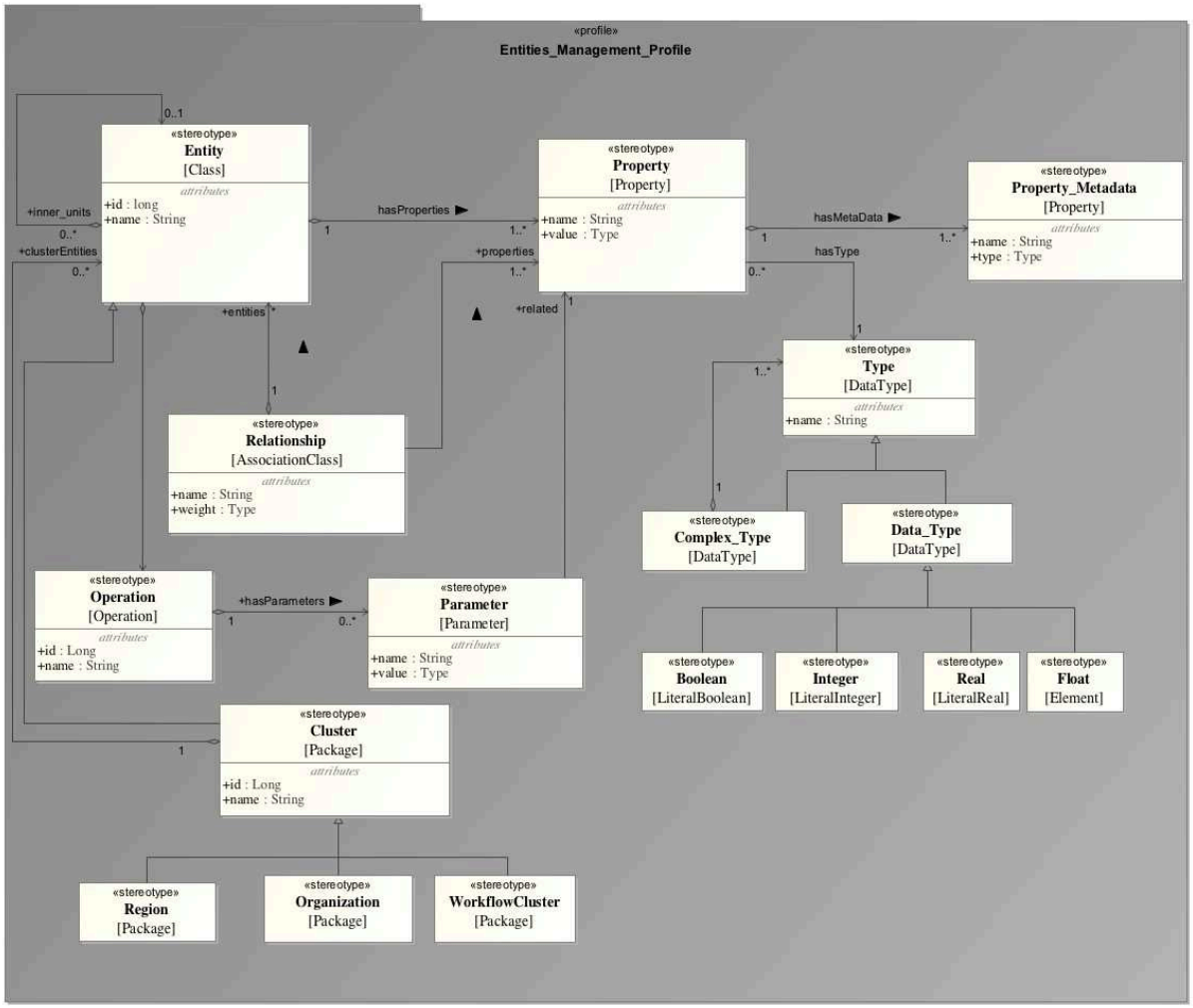


Figure 15. Profil UML des entités logistiques

**Profil UML pour la gestion des Items**

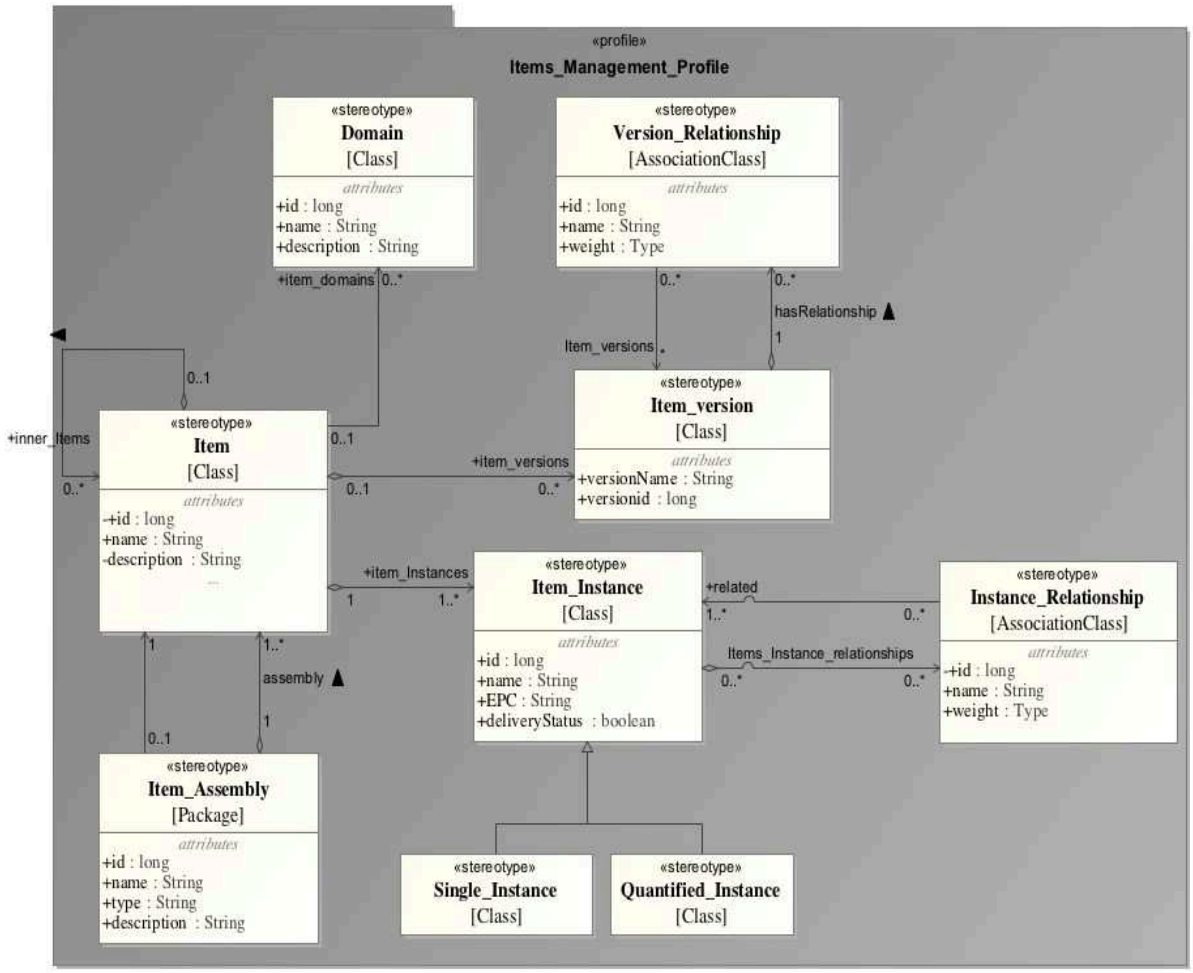


Figure 16. Profil UML des Items

Les Items sont classifiés selon leur domaine, comme expliqué précédemment. Un domaine (figure 16) est une catégorie d'Items ayant un nom et une description. Si nous classifions selon la nature de l'Item, nous entendons par le domaine, à titre d'exemple, des produits issus du pétrole, le domaine des matières premières de l'industrie manufacturière, ou le domaine des produits agricoles. Si nous classifions selon le type de produit, nous pouvons sous-classer par exemple en plusieurs sous domaine (agricole par exemple) comme le fret, les produits céréaliers, les légumes, les viandes, les poissons frais, etc.. Les instances d'un Item ont un numéro EPC unique qui permet d'assurer leur traçabilité tout au long de la chaîne logistique. Le statut de la marchandise est mise à jour régulièrement via l'attribut *deliveryStatus*. Les instances d'un Item peuvent avoir des relations d'interdépendance multiforme, pondérées par le poids de la relation ou de la dépendance. Les assemblages d'Items sont des produits (marchandises) agrégés à partir d'autres marchandises (Items). Il s'agit par exemple de palettes contenant un ensemble de produits. Les versions des Items permettent de gérer la périodicité dans le cycle de fabrication des produits, tant au niveau du client qu'au niveau fournisseur. Un Item peut avoir donc plusieurs versions caractérisées par le nom et l'identifiant de la version. Les versions sont aussi interdépendantes, par exemple dans les flux de logistique automobile une voiture possède une version et cette version a une

version de moteur. Le raisonnement est identique pour les autres Items des flux logistiques issus d'autres chaînes logistiques.

**Profil UML pour la gestion des processus Workflow (Workflow process)**

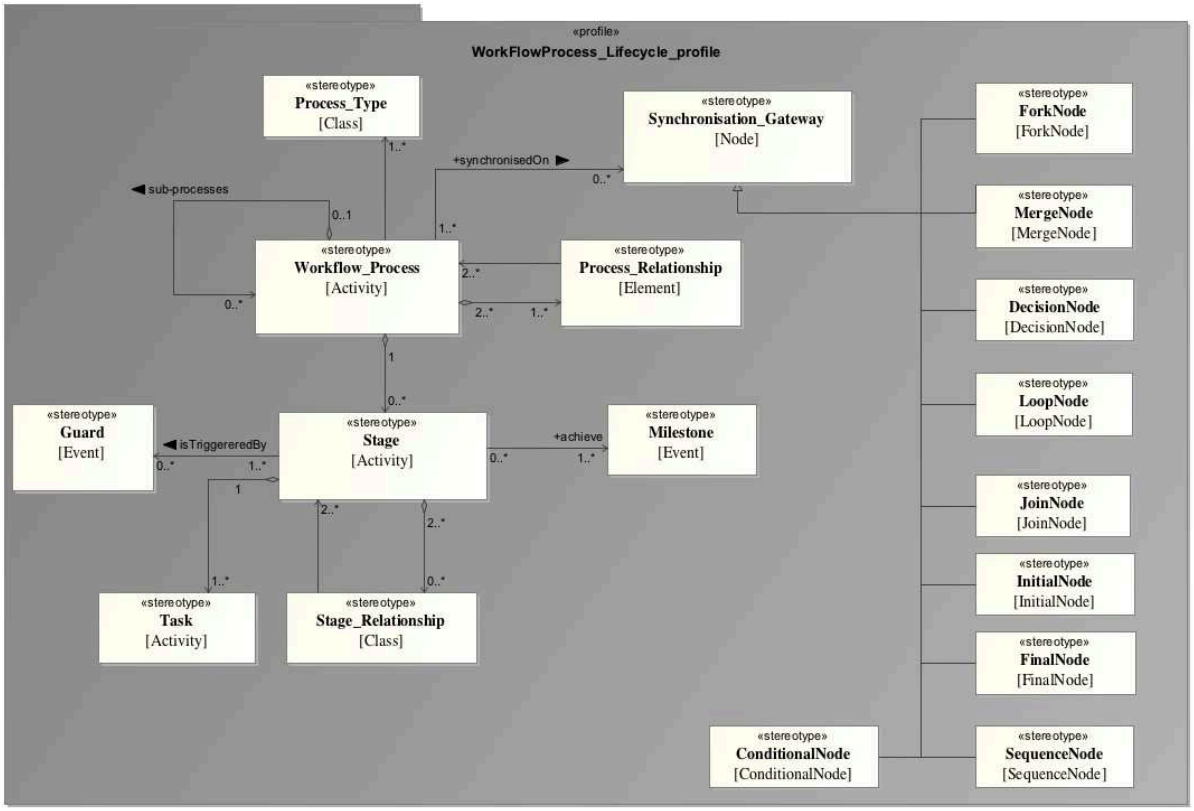


Figure 17. Profil UML des Workflow Process

Un processus est la composition d'un ou plusieurs activités (Stages) (figure 17), les stages sont composés de tâches (Task). Les guards sont des déclencheurs de processus, ils peuvent être utilisé pour démarrer un stage ou une tâche (task). Un stage peut atteindre un ou plusieurs objectifs (Milestone). Les relations d'interdépendance entre processus sont représentées par la classe Process_Relationship. Lorsque plusieurs processus se synchronisent sur un canal, il est necessaire de préciser le type du canal (Synchronisation_Gateway).

**Profil UML pour la gestion du contexte des entités et des évènements**

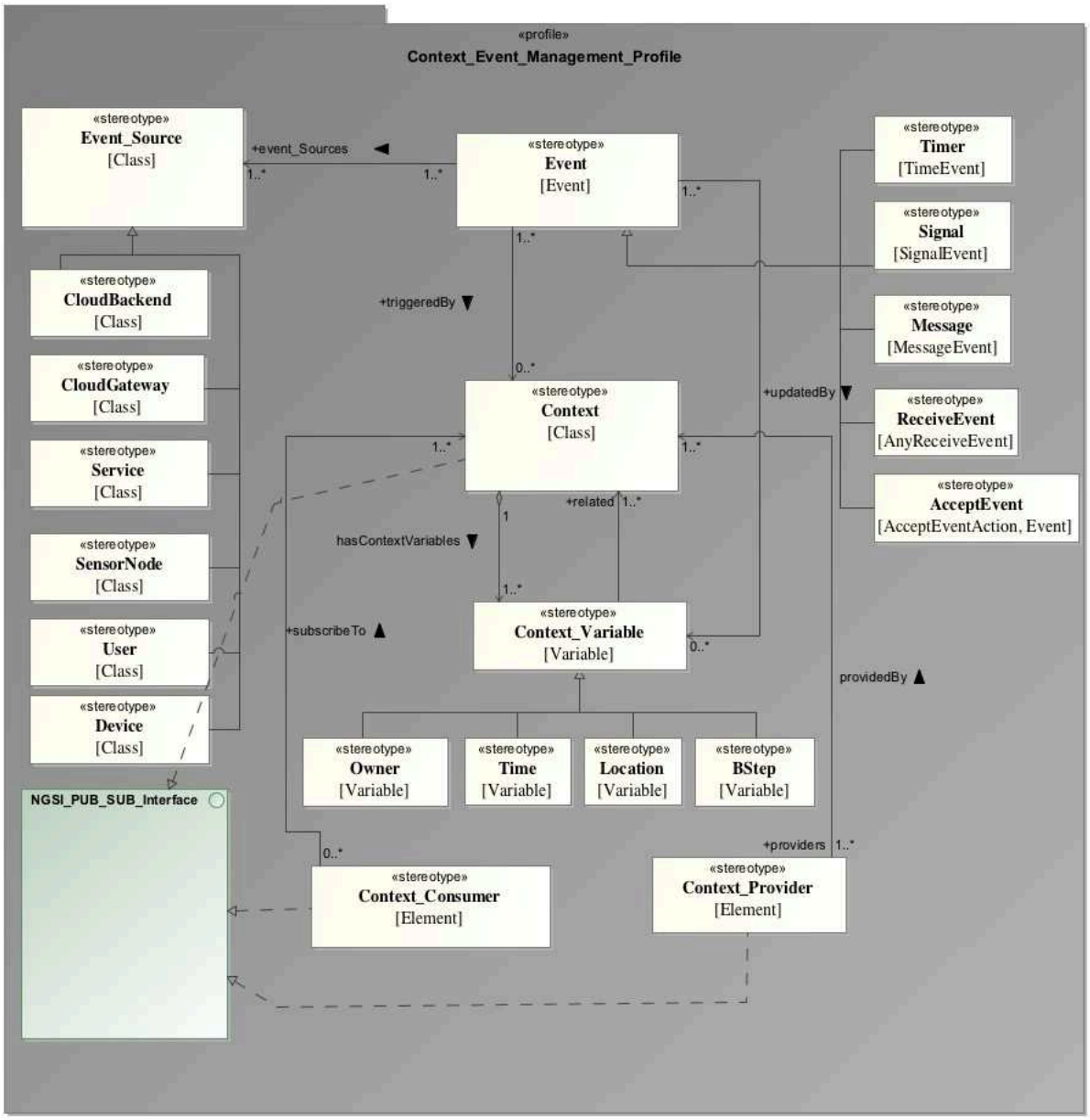


Figure 18. Profil UML pour le contexte de l'objet logistique

Le contexte de l'objet logistique est décrit par un ensemble de variables (***Context_Variable***) (figure 18). Ces variables sont mises à jour par des événements qui perturbent le contexte lorsque le flux évolue (changement d'état). Un événement a une source, il s'agit de l'entité qui a émis cet événement (capteur, utilisateur, service, etc.). Plusieurs entités peuvent souscrire à un contexte (***Context_Consumer***) afin de consommer les événements qui sont publiés dans ce contexte. De même, les entités logistiques peuvents publier des événements dans un contexte précis (***Context_Provider***).

**Profil UML pour la gestion des organisations**

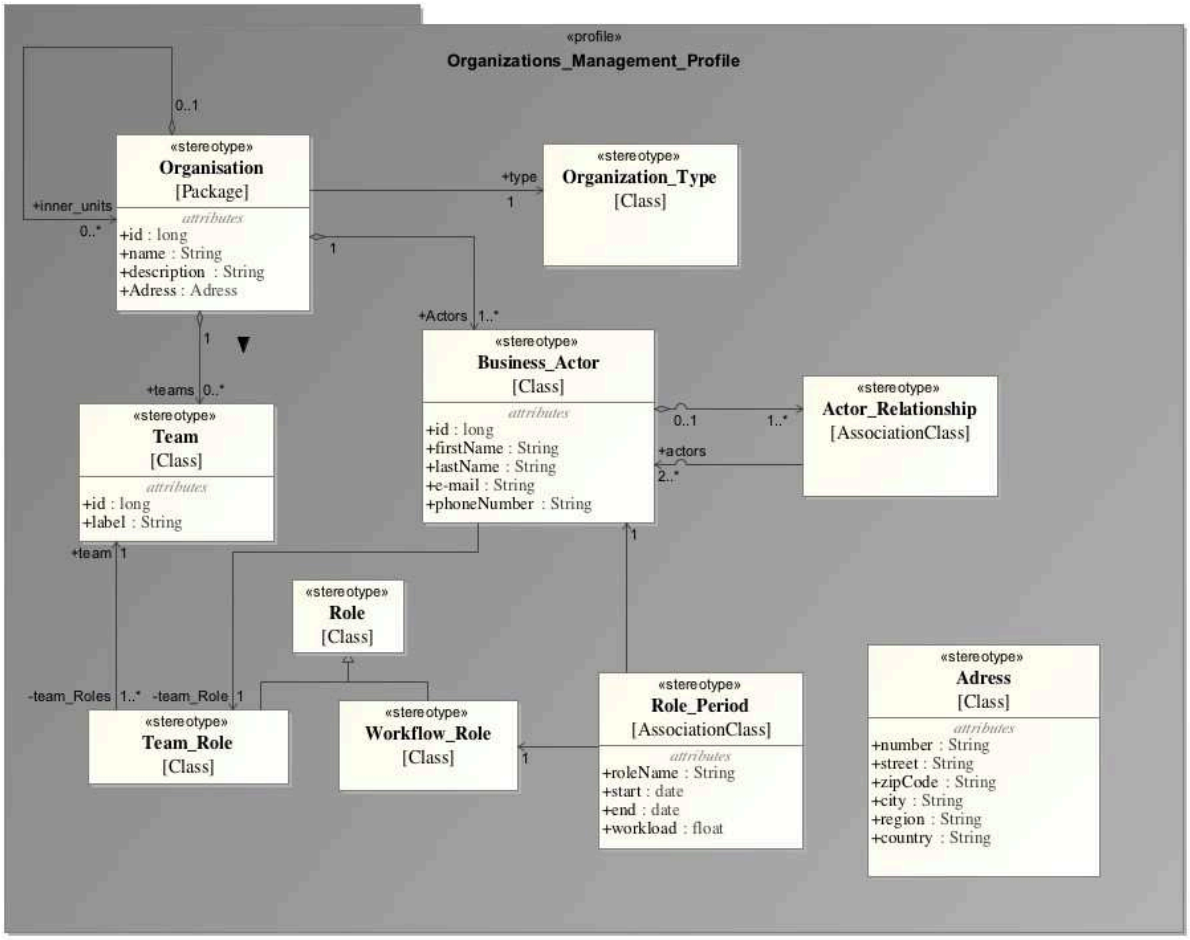


Figure 19. Profil UML pour la gestion des organisations et l'attribution des rôles

L'organisation est décrite par son nom et son adresse, elle a un identifiant unique dans la plateforme. L'adresse d'une organisation est de type Adresse, comportant toutes les informations nécessaires à la localisation de l'organisation. Les acteurs de la supply chain sont censés exercer leur rôle dans le cadre d'une organisation (figure 19). Les acteurs ont un nom, un email pour la notification des événements du flux, et un numéro pour le même besoin. Les rôles d'acteurs peuvent être périodiques ou non, d'où l'intérêt de préciser quand l'acteur exerce ce rôle sur le flux. Le rôle a donc un nom de rôle, sa date de début et de fin, et la charge de travail (workload) optionnelle.

**Profil UML pour la gestion des droits d'accès et des autorisations**

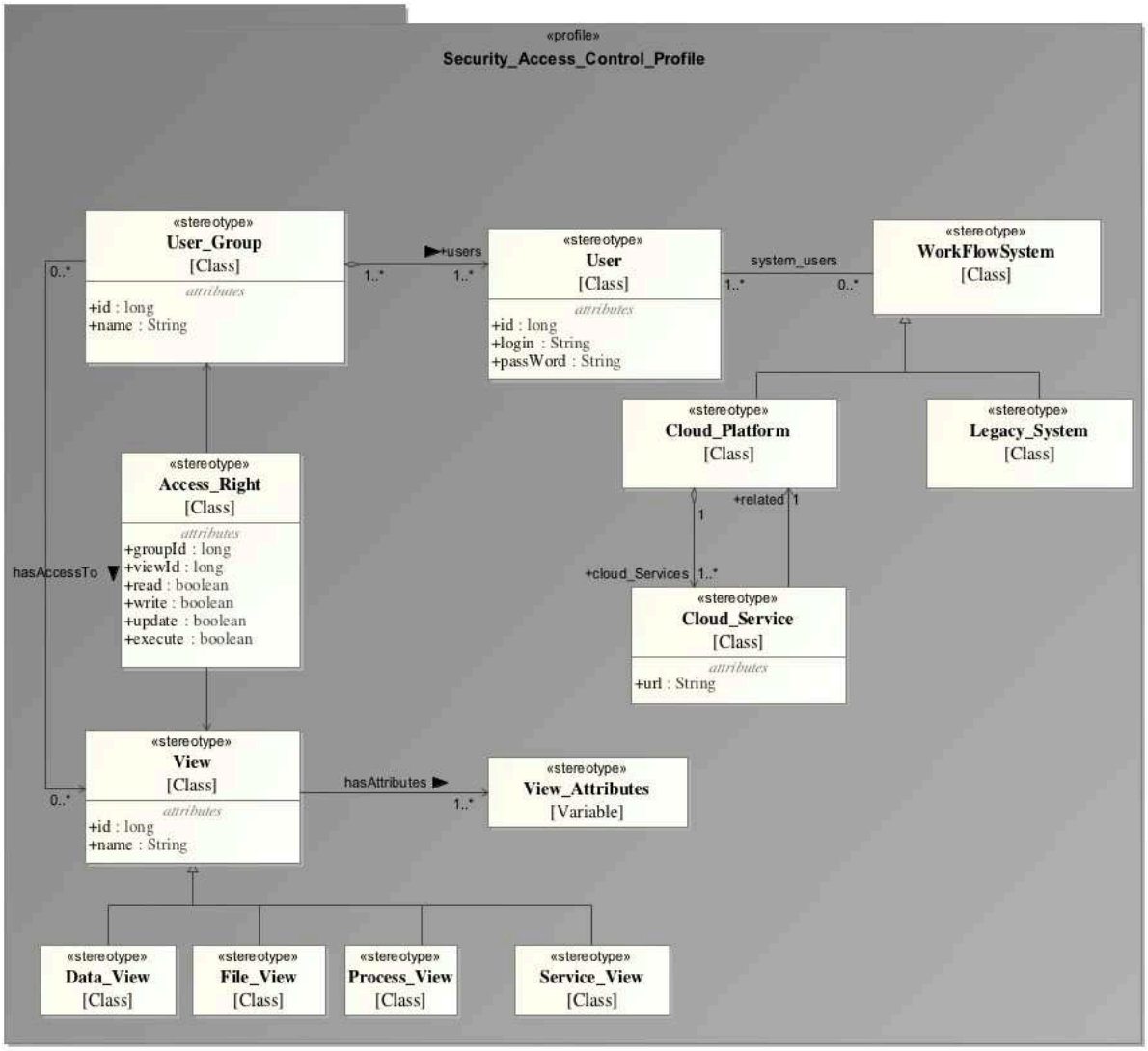


Figure 20. Profil UML pour la gestion de la sécurité et droit d'accès

La sécurité est un aspect important dans la gestion des flux de marchandises. Par sécurité, nous entendons la sécurité des biens et services, la sécurité des acteurs logistiques, qui interagissent avec le flux, et surtout la sécurité du flux d'information associé au flux physique. Dans le diagramme de la figure 20, les utilisateurs (users) sont regroupés par groupe d'utilisateurs (User_Group), ayant des vues différentes sur les données, les fichiers, les processus et les services logistiques. Le groupe d'utilisateurs est autorisé à accéder à une vue du flux selon ses droits d'accès, allant d'un simple droit de visualisation à la mise à jour des informations concernant cette vue. Une vue est une agrégation de propriétés qui permettent de décrire un état du Workflow, du service concerné. Les acteurs logistiques accèdent aux systèmes de gestion d'information en tant qu'utilisateurs de tels systèmes. Un système de gestion peut être une plateforme de collaboration Cloud, ou tout système propriétaire, ERP, WMS, etc..

**Profil UML pour la gestion des contraintes et des règles métier**

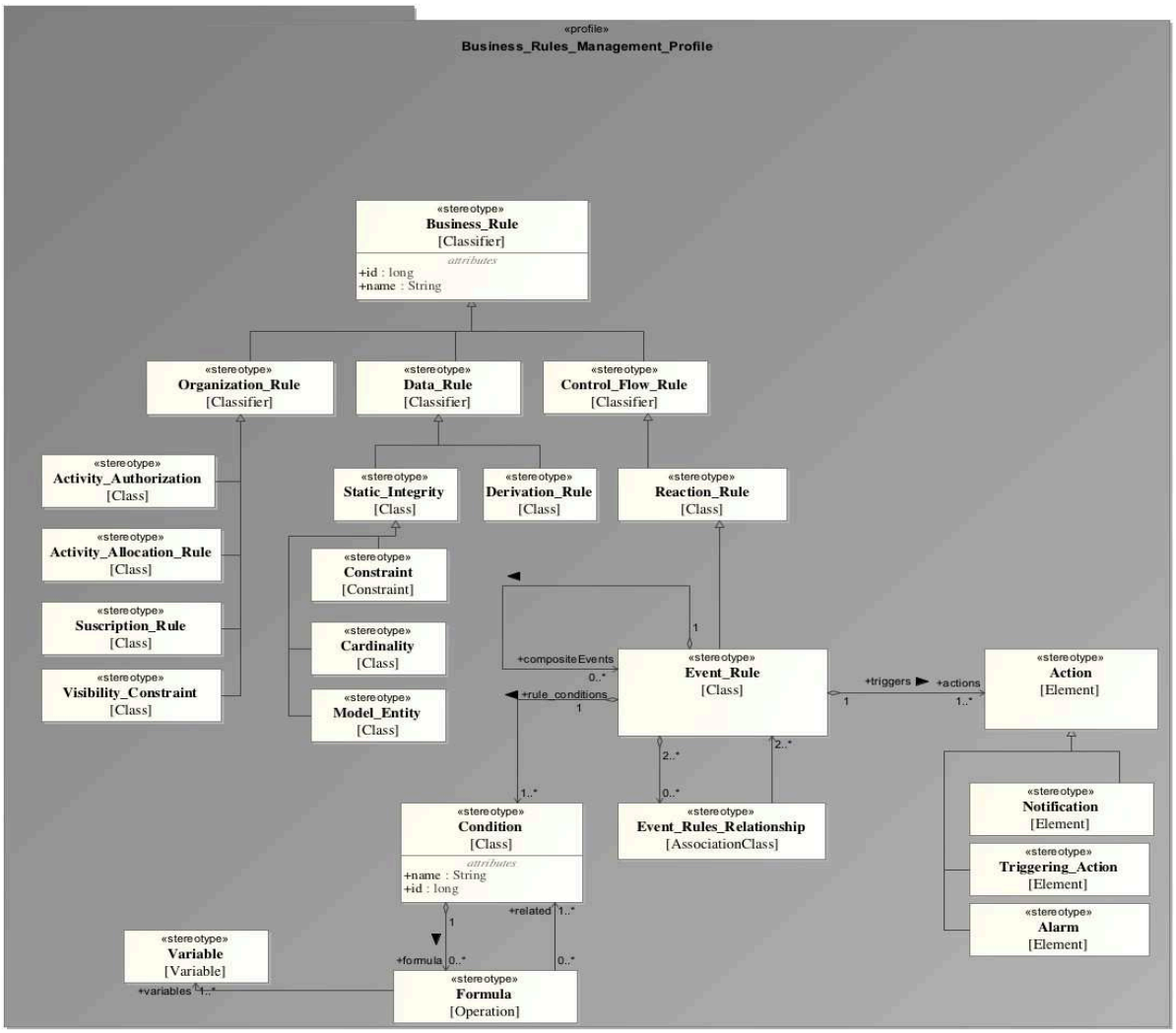


Figure 21. Profil UML pour la gestion des cotraintes et des règles metier

La gestion des règles métier est du ressort des DBRMS (Data Base Rules Management Systems), nous assistons de plus en plus à une forte intégration des DBRMS avec les applications propriétaires, ou des plateformes Cloud comme dans Oracle Service Bus, Apache ECA, Drools fusion middleware. Le profil UML que nous proposons (figure 21) est un extrait du méta-modèle de la spécification SBVR [104] et adapté aux besoins du projet COM-SLoT.

**Expression des contraintes sur le profil UML avec des règles OCL**

Le tableau 3 donne un ensemble des contraintes exprimées sur le méta-modèles des Items en utilisant le langage OCL [102]. Il s'agit des contraintes sont des contraintes d'unicité, d'agrégation et des invariants sur les relations entre Business units.

### Tableau 3: Exemple de contraintes sur l'Item

| Libellé de la contrainte | Description | Expression OCL |
|---|---|---|
| ItemCtr1 | Unicité du code EPC, du nom et de l'id de deux instances d'un même Item. | **context** Item<br>**inv**: Item.item_instances()->forAll(B1, B2 \| B1 <> B2 **implies** (B1.EPC <> B2.EPC AND B1.name<>B2.name AND B1.id<>B2.id)<br>) |
| ItemCtr2 | Les assemblages d'Items doivent contenir des Items différents. | **context** Item_Assembly<br>**inv**: Item_Assembly.assembly->forAll(assb1, assb2 \| assb1 <> assb2 **implies** assb1.name <> assb2.name AND assb2.id<>assb1.id) |
| ItemCtr3 | Les entités agrégées doivent être différents les uns des autres | **context** Item<br>**inv**: Item.inner_Items()->forAll(inner1, inner2 \| inner1 <> inner2 **implies** (inner.id <> inner2.id AND inner1.name<>inner2.name) |
| ItemCtr4 | Un Item doit avoir au moins une version, et de deux Items différents ne peuvent avoir les mêmes versions. | **context** Item<br>def:itemVersions = self.item_versions()<br>**inv**: Item->forAll(item1, item2 \| item1 <> item2 **implies (**<br>item1.itemVersions -> notEmpty()<br>AND item2.itemVersions -> notEmpty()<br>AND item1.itemVersions -> intersection( item2.itemVersions) -> isEmpty()) |
| ItemCtr5 | Les instances d'un même Item sont toutes soit unique ou quantifié. | **context** Item<br>def:itemVersions = self.item_versions -> collect(id)<br>**inv**: Item->forAll(item1, item2 \| item1 <> item2 **implies (**<br>item1.itemVersions -> notEmpty()<br>AND item2.itemVersions -> notEmpty()<br>AND item1.itemVersions -> intersection( item2.itemVersions) -> isEmpty()) |
| ItemCtr6 | Tous les Items Agrégés ont comme référence l'item agrégat et vice versa. | **context** Item<br>def:singleInstances = self.item_instances.collect(ocllsTypeOf(Single_Instance))<br>def:quantifiedInstances = self.item_instances.collect(ocllsTypeOf(Quantified_Instance))<br>**inv**: self.singleInstances -> Intersection (self.item_instances) -> isEmpty() |
| ItemCtr7 | Un Item doit avoir au moins un domaine, une version, et une instance | **context** Item<br>**inv**:<br>self.item_domains->size() > 0 AND<br>self.item_versions->size() > 0 AND<br>self.item_instances->size() > 0 |
| ItemCtr8 | Une relation entre instances lie au moins deux instances différentes du même Item ou de deux Items différents. | **context** Item_Instance<br>**inv:**<br>self.items_instance_relationships()->notEmpty() **implies**<br>**context** Instance_Relationship<br>**inv**:<br>self.related.collect(Id) -> size() > 2 |
| ItemCtr9 | Les relations entre deux versions différentes d'une même entité | **context** Item_Version<br>**inv:**<br>self.hasRelationship()->notEmpty() **implies**<br>**context** Version_Relationship<br>**inv**:<br>self.item_versions.collect(Id) -> size() > 2 |

**Personnalisation des diagrammes**

Dans une démarche MDA, il est nécessaire de personnaliser les diagrammes afin de faciliter la représentation des instances du modèle à partir du DSML proposé. Pour cette tâche, nous proposons deux diagrammes, un diagramme qui regroupe les concepts du niveau stratégique du flux (Item, Service, Provider, From, To). Ces concepts sont regroupés au sein de la DDSN node comme illustré dans la figure 22. L'exemple suivant est obtenu par le diagramme du niveau stratégique personnalisé dans MagicDraw. Il présente le processus de "custom packaging" du sucre entre Houtch, Tereos et Soffresco.

- **Diagramme du niveau stratégique du flux de sucre entre Houtch-Tereos-Sofresco**

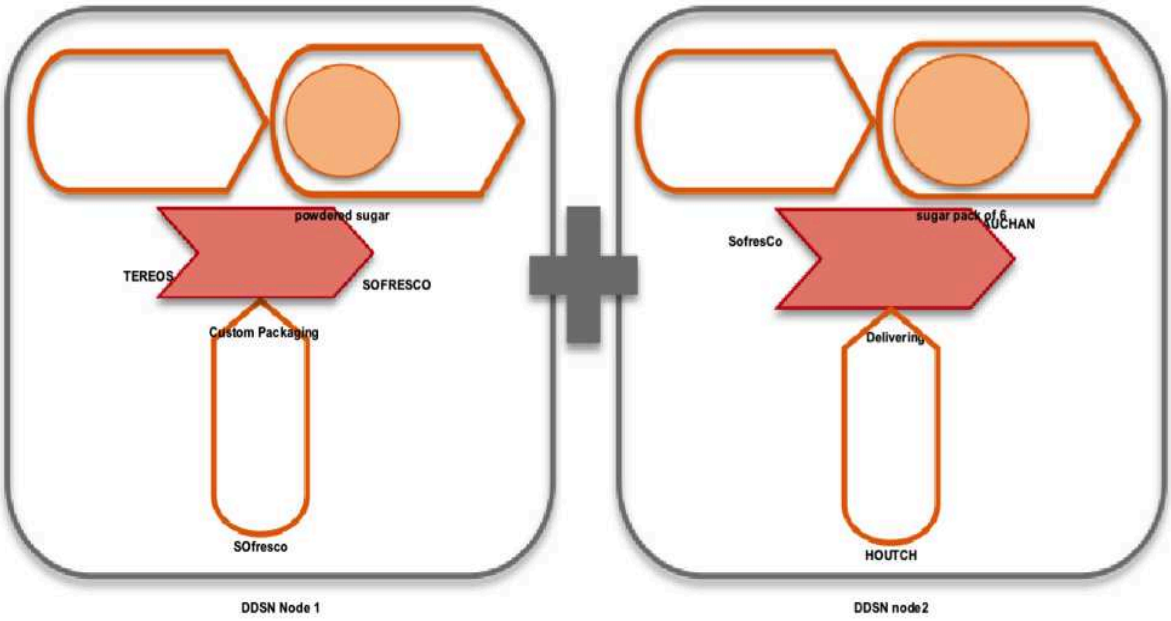


Figure 22. Diagramme du niveau stratégique d'un flux physique

- **Diagramme du niveau opérationnel ou Cycle de vie du processus**

Le niveau opérationnel du flux est une vue qui présente le cycle de vie du flux, c'est à dire l'ensemble des événements, qui surviennent dans le flux, l'ensemble des tâches, des sous processus qui compose le flux principale. Il ressort aussi les objectifs visés par chaque sous processus, et les objectifs finaux du workflow principal. Comme illustré par la figure 23 et par rapport au processus de reconditionnement du sucre de l'entreprise TEREOS par l'entreprise HOUTCH-SOFRESCO, on observe qu'à la réception du planning prévisionnel, la condition (appelée aussi Guard dans le jargon du concept GSM : Guard-Stage-Milestone) *Draft received* est satisfaite et déclenche le processus de reconditionnement. Le conditionnement effectif du sucre ne se fera qu'après réception des premiers lots ou batch. La condition sucre disponible sera donc satisfaite (powdered sugar available). Dans ce cas, le sous processus "Custom packaging" est déclenché, à condition que le matériel nécessaire soit aussi disponible (cartons d'emballage, film, etc.). Les tâches de déchargement,

stockage, et packaging proprement dit sont effectué sur le sucre en poudre pour atteindre l'objectif business "Custom Packaging completed". Ainsi, après l'achèvement du premier sous processus, le processus de livraison est déclenché en cascade, sous la condition supplémentaire d'avoir un camion disponible pour le transport. Ce dernier se termine par la livraison du sucre reconditionné à tous les clients concernés, et ainsi l'atteinte du milestone "packaged sugar delivered" qui marque la fin des livraisons. Le processus principal se termine donc par l'atteinte de son objectif principal qui consiste à livrer le produit reconditionné aux clients finaux, d'où le milestone "*Delivery completed*".

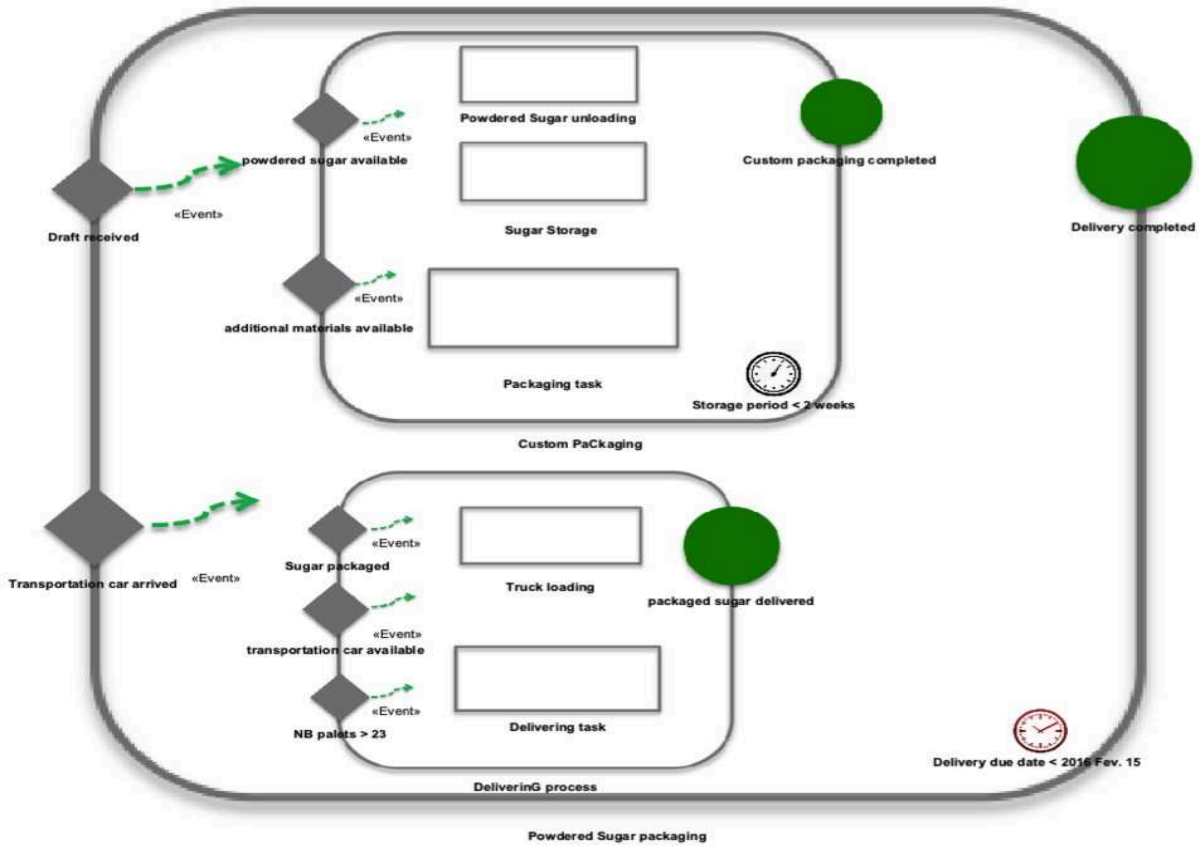


Figure 23. Diagramme du niveau tactique d'un flux physique (cycle de vie du flux)

### 2.3.3. PSM pour le modèle de déploiement du flux dans des environnements Cloud: cas de Google App Engine (GAE)

La plateforme cible que nous étudions est celle de Google App Engine. GAE ou App Engine est une des PaaS (Platform as a Service) de Google dédiée à la mise en place et au déploiement d'applications web. GAE est aussi utilisée pour la construction de Backends d'applications mobiles. Google propose d'autres PaaS dédiées au stockage de données. Parmi ces PaaS, le Cloud Storage qui fournit un service de stockage d'objets avec gestion du *caching*; le Cloud Bigtable pour le stockage de bases de données NoSQL; le Cloud Datastore qui représente une base de données NoSQL pour le stockage de données non relationnelle; et le Cloud SQL pour le stockage de données relationnelle type MySQL. Les infrastructures de Google sont nombreuses. Nous pouvons citer Container Engine dédié à

l'exécution de container Docker; Compute Engine pour l'exécution de workloads sur des machines virtuelles hébergées par Google; Cloud Functions (en version Alpha), une plateforme dédiée aux micro-services basée sur une architecture EDA (Event Driven Architecture). Google propose d'autres services et plateformes pour la virtualisation (Cloud Virtual Network): le Load Balancing (Cloud Load Balancing), le DNS (Cloud DNS) et la gestion des interconnections d'infrastructures d'entreprises à la plateforme Cloud de Google (Cloud Interconnect).

En raison des multitudes de plateformes existantes, nous nous intéressons plus particulièrement au ***Cloud Datastore*** sur lequel cette étude est axée. Le Datastore est une base de données NoSQL qui permet de stocker de grands volumes de données, de gérer l'évolutivité, la réplication et le redimensionnement automatique en fonction de la charge de l'application. Le Datastore prend aussi en charge la gestion des transactions ACID, la gestion des requêtes de type SQL et des index. La structure de données du Datastore est celle des tables de hachage hiérarchique dans laquelle on accède à la valeur d'un élément grâce à sa clé unique dans la base. Le schéma de la figure 24 montre un exemple d'une telle structure.

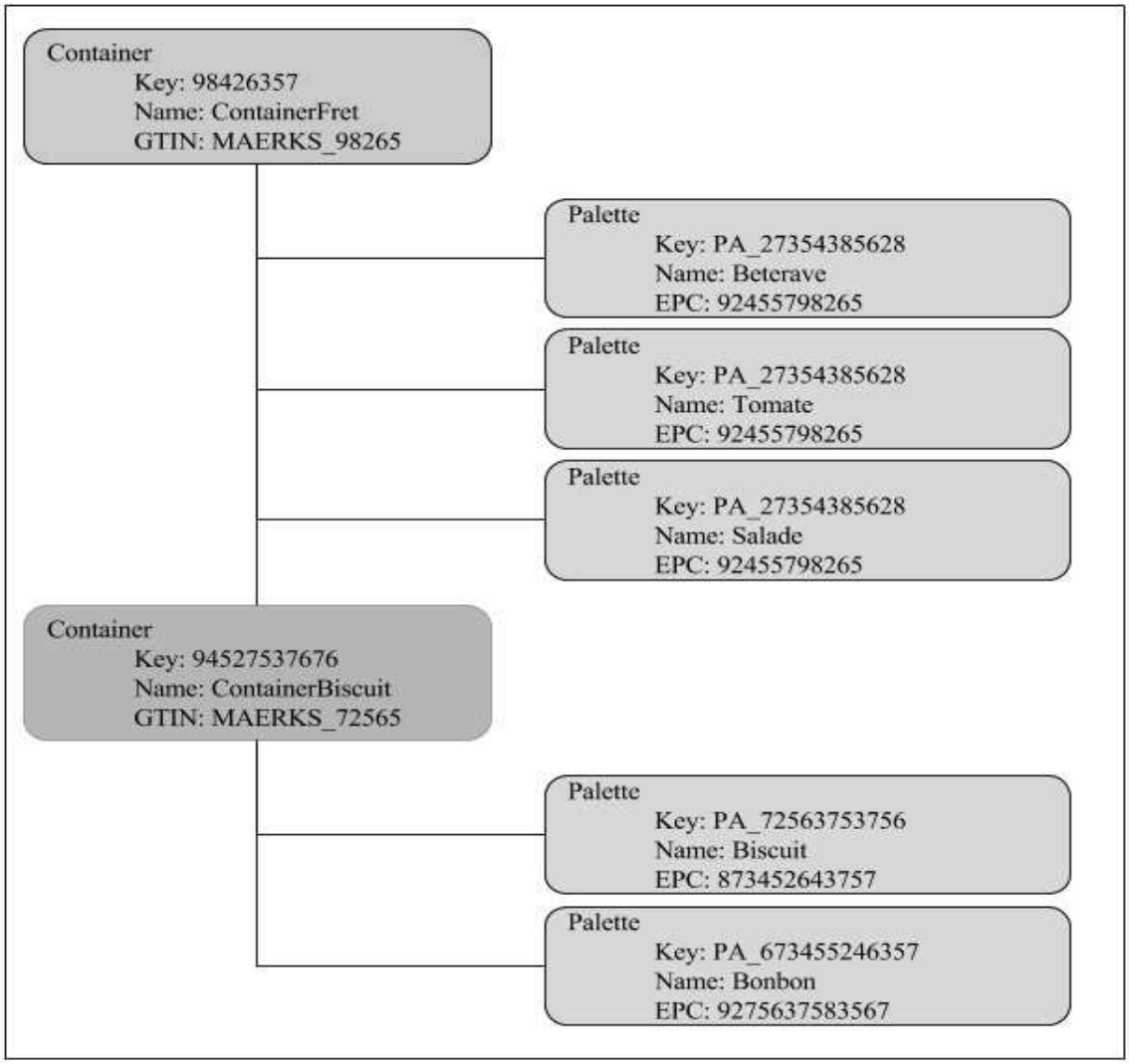


Figure 24. Exemple de stockage dans le Cloud Datastore

Dans l'exemple de la figure 24, nous avons deux types d'entités ou Kind (terminologie de Google), les entités de type Container et les entités de type Palette. Pour le type Container, nous avons deux instances différenciées par leur clé (Key) et leur GTIN (Global Trade Item Number). Pour le container de biscuit, nous avons créé deux entités fils de type Palette, et de même, nous avons créé trois palettes pour le conteneur de Fret. Chaque palette a comme parent (Root Entity) un objet de type Container. Les Container et les Palettes sont vus par le Datastore comme étant tous des Entity, l'élément de stockage de base du Datastore.

Dans ce qui suit, nous créons le modèle de stockage de données du Datastore en vu de réaliser un PSM pour le GAE Cloud Datastore (figure 25). Cette spécification est incomplète de part le manque d'information et de documentation sur la structure interne et du Datastore et son fonctionnement. Le PSM que nous présentons ci-dessous représente donc une partie du méta-modèle du Datastore que nous avons réalisé en nous basant sur le peu de documentation officielle publiée par Google.

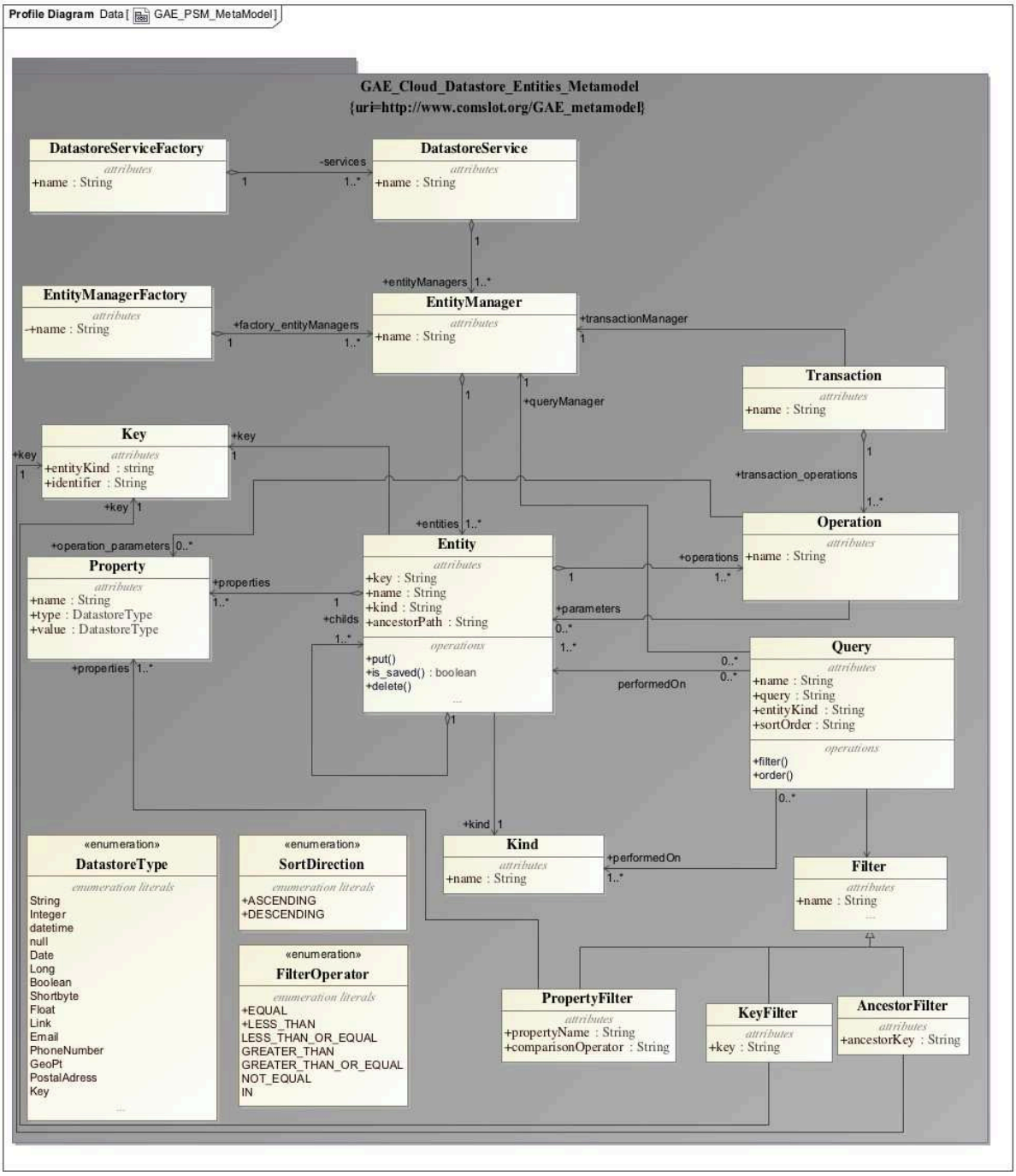


Figure 25. Méta-modèle des entités du Cloud Datastore

Le Méta-modèle du Cloud datastore (figure 25) a pour pilier l'entité (***Entity***). Une entité est l'équivalent d'une classe dans la POO ou dans la spécification MOF. L'Entity a une clé unique, un nom et des propriétés (Property). Les entités sont stockées de manière hiérarchique dans le sens de l'héritage ou de la composition. Ainsi, une entité peut avoir plusieurs entités fils (***childs***) et les fils ont une seule entité comme parent (*rootEntity*). La méta-classe ***Property*** est l'équivalent de Attribut dans la spécification MOF. Elle permet de définir les propriétés d'une entité au même titre que les attributs d'une classe dans un diagramme de classes UML. Une entité a un type (Kind).

La méta-classe ***Kind*** est équivalente à la méta-classe Classifier du MOF, et permet ainsi de classifier les entités (Entity) selon leur type. Une clé (***Key***) est un champ composé, ayant deux parties: le type de l'entité (entityKind) et l'identifiant de l'entité dans la base de donnée (Identifier). En général, la clé est générée par le Datastore, et cette génération automatique peut être remplacée par un algorithme de gestion des identifiants propres au programmeur. Sur une entité, nous pouvons réaliser des opérations (***Operation***) qui peuvent être des opérations basiques de type CRUD (Create, Read, Update, Delete), ou des opérations complexes. Les opérations prennent en paramètres soit des entités (*parameters*) ou des propriétés d'entités (*operation_parameters*). Les opérations transactionnelles, comme la mise à jour d'entités fortement liées, sont regroupées dans une même transaction qui est gérée par un manager de transactions (***EntityManager***).

Les requêtes (***Query***) sont réalisées par l'opération (*performedOn*) sur des classes d'entités, et peuvent avoir des filtres soit sur les propriété de l'entité (***PropertyFilter***), soit sur les clés (***KeyFilter***) ou sur les entités parents (***AncestorFilter***). Une requête est gérée par un EntityManager qui se charge de son instanciation, son exécution et l'interprétation de ses résultats. Les résultats d'un filtre peuvent être ordonnés (***SortDirection***). Un filtre sur une propriété a un opérateur de comparaison (***ComparisonOperator***) qui permet de comparer les propriétés spécifiées.

Dans le paragraphe qui suit, nous proposons une transformation des modèles de flux logistiques (PIM) vers les modèles d'entités du Cloud Datastore (PSM).

### 2.3.4. Transformation du PIM vers le PSM avec QVT-O (*QVT-Operational*)

- **Transformation de modèle à modèle avec QVT-O**

La transformation de modèle à modèle est au coeur de la démarche MDA. En effet, MDA voit tout comme modèle. Ceci dit, qu'à travers l'approche MDA, on peut construire une application de bout en bout en passant d'un modèle à un autre par des opérations de transformation jusqu'au code, qui est lui aussi considéré comme un modèle (figure 26). Au regard des quatre niveaux de méta-modélisation du MOF, il existe trois transformations majeures. La première transformation est en général un processus d'extension du méta-modèle UML pour la mise en place d'un DSML que nous avons évoqué dans le paragraphe précédent. Enfin, la transformation du PIM vers le PSM, c'est à dire l'automatisation de la transformation du modèle source conforme au PIM en modèle destiné à la plateforme, conforme au PSM.

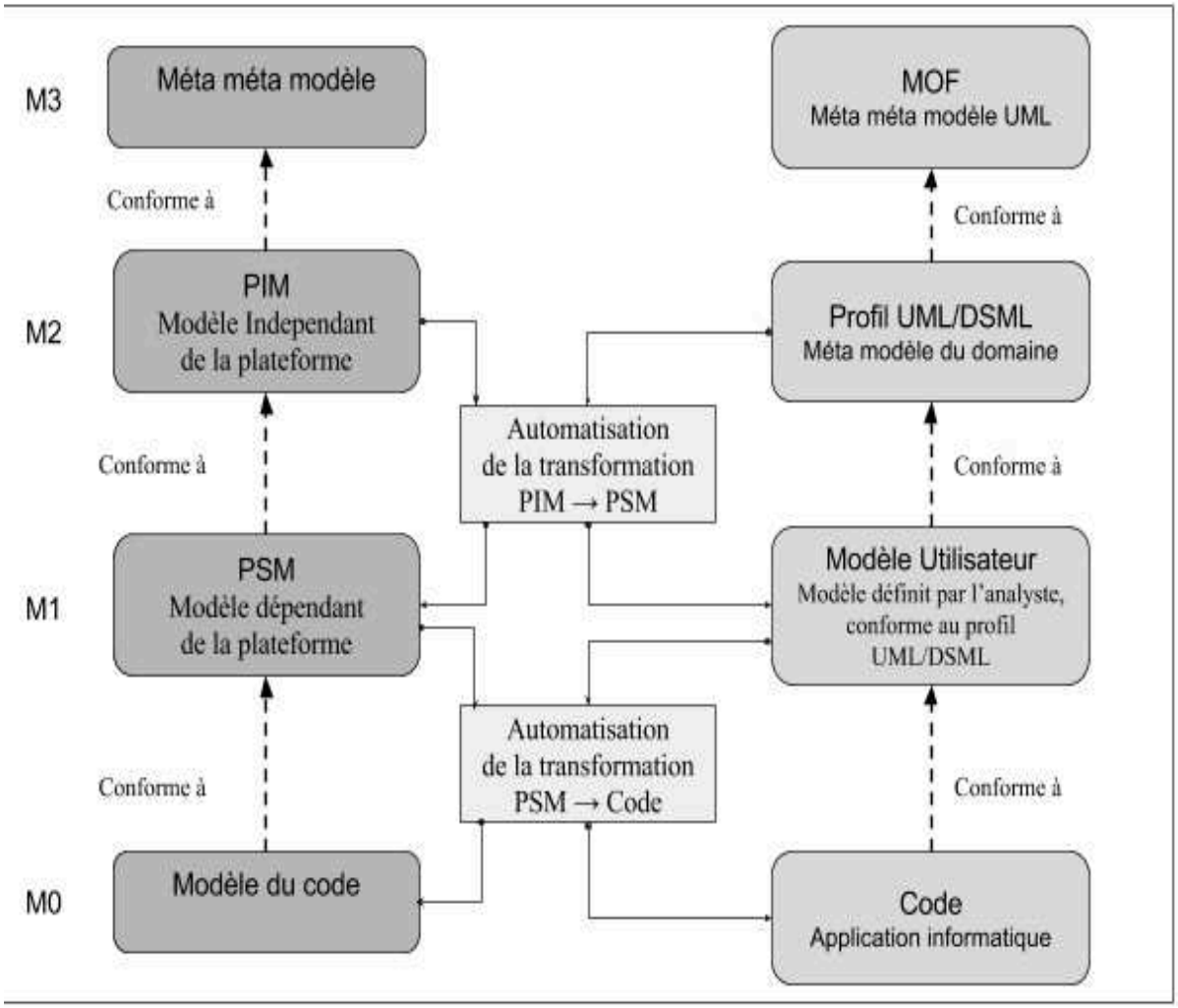


Figure 26. Niveau de méta-modélisation du MOF avec transformation

- **Les outils de transformation**

Il existe plusieurs méthodes et approches de transformation de modèle à modèle. L'une des plus anciennes consiste à utiliser les graphes de transformation, parmi lesquelles l'approche TGG (Tripple Graph Grammar) qui s'intéresse plus à la semantic, en offrant une facilité de liaison des noeuds du graphe source au graphe de destination [100]. Le MTL Model Transformation Language est un langage issu du MOF, proposé par IBM pour des fins de transformations de modèles. Atlas Transformation Language (ATL) est l'un des plus utilisés pour la transformation de modèle à modèle. ATL support le langage OCL qui permet d'exprimer des contraintes sur les modèles et les méta-modèles. Query View Transform (QVT) [103] apparaît comme le mieux recommandé du fait qu'il est un standard de OMG, avec une large communauté qui le supporte et une facilité de mise en oeuvre. QVT supporte aussi plusieurs formats de sérialisation de données comme le format ECore, XMI, XML, et s'intègre facilement dans les outils de développement de modèle tels que Eclipse à l'aide d'un plugin dédié. ATL est un langage de transformation de modèles, basé sur les graphes tout comme QVT.

- **Les concepts de base de QVT-O**

Query View Transform est le standard proposé par OMG pour la transformation de modèles. Il repose sur le MOF (Meta Object Facility) et OCL (Object Constraint Language). OCL est utilisé pour exprimer des contraintes sur tout modèle UML ainsi que les méta-modèles UML. De plus, OCL est aussi utilisé comme langage d'interrogation de modèle. OCL permet d'accéder aux éléments d'un modèle, et de parcourir son graphe de structure pour accéder et réaliser des opérations sur les éléments le constituant. Pour des modèles basés sur UML, OCL permet ainsi d'accéder à une classe ou plusieurs classes du modèle, aux attributs et méthodes de ces classes. OCL permet entre autre de spécifier des pré et post conditions sur les méthodes d'une classe ou d'un modèle. QVT est composé de trois grands packages: Operational Mappings, Core language et Blackbox implementation.

**Transformation du modèle de flux vers le modèle du DataStore**

- **Mappings des méta-classes**

Mapping des entités business (BusinessEntity) aux entités du Datastore (Entity)

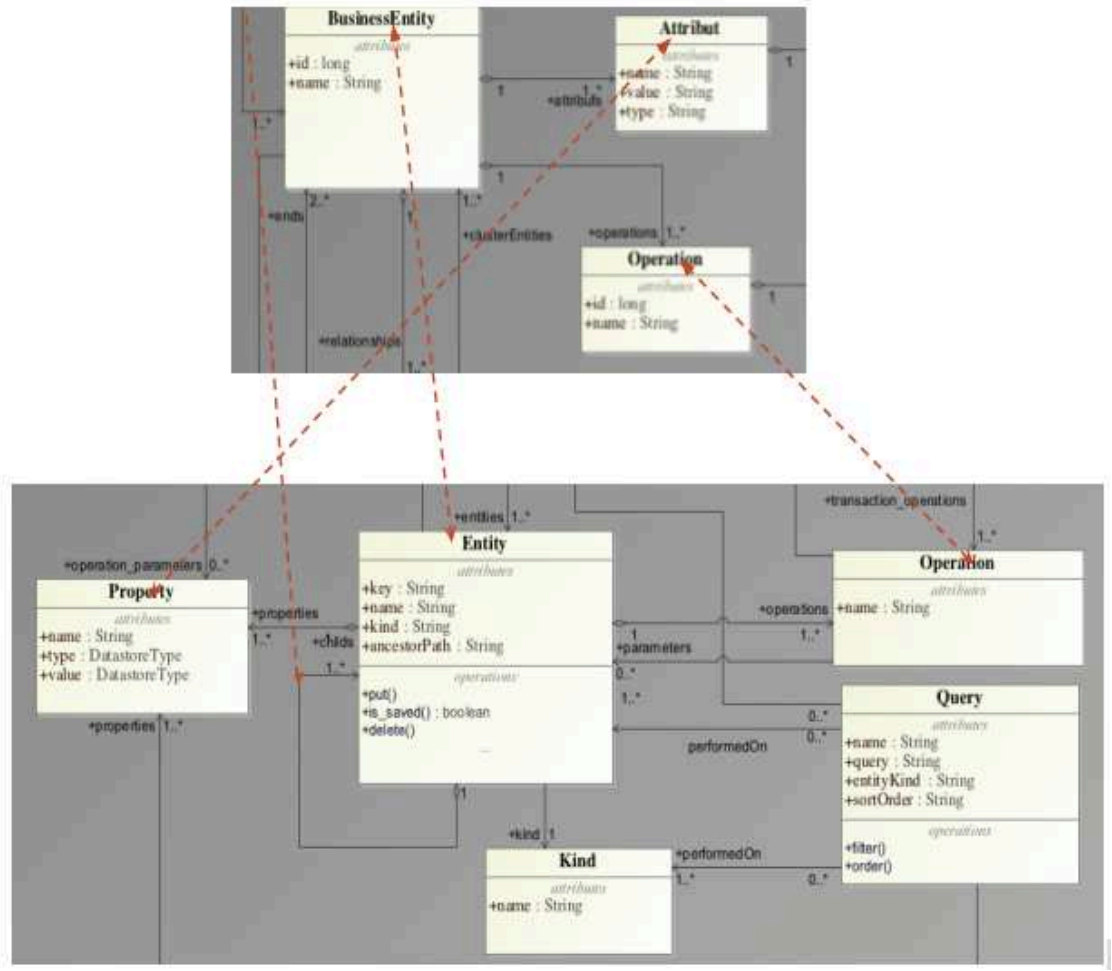


Figure 27. Mapping des entités Business aux entités du Datastore

Le mapping que nous proposons dans la figure 27 consiste à transformer les Business Entity (BE) du modèle source en Entity du Datastore dans le modèle destination. L'indentifiant du BE devient ainsi la clé de l'entité GAE. Il en est de même pour le nom. L'entité parent ou root désignée par la méta-attribut *ancestorPath* est remplacée par le container du Business Entity, obtenu par la méthode ***self.container*()**. Le type de l'entité est par défaut le nom de la classe de l'entité. Après avoir transformé Business Entity en Entity du Datastore, la prochaine étape consiste à transformer ses attributs, ses opérations et ses relations avec les autres méta-classes du modèle source. Pour ce faire, nous transformons tous les attributs du BU en Property du Datastore (*properties += self.attributs -> map attributToProperty()*;). Les opérations susceptibles d'être exécutées sur le BU sont transformées en opérations du Datastore (*entityOperations += self.operations -> map operationToEntityOperation()*;). De même, le Business Entity peut être un agrégat d'entités comme l'exemple du container

avec ses palettes. Dans ce cas, nous créons une entité composite du côté du Datastore, qui aura la même structure, en transformant la relation d'agrégation *innersEntities* du modèle source en agrégation *childs* contenant les entités composites de l'entité Datastore (*childs += self.innerEntities -> map toInnerEntities();*

Le code QVTo correspondant à ce mapping est le suivant:

```
//BusinessEntity mappings
mapping EntitiesMetamodel::BusinessEntity::businessEntityToGAEEntity() : GAE_Metamodel::Entity
	{
		init {log("Enter Business Entity Class mapping ...");}
			keyId := self.id;
			name:=self.name;
			kind := self.name;
			ancestorPath := self.container().toString();
			//mapping relations
			properties += self.attributs -> map attributToProperty();
			entityOperations += self.operations -> map operationToEntityOperation();
			entityKind := self.entityTypes -> map typeToKind();
			childs += self.innerEntities -> map toInnerEntities();
		end {log("End Mapping Business Entity Class to GAE Entity!");}
	}
```

Figure 28. Implémentation QVT du mapping des entités business en entity du Datastore.

- **Mappings des Attributs aux Property**

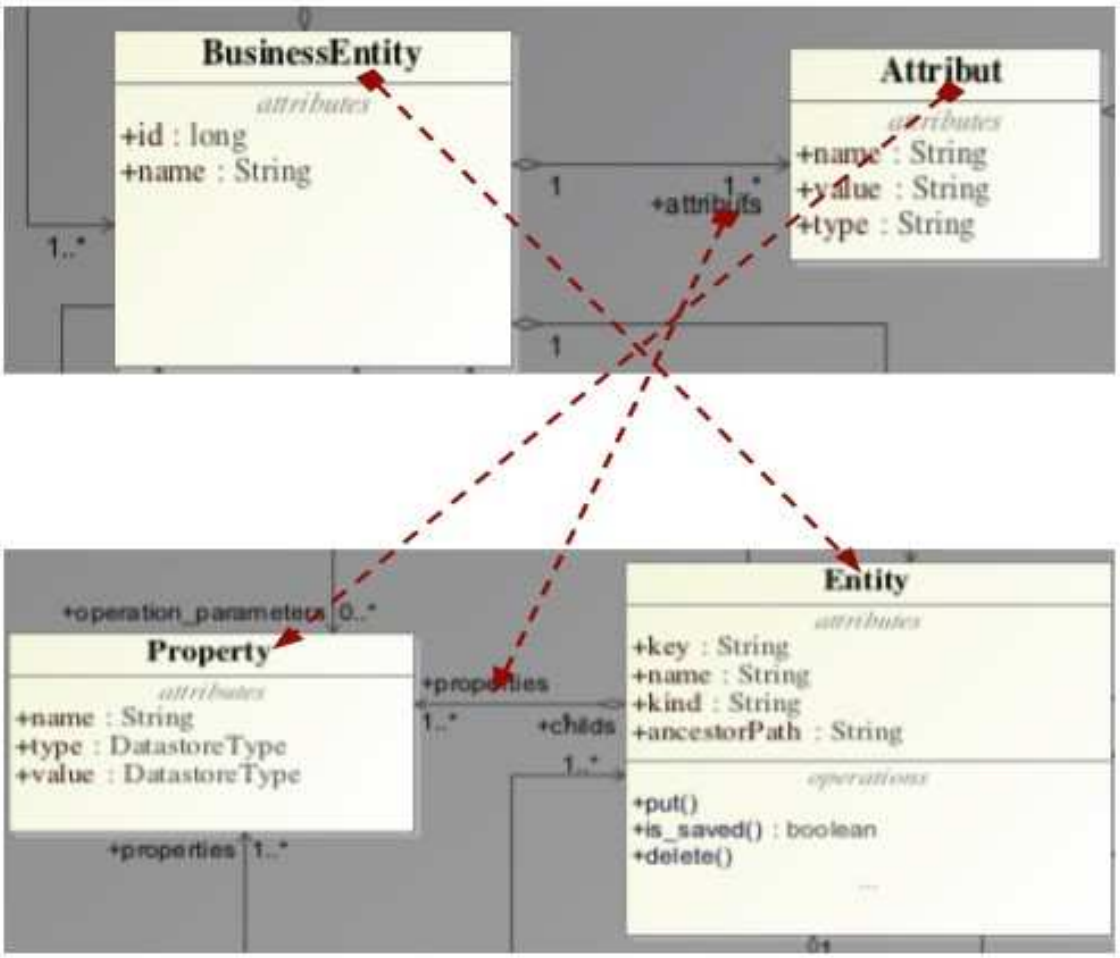


Figure 29. Mappings des Attributs aux Properties

Les attributs d'un Business Unit sont transformés en propriétés de l'entité Datastore correspondant, ce qui donne cette transformation QVT:

```
mapping EntitiesMetamodel::Attribut::attributToProperty() :
GAE_Metamodel::Property {
        init {log("Enter Attribut mapping section");}
        name:=self.name;
        type := self.type;
        value := self.value;
        end {log("End Mapping Business attributs to GAE
Properties!");}
}
```

Figure 30. Implémentation QVT du mapping des attributs du BU en Property

- **Mappings des operations**

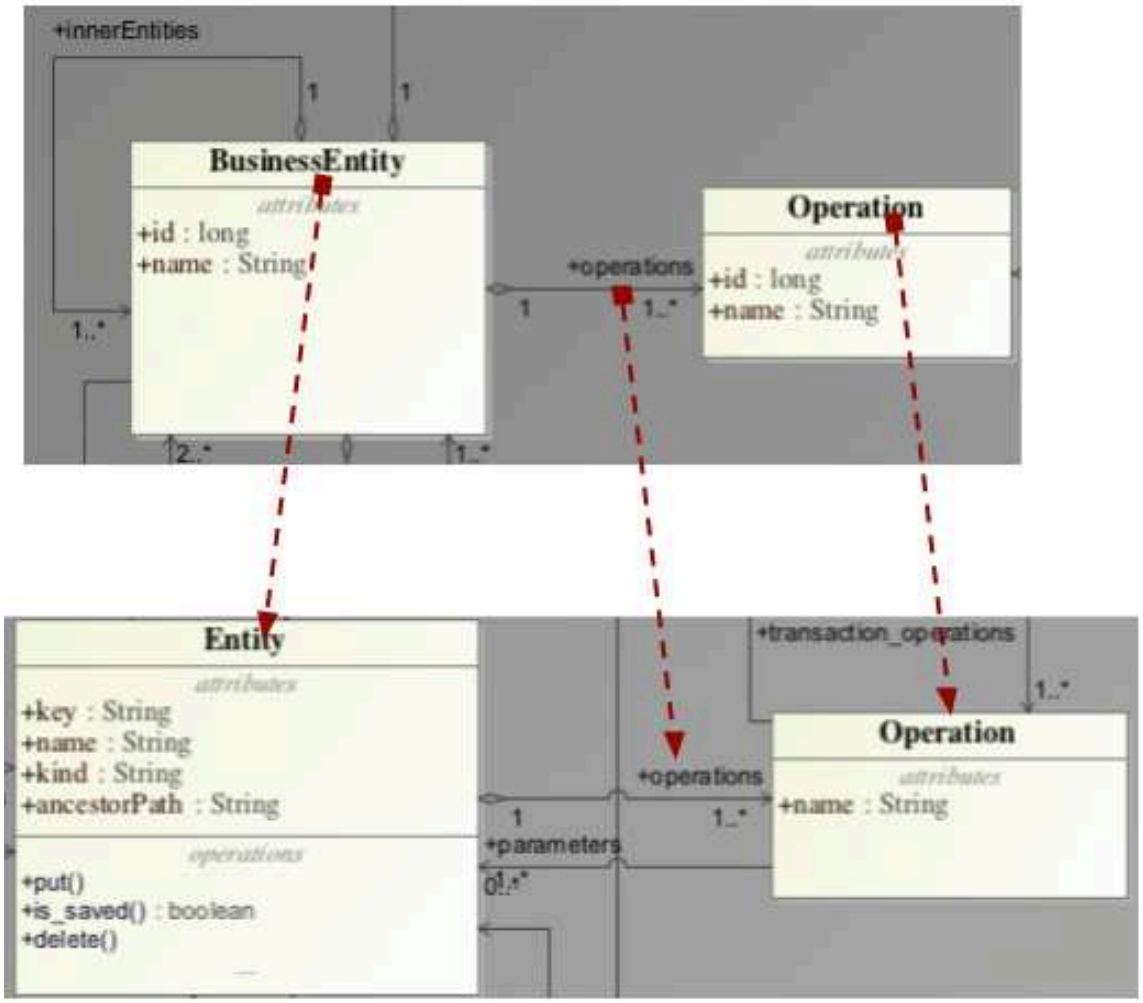


Figure 31. Mapping des operations du Business Entity vers des opérations de l'entité GAE

Les opérations exécutées sur le Business Entity sont des opérations CRUD ou des opérations complexes comme par exemple les opérations de mise à jour conflictuelles.

- **Mappings des relations de composition et d'agrégations entre business Entities**

Dans une opération de transformation, les relations d'agrégation et de composition nécessitent une attention particulière. Dans notre cas ils s'agit des BU composites, dénotées par la relation d'agrégation *innerEntities* qui boucle sur le Business Entity. Il est aussi question des relations d'interdépendances entre Business entities, et il en est de même des opérations (*Operation*) et des attributs (*Attribut*) qui décrivent le BU, les paramètres de ces opérations.

```
mapping EntitiesMetamodel::Operation::operationToEntityOperation() :
GAE_Metamodel::Operation {
        init {log("Enter Operations mapping section ...");}
        name:=self.name;
}

mapping EntitiesMetamodel::EntityClass::typeToKind() : GAE_Metamodel::Kind{
        init {log("Enter Class mapping section");}
        name:=self.name;
        end {log("End Mapping Class  to GAE Kind!");}
}

mapping EntitiesMetamodel::BusinessEntity::toInnerEntities() :
GAE_Metamodel::Entity{
        init {log("Enter Attribut mapping section");}
        keyId := self.id;
        name:=self.name;
        ancestorPath := self.container().toString();
        end {log("End Mapping Business attributs to GAE Properties!");}
}
```

Figure 32. Traduction des relations de mappings avec QVT

## 2.4. Conclusion

L'objectif de ce chapitre est de proposer une méthodologie et une nouvelle approche de modélisation de la statique du flux logistique dans le contexte du suivi de ses données. Ces dernières proviennent des objets logistiques via l'internet des objets, et partagées entre les acteurs logistiques à travers les plateformes collaboratives du Cloud.

Pour ce faire, nous avons adopté la démarche MDA dans la mesure où elle est la mieux appropriée pour ce type de tâche. La mise en place d'un méta-modèle de flux est le point d'entrée de cette démarche, permettant ainsi de capturer les concepts génériques du domaine. Suivant cette approche MDA, nous avons proposé un nouveau DSML sous forme de profil UML pour la gestion du flux de marchandises, packagé suivant les différents aspects du flux. Parallèlement, nous avons étudié la possibilité de transformer un modèle de flux conforme au DSML proposé en un modèle propre à une plateforme Cloud, en l'occurrence Google APP Engine, pour lequel nous avons proposé un méta-modèle. L'automatisation de cette transformation par la norme QVTo permet de s'inscrire dans la logique de programmation générative, qui est l'un des objectifs visés dans cette étude.

Ce chapitre a permis d'élucider les différents aspects de la statique d'un flux de marchandises et traiter de manière sommaire le cycle de vie du flux par des diagrammes du niveau opérationnel. Dans le chapitre suivant, nous abordons la dynamique du flux logistique par des outils formels de la même famille que les langages de calcul de processus LOTOS.

# Chapitre 3: Modèle dynamique du flux logistique

## 3.1. Introduction

Le problème de modélisation de la dynamique des flux logistiques et particulièrement celui de l'interopérabilité entre des systèmes communicants d'une part, et entre ces systèmes et les acteurs logistiques d'autre part. De manière générale, le problème de modélisation de la dynamique de ces systèmes peut être considéré comme un problème d'ordonnancement spatio-temporel des systèmes distribués, de part la prédominance du facteur temps et de la notion d'événements au centre du flux logistique.

Dans la littérature, l'ordonnancement tem porel des systèmes à événements discrets et des systèmes distribués a été adressé par plusieurs auteurs, suivant plusieurs axes de recherche. Un premier axe consiste à considérer le flux d'un système comme étant un processus composé d'une séquence d'activités. Ces activités sont à leur tour constituées d'une séquence de tâches et dont la synchronisation peut être faite par des événements ou des points de jonction. Dans ce premier axe, nous citons les travaux précurseurs de Van Der Aalst [105], qui considère l'ordonnancement du flux logistiques comme étant du "*case management*", avec l'introduction de la notion d'activité, de transition, de point de routage ou *gateway*, combiné à la notion d'événement. Cette approche a évolué et subit plusieurs critiques, bien que plusieurs auteurs y aient contribués aussi. Nous pouvons dire que cet axe de recherche orientée processus et activités (***activity-centric***) a servi de base à la mise en place du standard *Business Process Model and Notation* (BPMN) par le BPMI (*Business Process Management Initiative*) [106] en 2005 et du langage de sérialisation, format d'échange de données entre workflows, le *XML Process Definition Language* (*XPDL*) [107] par le *Workflow Management Coalition* (*WFMC*) en 2002.

Cette première approche de modélisation centrée sur le processus et les activités qui le constituent marginalise le temps. D'où le développement d'un deuxième axe ou groupe de langages et de formalismes, qui sont plutôt basés sur la discrétisation du temps et des événements. Ceci est dû à la prédominance du temps dans l'ordonnancement et la synchronisation des systèmes complexes dont fait partie le workflow. Dans cette deuxième catégorie, nous classons le standard *Language of Temporal Ordering Specification* (*LOTOS*) [108] qui se base sur le *Communicating Sequential Processes* (CSP). Il s'agit du CSP introduit vers 1980 par Robin Milner [109] et qui modélise l'interaction des systèmes. Le programme CSP intègre l'algèbre de processus, le Pi-calcul, les RdP (Réseaux de Petri) et la norme ISO LOTOS. Cette deuxième famille quant à elle propose une formalisation mathématique des systèmes concurrents. Elle met un accent sur l'aspect clé qu'est la communication entre processus, la description, l'analyse, la vérification et la validation des propriétés qualitatives des systèmes concurrents, à l'instar des ***deadlocks***, les ***lifelocks, la starvation***, etc.

Dans ce chapitre, nous proposons un modèle dynamique du flux logistique basé sur la norme ISO LOTOS afin de spécifier le workflow. Ce modèle nous permet d'analyser les propriétés quantitatives et qualitatives du flux. Aussi, nous présentons la spécification de la plateforme de collaboration et de partage d'information entres les acteurs impliqués dans un flux logistique en tant que système communicant avec des workflows. La première section consiste en un bref rappel des concepts fondamentaux de la spécification LOTOS, suivi par le modèle de workflow proposé. Le chapitre se termine par une expérimentation et l'étude d'un cas d'utilisation, et une autocritique du modèle afin de déceler ses avantages et ses insuffisances.

## 3.2. Concepts de base de E-LOTOS

### 3.2.1. Introduction du temps

Dans un système communicant, le temps peut être continu ou discret. Dans un domaine de temps continu, le temps peut prendre n'importe quelle valeur. Il existe par contre des systèmes dans lesquels l'ordre des événements est discontinu dans le temps, et parfois ces événements peuvent apparaître de façon ponctuelle. Cette deuxième catégorie de systèmes est classée dans le groupe des systèmes à événements discrets. C'est le cas des flux logistiques où l'ordre d'occurrence des événements est discret (arrivée des commandes clientes, dates effectives de livraison, départs et arrivées de marchandises, ...). En effet, l'ordre des événements ne suit pas une loi de probabilité connue ou prédéfinie à l'avance. Dans LOTOS et particulièrement dans E-LOTOS (Enhanced LOTOS), le temps est introduit via trois concepts fondamentaux:

- *when*: cette notion n'est pas un élément de la syntaxe, mais permet de spécifier quand est-ce que l'événement a eu lieu, et quand est-ce qu'une action doit être exécutée par le processus.

- *time*: la notion de temps (*time*), associée à un domaine temporel (*D*), est un ensemble dénombrable de variables temporelles vérifiant les axiomes suivants: *soient t1 et t2 dans D,*

$$t_1 + t_2 = t_2 + t_1 : Propriété\ de\ commutativité$$
$$t_1 + 0 = 0 + t_1 : 0\ est\ élément\ neutre\ dans\ D\ par\ rapport\ à\ +$$
$$t_1 + (t_2 + t_3) = (t_1 + t_2) + t_3 : Propriété\ d'associativité$$
$$si\ t_1 + t = t + t_2\ alors\ t_1 = t_2\ : Unicité\ du\ temps\ discret$$
$$t_1 \leq t_2 \Leftrightarrow \exists t, t_1 + t = t_2 : définit\ un\ ordre\ total\ dans\ le\ domaine\ D$$

- *Wait(d):* l'instruction *wait(d)* permet à un processus de laisser passer le temps pendant *d* unités de temps.

### 3.2.2. Le processus

Le processus dans la spécification E-LOTOS est la déclaration d'un comportement du système, en précisant le nom, la liste des paramètres qui sont des variables d'environnement du système, les canaux de communication du processus (système) avec d'autres processus, et éventuellement les exceptions que le processus peut gérer. La synthaxe d'un processus dans E-LOTOS peut être donnée par :

***Process Pname*** *[[$G_1$[ :$T_1$], $G_2$[ :$T_2$], ..., $G_n$[ :$T_n$] ]]*
*[( [in |out]$V_1$ :$T'_1$, [in |out]$V_2$ :$T'_2$, ..., [in |out]$V_m$ :$T'_m$,)]*
*[raises[ $X_1$[ : $T''_1$], $X_2$[ :$T''_2$], ... , $X_n$[ :$T''_n$] ]]*
***is processBody***
***Endproc***

Où

- ***Pname*** est le nom du processus, son identifiant.
- *$G_1$[ :$T_1$], $G_2$[ :$T_2$], ..., $G_n$[ :$T_n$]* est la liste des canaux utilisés par le processus pour communiquer avec d'autres processus ou son environnement.
- *$T_i$* est le type du canal *$G_i$* (porte/gate), c'est à dire le type de valeur qui peut être communiquée via ce canal.
- *[([in |out]$V_1$ :$T'_1$, [in |out]$V_2$ :$T'_2$, ..., [in |out]$V_m$ :$T'_m$,)]* est la liste des variables d'environement du processus.
- *$T'_j$* est le type de la variable *$V_j$*
- *$T''_j$* est le type de valeur associée à l'exception *$X_j$* gérée par le processus
- *ProcessBody* est le code à exécuter par le processus, la liste des tâches qui constituent le *process*.

Dans ce qui suit, nous présentons l'exemple de spécification LOTOS de déclaration d'un processus collaboratif d'achat-vente d'ordinateurs. Dans cet exemple, nous considérons un fournisseur de matériel informatique, qui produit et vent des ordinateurs et des accessoires informatiques à ses clients. Le client achète au fur et à mesure que sa demande augmente et que son stock diminue. Le système peut être représenté par le diagramme ci-dessous.

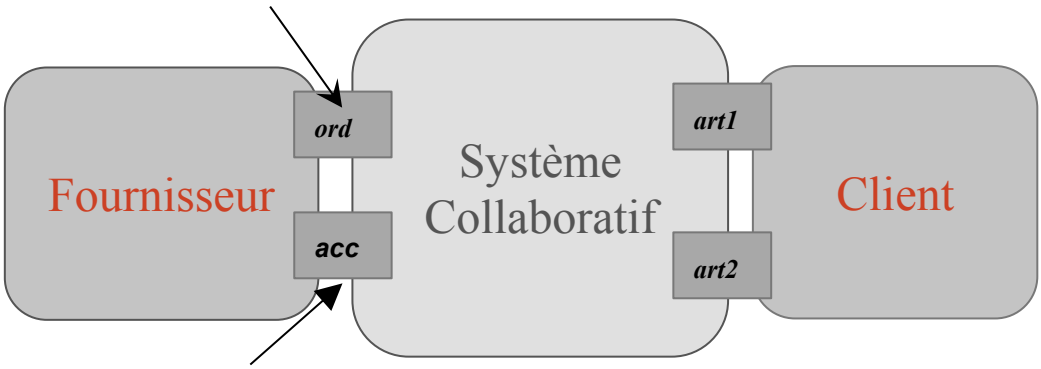


Figure 33. Exemple de système collaboratif dans la chaîne logistique

Dans l'exemple de la figure 33, le système collaboratif dispose de quatre ports ou gateways (ord, acc, art1, art2). Les ports ***ord*** et ***acc*** sont utilisées par le fournisseur pour transmettre respectivement la disponibilité des lots/ batchs d'ordinateurs ou d'accessoires fabriqués. Les canaux art1 et art2 sont utilisés par le système client pour interroger le système sur la livraison d'un lot d'ordinateurs ou d'un lot d'accessoires. A ce niveau de démonstration, le système collaboratif se contente de jouer le rôle de médiateur, de plateforme d'échange de messages entre les deux systèmes. La syntaxe LOTOS correspondant à la description de ce processus collaboratif est la suivante :

```
Process CollaborativeSupplyChain is
    hide ord: Ordinateur, acc: Accessoire, art1: Ordinateur, art2: Accessoire in
        Conc
            Fournisseur [ord,acc] ()
            | [ord,acc] |
            SystemCollaboratif [ord, acc, art1, art2] ()
            | [ord,acc] |
            Client [art1,art2] ()
        endConc
    Endhide
Endproc
```

### 3.2.3. Gateway et action

Une des notions importantes de LOTOS est la notion de "*action de port de communication*". Une action est une interaction entre deux ou plusieurs processus communicant à travers un canal de communication. Les canaux de communication sont symbolisés dans LOTOS par des portes ou gates.
La syntaxe de déclaration des canaux de communications est la suivante :

***Gate*** [(*Pattern1*)] [@*Pattern*2] [ [*Expression*] ]

Où ***Gate*** désigne le nom du canal de communication entre les processus, ***Pattern*1** et ***Pattern*2** sont des expressions d'écriture ou de lecture de messages sur le canal de communication, ou une combinaison des deux. ***Expression*** est une formule booléenne qui tient lieu de conditions sous lesquelles la communication doit être possible. Un processus peut ainsi lire ou écrire sur un port soit des valeurs des variables d'environnement du processus, soit des constantes. Le flux d'information ainsi échangé est matérialisé par la partie ***Pattern*1**.

Reprenons l'exemple du système collaboratif de la figure 33. Le fournisseur communique via les ports ***ord*** et ***acc*** avec le système et le client communique à son tour par les ports ***art1***, et ***art2***. Les ports **ord** et **acc** permettent d'écrire des messages concernant des ordinateurs, et les autres accessoires, de même que la lecture des notifications du client parallèlement. Les ports ***art1***, et ***art2*** jouent le même rôle du côté du client, et lui permettent

d'échanger des messages avec le fournisseur via le système collaboratif. Supposons que le fournisseur veut publier l'information "Lots N°3 disponible" pour une commande d'ordinateurs.
Cette information est spécifiée par la syntaxe LOTOS suivante:

```
ord(! "Lot N°3 disponible")      -- Le lot numéro 3 est disponible
art1(? msgOrd)                   -- réception d'un message du fournisseur
acc(! 45)                        -- 45 lots d'accessoires prêts pour expédition
art2(? msgAcc)                   -- réception d'un message du fournisseur
ord(! "Fin de cycle")            -- Fin d'un cycle de production
art2(? msgOrd)                   -- réception d'un message du fournisseur
```

La deuxième instruction indique la réception d'un message par le client. Ce message est stocké dans la variable *msgOrd* qui joue le rôle d'une boîte à lettre tampon. La troisième instruction indique l'écriture du nombre de lots d'accessoires prêts pour expédition. Outre ces écritures et lectures simplistes sur les ports de communication, il existe des opérations plus complexes, comme l'écriture de vecteurs, les écritures et lectures conditionnées par les valeurs des variables, ou des réceptions de messages d'un certain type uniquement par le processus. Ces éléments sont autant d'astuces qui permettent à plusieurs processus de communiquer en échangeant des messages ou des valeurs.

```
art1(qtDispo ?dispo, qtExpedie ?exp))    -- lecture des champs d'un tableau
```

Le tableau de message contient la quantité d'ordinateurs disponible chez le fournisseur et la quantité déjà expédiée.

### 3.2.4. Expression des contraintes temporelles

Il est important dans un système complexe de préciser les contraintes temporelles. En effet les timer de déclenchement des tâches dans une chaîne logistique, les délais d'exécution de certaines opérations sont des exemples concrets de telles situations. Pour ce faire la spécification LOTOS propose la syntaxe ***Gate*** [(*Pattern1*)] [@Pattern2]. La deuxième expression *@Pattern2* du port ou du canal de communication permet d'exprimer des conditions booléennes sur le temps. Ainsi, l'action ne pourra se réaliser que si ces conditions temporelles sont satisfaites.

Exemple:

```
art1("Article hors delai" ?dueDate)@ ? dueDate [ dueDate < 5])
```

Le client reçoit le message *"Article hors délai"* si la valeur de la date limite (*dueDate*) est dépassée de cinq jours.
Il existe deux formes particulières de ce pattern:

***? time [timeCondition]:*** ce pattern permet d'associer la variable temporelle time à l'action en cours, plus précisément au temps auquel cette action a commencé effectivement. Ensuite nous pouvons exécuter l'action suivant les restrictions imposées par la condition [*timeCondition*] qui est une formule booléenne.

***! time :*** ce pattern permet de comparer l'heure de démarrage de l'action en cours et la valeur de la variable ***time.*** Si l'heure de démarrage de l'action est égale à la valeur *time*, l'action sera exécutée, sinon rien ne se passe.

### 3.2.5. Synchronisation de processus dans E-LOTOS

E-LOTOS propose plusieurs éléments de synchronisation de processus. Parmi ces éléments nous citons la composition séquentielle, la récursivité, l'opérateur de sélection, le choix non déterministe, l'action interne au processus. Nous introduisons dans ce qui suit chacun de ces éléments.

- **L'opérateur de sélection**

L'opérateur de sélection permet à un point de branchement de choisir entre deux processus celui qui sera exécuté. L'opérateur de sélection est l'équivalent du XOR de la logique booléenne, à la seule différence qu'il s'applique aux processus.

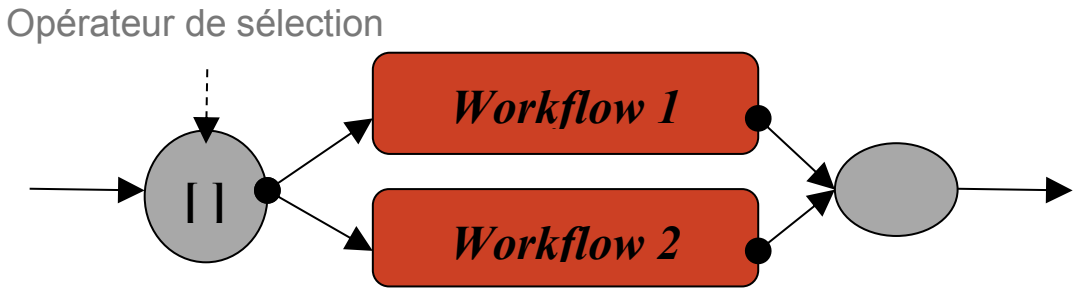


Figure 34. Opérateur de sélection de processus

L'opérateur de sélection exécute le premier processus ou le deuxième selon les actions en cours et les valeurs des variables d'environnement. Si l'action est destinée au processus 1, alors il sera exécuté; sinon si l'action est destinée au processus 2, ce dernier sera exécuté. Exemple:

***Process*** *CoPackaging* is
***hide*** *perf: Parfum, sug: Sugar, perfOut: CustomPerfum, sugarOut: customSugar* ***in***
*perfumCustompackaging* ***[****perf, perfOut****] ()***
***[ ]***
*sugarCustompackaging* ***[****sug, sugarOut****] ()***
***Endhide***
***Endproc***

Le processus de custom packaging prend en entrée soit du sucre ou du parfum; s'il reçoit du parfum, le sous-processus de conditionnement du parfum sera exécuté. Dans le cas contraire, c'est le sous processus du conditionnement à façon du sucre qui sera exécuté.

- **La composition séquentielle**

L'opérateur de composition séquentielle permet d'exécuter des processus à la chaîne. Lorsque le premier processus termine son exécution, le deuxième commence son exécution, et ainsi de suite pour le troisième jusqu'au dernier.

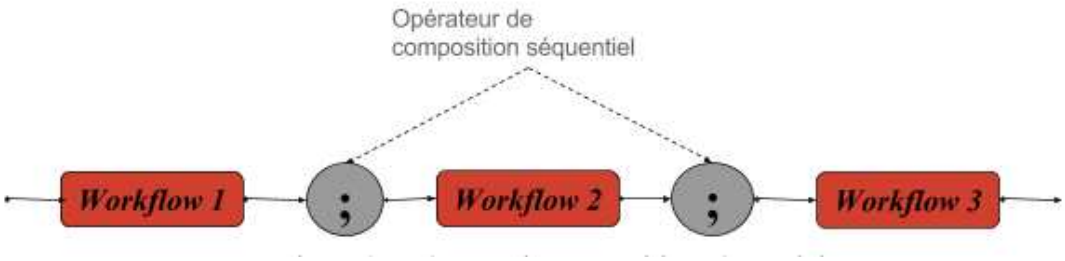


Figure 35. opérateur de composition séquentiel

Dans la figure 35, le workflow s'exécute dans la séquence indiquée. En suivant l'ordre imposé par l'opérateur de composition séquentielle.

Exemple:

```
Process sugarCustompackaging is
    hide sugIn: Sugar, sugarOut: customSugar , msg: data in
        msg(! "Sucre en poudre disponible ")        -- disponibilité du sucre
        ;
        sugIn(? stockDispo)                         -- réception du sucre
        ;
        msg(! "Opération de custom packaging en cours")
        ;
        sugarOut(? stockForDelivery)        -- stockage du sucre reconditionné
        ;
        msg(! "Fin de conditionnement")     -- Fin de conditionnement du sucre
    Endhide
Endproc
```

Les actions de ce processus sont exécutées dans l'ordre séquentiel. Le sucre est d'abord réceptionné, puis stocké, ensuite reconditionné. L'envoi du message "fin de conditionnement" marque la fin du processus séquentiel.

- **La composition parallèle**

La chorégraphie est une des méthodes de synchronisation de processus qui diffère de l'orchestration. La première gère la synchronisation de processus parallèles en considérant les processus comme étant des unités de traitement ou d'exécution indépendantes et autonomes. Ces unités échangent des messages en vue d'assurer la cohérence de l'ensemble des traitements. Contrairement à l'orchestration qui centralise les instructions d'ordonnancement dans un ordonnanceur (chef d'orchestre) qui est chargé de planifier l'exécution des processus afin d'assurer la cohérence de l'ensemble.

Dans E-LOTOS, la synchronisation des processus qui s'exécutent en parallèle est assurée par un opérateur spécial, l'opérateur de synchronisation parallèle.

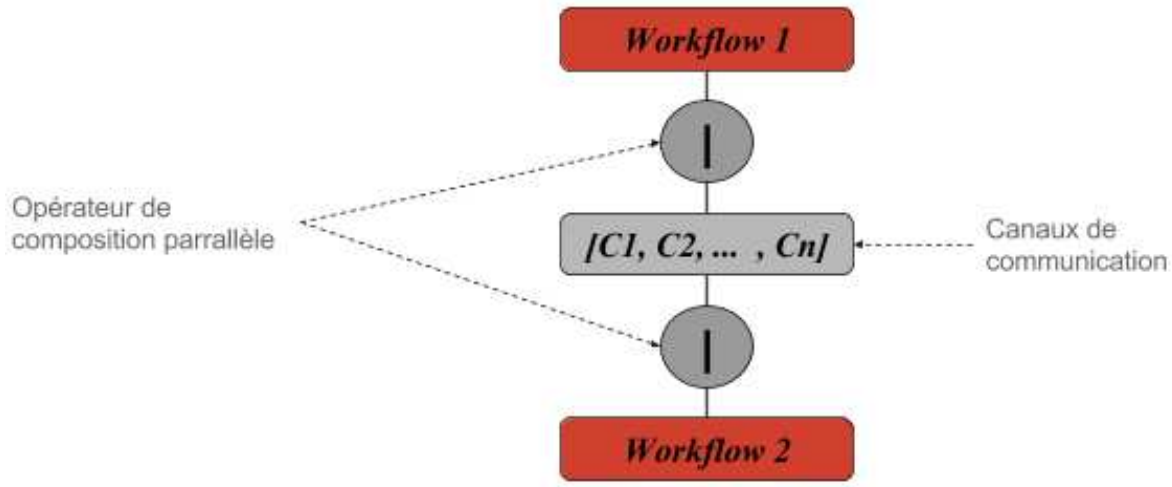


Figure 36. Opérateur de composition parallèle de workflows

Sur la figure 36, les workflow 1 et 2 s'exécutent en parallèle de manière autonome. La synchronisation des deux workflow est assurée par l'opérateur de composition parallèle et les canaux/ports de communication via lesquels les workflow s'échangent les messages et les signaux. Les ports de communication permettent d'exécuter les actions communes, qui doivent être exécutées simultanément. Si un workflow est prêt à exécuter une action de synchronisation et que l'autre ne l'est pas, alors le premier processus attend le deuxième afin d'exécuter les actions en parallèle. Lorsque les deux processus communiquent par un port, l'action de synchronisation n'est possible que si les conditions ou patterns (paragraphes précedent) sont satisfaites. Il faut noter tout de même que l'opérateur de composition parallèle se décline en cinq versions différentes dans E-LOTOS.

- **L'inaction et le blocage de processus**

L'inaction et le blocage consistent à empêcher l'exécution d'un processus. L'inaction est matérialisée par l'opérateur stop qui permet de faire attendre le processus pendant un certain temps.

Exemple:

***stop*** || *wait(5)* **;** *ord(!9)*

L'instruction *ord(!9)* ne s'exécute qu'après que le processus ait attendu 5 unités de temps. C'est à ce moment que le canal de communication sera disponible.

***block*** || *wait(8)* **;** *ord(!4)* L'instruction block permet de bloquer le processus et de laisser passer le temps.

- **La recursivité**

Un processus récursif est un processus auto-appellant. La récursivité permet de faire des traitements répétitifs, parfois imbriqués les uns dans les autres, avec l'exécution du même processus soit avec les mêmes variables d'environnement, soit avec des valeurs différentes des variables d'environnement. La récursion permet aussi d'exprimer qu'un processus continue de s'exécuter dans le temps sans interruptions.

Exemple

***Process*** *computerDelivery* ***[****ord: ordinateur, Item: data****] is***

***hide*** *ord: Ordinateur Item: Ordinateur,* ***in***

*Var v: Ordinateur in*

***ord(? v) ; Item (! v) ; computerDelivery [ord, Item]*** *-- recursion du workflow*

***Endhide***

***Endproc***

Dans l'exemple qui suit, le processus de livraison des ordinateurs est continu dans le temps. Dès qu'un lot est acheminé et réceptionné, un autre lot est en cours et ainsi de suite. La récursivité exprimée par l'appel du processus à l'intérieure de lui même pendant son exécution exprime ce comportement.

- **L'action interne**

L'action interne dans E-LOTOS permet d'exprimer des comportements du système qui ne sont pas observables, ou que l'on aimerait masquer à son environnement. L'action interne permet aussi d'exprimer des cas de compensation de processus. Il s'agit des cas où le processus est censé exécuter une action et finalement l'action n'a pas lieu ou n'est pas exécutée comme prévue. L'action interne est urgent, ça veut dire qu'elle s'éxécute avant tous les autres procesus.

- **Choix déterministe et choix non déterministe**

E-LOTOS étant destiné à la spécification de systèmes temporelles, il est important d'éviter certaines configurations qui conduisent au blocage du système, ou qui l'entrainent dans une boucle infinie. Ces cas de figure sont parfois dus au non déterminisme du

processus, que nous pouvons déceler dès la phase de spécification. Voici quelques cas où le processus est non déterministe.

***customPakaging ≡***
***(sucreEnPoudreDisponible;***
***customiserEnPackDe6())[](parfumDisponible;embouteiller())***

Ici l'opérateur de sélection permet de faire un choix déterministe, soit le sucre en poudre est disponible, dans ce cas le processus d'empaquetage en pack de 6 se déclenche, soit c'est plutôt du parfum qu'on reçoit à l'entrée de la chaîne et c'est le processus de mise en bouteille qui est exécuté dans ce cas. L'ensemble est donc déterministe.

***customPakagingV1 ≡***
***(sucreDisponible; customiserEnPackDe6())***
***[]***
***(sucreDisponible; ensacherPar500gr())***

Dans cette version du processus, le choix est non déterministe car le même événement déclenche deux actions différentes par l'opérateur de sélection. Le sucre étant disponible, le système ne saura s'il faut transformer en pack de 6 ou bien mettre dans des sachets de 500 grammes. Le choix est donc non déterministe, entraînant le blockage du processus.

***customPakagingV2 ≡***
***sucreDisponible; (customiserEnPackDe6() [] ensacherPar500gr())***

Dans cette nouvelle version du processus, la disponibilité du sucre en poudre déclenche l'opérateur de sélection qui bloque le système en attente du choix de l'opérateur. Ce dernier peut choisir entre le processus de mise en pack de 6 ou bien le processus de mise en sachets de 500 grammes. Cette nouvelle version du processus est donc déterministe.

- **L'opérateur de synchronisation**

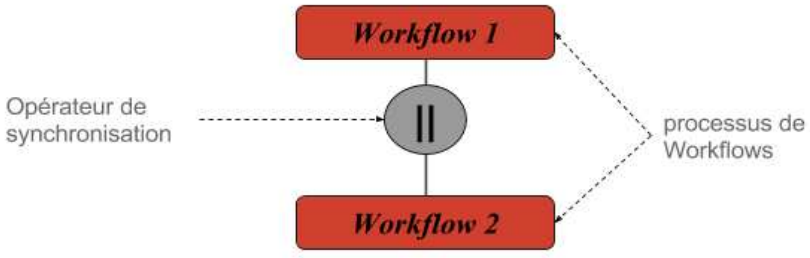


Figure 37. Opérateur de synchronisation de workflows

Cet opérateur permet la synchronisation de deux ou plusieurs processus sur toutes les actions communes observables. Les processus communiquent à travers leurs canaux partagés, en échangeant des messages et des signaux. Cet opérateur est équivalent à

l'opérateur de composition parallèle, à la seule différence que les canaux ou ports de communication sont omis, donc sous en entendus.

- **Opérateur d'entrelacement (Interleaving operator)**

L'opérateur d'entrelacement permet d'exécuter les opérations de merging entre les processus qui s'exécutent en parallèle. Il peut être utilisé pour synchroniser deux processus dont la liste des canaux de communication est vide. Dans ce cas l'opérateur va associer les deux processus.

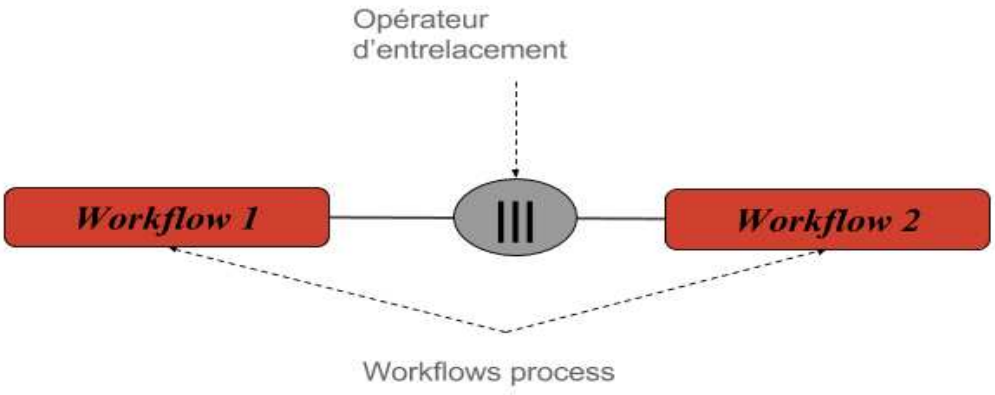


Figure 38. Opérateur d'entrelacement de processus

Exemple:

***Process*** *CustomPackaging* [*sugIn: data, sugarOut: data , perfumIn:data, perumOut: data, Msg: data*] ***is***

***var*** *y1: data, y2: data* ***in***
*msg(! "Démarrage du processus de custom packaging ...") ;*
*sugIn(? y1 : data); -- réception du sucre en poudre*
***(****perfumIn(y2: data)*
*|||* *-- merging des deux processus*
***(****msg(! "Transformation du sucre en pack de 6 ..."); sugarOut(!y1)****)***
***);***
***(****msg(! "Mise en bouteille du parfum ..."); perumOut(!y2)****);***
*msg(! "Fin de conditionnement") -- Fin de reconditionnement des produits*
***Endvar***
***Endproc***

Dans cet exemple, le processus a en entrée soit du sucre à transformer en pack de 6, soit du parfum à embouteiller. Le processus peut bien réceptionner et embouteiller du parfum après avoir reçu et conditionner du sucre.

## 3.3. Modélisation de la dynamique du workflow avec E-LOTOS

Après avoir étudié les concepts fondamentaux de E-LOTOS, nous utilisons la syntaxe de cette spécification pour décrire la dynamique d'un système logistique en tant que système collaboratif fortement assujetti à une évolution temporelle.
Le but de cette section est de simuler le flux logistique comme étant un système complexe communicant, en prenant en compte les contraintes temporelles auxquelles sont assujetties les processus et les objets logistiques. Le système étant orienté vers des besoins de collaborations des acteurs logistiques, nous portons un intérêt particulier sur la synchronisation des actions et des messages entre ces opérateurs afin d'acheminer le flux de marchandises de son point de départ jusqu'à sa destination finale. Nous partons du modèle générique du workflow que nous spécialisons par la suite dans un exemple concret de plateforme de collaboration.

### 3.3.1. Le workflow en tant que système communicant

Pour illustrer l'application du formalime E-LOTOS à la spécification des systèmes logistiques, nous considérons l'exemple d'un processus collaboratif de livraison de containers. Pour simplifier la tâche, nous supposons que les containers contiennent les articles (article 1, article 2, article 3) produis par des fournisseurs différents, à destination des clients clt1, clt2 en attente de ces produits. Les types de produits article 1, article 2, sont destinés aux clients de type client1 et le type de produit article 3 aux clients de type client2. La collaboration consiste à faire communiquer les fournisseurs des produits (articles) avec les clients qui attendent ces produits au bout de la chaîne. Une autre exigence est d'utiliser le même flux de containers pour effectuer des livraisons groupées d'articles, et non au fil de l'eau pour chaque client. Enfin le système devra notifier chaque client de la disponibilité de ses produits et de la quantité disponible.


Figure 39. Schéma du système de communication entre les fournisseurs et les clients

La disponibilité des articles et leur quantité sont publiées via les ports respectifs Art1, Art2, et Art3 comme illustré par la figure 39. Les clients communiquent avec le système via les canaux clt1 et clt2. Ainsi, le système joue le rôle de médiateur entre les clients et les fournisseurs.

**Spécification du système par E-LOTOS**

```
Process shippingContainer [ Art1: data, Art2:data, art3:data, clt1:data, clt2:data] is
    Var x1,: data, x2:data, x3: data, x4, data in

        ((Art1(?x1) ; clt1("Article 1 disponible chez le fournisseur !"); clt1(!x1)
            []
        Art2(?x2) ; clt1("Article 2 disponible chez le fournisseur !"); clt1(!x2)
            []
        Art3(?x3) ; clt2("Article 3 disponible chez le fabricant !"); clt2(!x3)
        )
        |||
        (Art1("Lot client 1 prêt pour expedition !") [] Art1("Lot client 2 prêt pour
        expedition !"))
        |||
        ( closingAndZingingTheContainer[Art1, Art2, Art3, clt1, clt2]();
        );
        |
        shipping[Art1, Art2, Art3, clt1, clt2]();
        shippingContainer [ Art1: data, Art2:data, art3:data, clt1:data, clt2:data]
    endvar
endproc
```

Le système commence par la réception des articles par lots, qu'il stocke dans un container. L'opérateur de sélection permet de recevoir les articles indifféremment de leur ordre d'arrivé. Ceci dit, le système peut recevoir les articles du fournisseur d'articles de type article 3 bien après qu'il ait reçu les articles de type 2 ou 1. L'opérateur d'entrelacement permet de merger tous les lots reçus et les différents messages de confirmation des fournisseurs avant de procéder à l'opération de fermeture du container. La récursivité exprimée par l'appel du système joue ici le rôle d'auto-exécution du processus. Le système va continuer ainsi son exécution en recevant de nouveaux lots d'articles.

### 3.3.2. Modèle générique du workflow

**Formalisation mathématique via les automates états / transitions**

Le but de ce paragraphe est de définir un modèle générique de flux logistique, qui soit facilement transposable en E-LOTOS. Dans ce modèle, il est question de définir de façon

générique le workflow process tout en intégrant les différents aspects de notre problématique. Nous proposons la définition ensembliste suivante pour le modèle générique du workflow :

$P = < S,\ G,\ M,\ C,\ R,\ L >$ comme suit :

- $S = \{s_1,\ s_2, s_3,\ \cdots, s_m,\}$ est l'ensemble des sous processus ou stages du flux principal. S contient des états du flux. Ces états sont les différentes étapes par lesquelles un élément du flux est censé passer jusqu'à la fin du processus. S peut aussi être vu comme l'ensemble des tâches qui sont exécutées sur l'Item pendant son passage dans le processus, s'il s'agit d'un workflow de transformation.

- $G = \{g_1,\ g_2, g_3,\ \cdots, g_k,\}$ est un ensemble de K guards ou conditions. Les Guards sont des formules booléennes qui conditionnent le déclenchement d'un stage ou d'une tâche du flux. Les conditions (guards) peuvent être combinées pour déclencher un seul stage.

- $M = \{m_1,\ m_2, m_3,\ \cdots, m_m\}$ est un ensemble d'objectifs ou milestone à atteindre par le flux principale ou ses sous flux. Un milestone est atteint par une tâche ou stage. Il est à noter qu'un stage peut permettre l'accomplissement de plusieurs milestones.

- $C = \{c_1,\ c_2,\ c_3,\ \cdots, c_m,\}$ désigne l'ensemble des canaux de communication inter-processus, inter-stages. Les canaux ou ports de communication permettent aux sous processus de communiquer entre eux en s'échangeant des événements ou des messages. Il y a dans ces canaux ceux qui permettent au processus de communiquer avec son environnement extérieur.

- $R = \{< s_i \rightarrow g_j >\} \cup \{< m_k \rightarrow s_t >\}$ est un ensemble des paires qui décrivent la relation de déclenchement des stages à partir des guards. Cette relation spécifie que le guard $g_j$ déclenche le stage $s_i$ dans un premier temps, et de même précise que le stage $s_t$ permet d'atteindre le milestone $m_k$.

- $L = \{< S_i,\ s_j,\ w_{i,j} >\}$ est une relation d'ordre partielle entre les sous processus ou stages qui définissent le cycle de vie du processus. L'ensemble L est formé de triplets $< s_i,\ s_j,\ w_{i,j} >$ qui signifie que le sous processus ou stage $s_j$ est exécuté après que le sous processus $s_i$ ait terminé son exécution. Ou plus généralement spécifie une étape business entre les deux sous processus.

- $w_{i,j} = \begin{cases} 1,\ si\ \ s_i\ est\ exécuté\ avant\ \ s_j \\ 0,\ si\ \ s_i\ est\ un\ stage\ de\ \ s_j \\ -1,\ si\ \ s_i\ et\ \ s_j\ \ s'exécutent\ en\ parrallèle \end{cases}$

### 3.3.3. Spécification du modèle générique du workflow avec E-LOTOS

Dans cette spécification, le workflow principal est décomposé en différents sous processus ou stages le constituant ***(stage1, stage2, …, stagem).*** Les stages ou sous workflow sont synchronisés par des opérateurs de synchronisation ***(operator1, operator2, … operator (m-1)).*** Les différents processus communiquent via des canaux de communication qui peuvent être de type variés, en utilisant l'expression $c_{ij}[:T_{ij}]$ qui signifie que le canal de communication $c_{ij}$ est de type $T_{ij}$, et dans ce cas le processus ne peut accepter que ce type de variable en entrée de ce canal.

Pour mieux comprendre cette spécification et faire une évaluation des performances, nous décrivons un scénario concret de collaboration entre flux logistiques. Ce scénario est présenté dans le paragraphe suivant.

***Process WorflowProcess*** $[[C_1[:T_1], C_2[:T_2], \dots, C_n[:T_n]]]$
$[([in]\ G_1 : T'_1, [in]\ G_2 : T'_2, \dots, [in]\ G_k : T'_k)]$
$[([out]\ M_1 : T''_1, [out]\ M_2 : T''_2, \dots, [out] M_m : T''_m)]$
$[raises\ [\ X_1[:T''_1]\ ,\ X_2[:T''_n], \dots,\ X_n[:T''_n]\ ]]$

***is***

***Stage1*** $[[C_{11}[:T_{11}], C_{12}[:T_{12}], \dots, C_{1n}[:T_{1n}]]]$
$[([in]\ G_{11} : T'_{11}, [in]\ G_{12} : T'_{12}, \dots, [in]\ G_{1k} : T'_{1k})]$
$[([out]\ M_{11} : T''_{11}, [out]\ M_{12} : T''_{12}, \dots, [out] M_{1m} : T''_{1m})]$
$[raises\ [\ X_{11}[:T''_{11}]\ ,\ X_{12}[:T''_{1n}], \dots,\ X_{1n}[:T''_{1n}]\ ]]$
***Stage1Body***

*operator1*

***Stage2*** $[[C_{21}[:T_{21}], C_{22}[:T_{22}], \dots, C_{2n}[:T_{2n}]]]$
$[([in]\ G_{21} : T'_{21}, [in]\ G_{22} : T'_{22}, \dots, [in]\ G_{2k} : T'_{2k})]$
$[([out]\ M_{21} : T''_{21}, [out]\ M_{22} : T''_{22}, \dots, [out] M_{2m} : T''_{2m})]$
$[raises\ [\ X_{21}[:T''_{21}]\ ,\ X_{22}[:T''_{2n}], \dots,\ X_{2n}[:T''_{2n}]\ ]]$
***Stage2Body***

*operator2*

…

*operator(m-1)*

***Stagem*** $[[C_{m1}[:T_{m1}], C_{m2}[:T_{m2}], \dots, C_{mn}[:T_{mn}]]]$
$[([in]\ G_{m1} : T'_{m1}, [in]\ G_{m2} : T'_{m2}, \dots, [in]\ G_{mk} : T'_{mk})]$
$[([out]\ M_{m1} : T''_{m1}, [out]\ M_{m2} : T''_{m2}, \dots, [out] M_{mm} : T''_{mm})]$
$[raises\ [\ X_{m1}[:T''_{m1}]\ ,\ X_{m2}[:T''_{mn}], \dots,\ X_{mn}[:T''_{mn}]\ ]]$
***StagemBody***

***Endproc***

## 3.4. Expérimentation de la spécification du workflow

### 3.4.1. Description de la plateforme collaborative

Dans ce paragraphe nous décrivons la plateforme collaborative avec des diagrammes d'états transitions suivi de la spécification E-LOTOS de ces Workflows. Le flux considéré est celui de planification et de livraison des marchandises à grande échelle, impliquant les fournisseurs, les transporteurs, les clients finaux qui passent les ordres de commandes. Les commandes sont de plusieurs types de produits et les fournisseurs sont variés tout de même. Le système est semblable à un bus collaboratif en mode Cloud basé sur des événements.

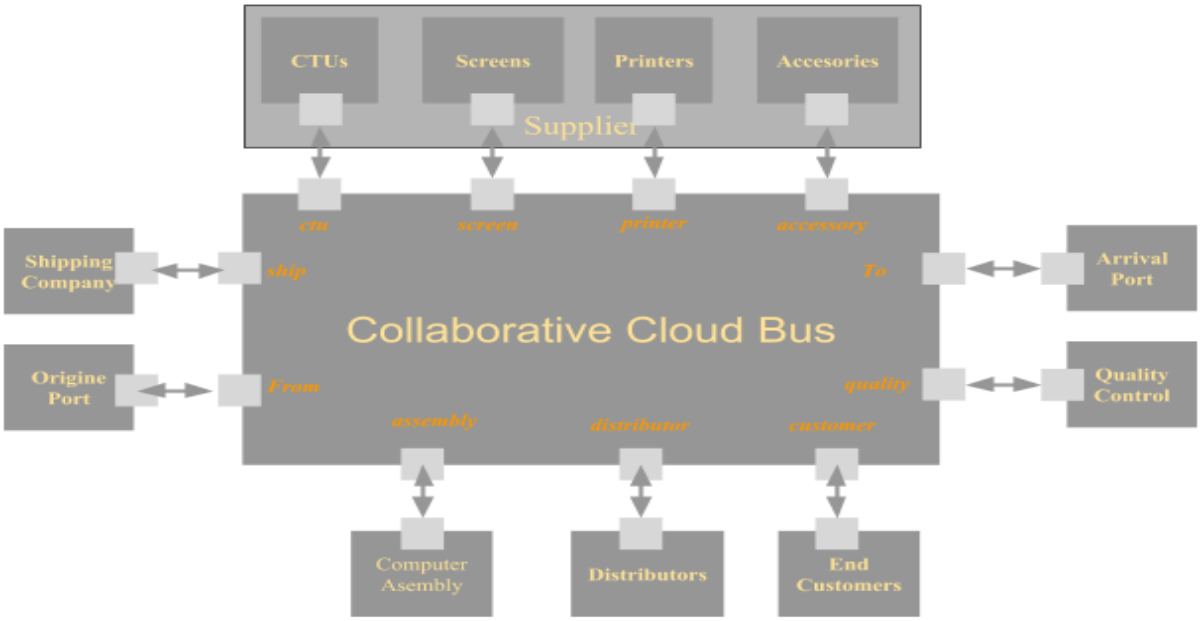


Figure 40. Chaîne logistique communicante et collaborative

Le système à modéliser est un bus collaboratif en mode Cloud comme illustré par la figure 40. Il permet à des fabricants de composants d'assembler des ordinateurs, de les redistribuer, et de revendre aux clients finaux. L'acheminement des articles se fait par une compagnie de transport maritime qui assure le transport des marchandises du port d'origine vers le port de destination. Les autorités de régulation effectuent le contrôle qualité sur les marchandises transportées.

- **CTUs manufacturer:** le fournisseur d'unités centrales envoie des lots à la demande, soit aux distributeurs, au concessionnaire, soit directement au client final. Le sous-système chargé de la fourniture d'unités centrales n'accepte que des commandes d'unités centrales. Toute autre commande relève une exception qui notifie au client ayant passé la commande que l'ordre ne sera pas prise en compte.

- **Distributeurs:** le client passe des ordres de commandes de façon périodique au fournisseur. Les commandes d'UCT se font tous les quinze jours, et la quantité commandée varie en

fonction du stock restant. Le stock ne doit pas dépasser un nombre maximal *maxCpu* fixé à 200 articles, et ne doit aller en deçà d'un nombre minimal *minCpu* fixé à 50 articles. Le client peut commander aussi des ordinateurs, des imprimantes, des écrans, des accessoires. Un ordre de commande est décrit par une date de commande, une date de livraison souhaitée et une date de livraison effective. Lorsque la date de livraison souhaitée est dépassée, le système envoie une notification au fournisseur pour lui rappeler que l'article n'a toujours pas été reçu. Si la date de livraison effective dépasse la date prévisionnelle de plus de 3 jours, le système enverra une alerte au fournisseur pour le prévenir des pénalités de retard. Lorsque le client reçoit une commande, il l'enregistre et incrémente le stock de produit ainsi livré. Les bilans des stocks disponibles sont affichés à la fin de chaque journée, et envoyés au fournisseur pour information. Pour finir, le système attend l'accusé de réception du fournisseur qui confirme que l'ordre de commande a été bien reçu.

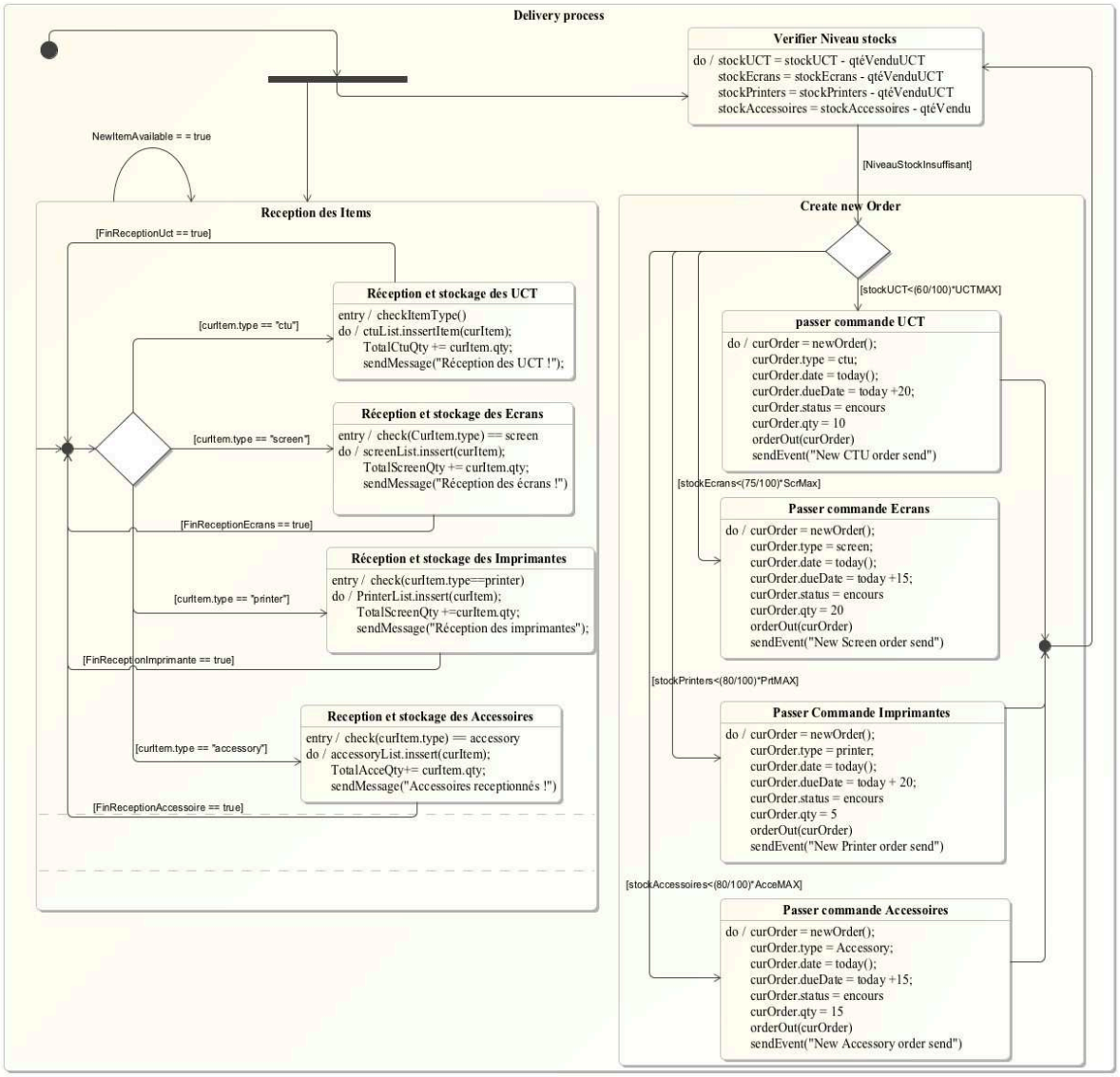


Figure 41. diagramme d'états transitions SysML du processus de distribution

**Spécification LOTOS du processus de distribution**

```
Process Distributor [ orderOut: Order, ItemIn: Item, DistInfo: Info] is
        Var curOrder: Order, msg:Info, curItem:Item, ctuList: Ctu, screenList: Screen,
                printerList: Printer, acceList:Accessory in
While (true)
        (ItemIn(?curItem); -- mode réception de commande
                Case curItem ->type == ctu
                        ctuList - >head := curItem;
                        TotalCtuQty := TotalCtuQty + curItem -> qty;
                        DistInfo(?"Reception des UCT !");
                Case curItem ->type == screen
                        screenList - >head := curItem;
                        TotalScreenQty := TotalScreenQty + curItem -> qty;
                        DistInfo(?"Reception des Ecrans !");
                Case curItem ->type == printer
                        printerList - >head := curItem;
                        TotalPrinterQty := TotalPrinterQty + curItem -> qty;
                        DistInfo(?"Reception des Imprimantes !");
                Case curItem ->type == accessory
                         acceList - >head := curItem;
                        TotalAcceQty := TotalAcceQty + curItem -> qty;
                        DistInfo(?"Reception des Accessoires !");
        )
        []
        (newOrder[orderOut: Order, DistInfo: Info]() -- mode passation de commandes
        (TotalCtuQty < (60/100)*CTUMax;
         curOrder : Order; curOrder -> type := ctu; curOrder -> date := today;
                curOrder -> dueDate := today + 15 -- 15 jours pour la livraison
                curOrder -> status := encours; curOrder -> qty := 20; -- par lot de 20
        orderOut(?curOrder);
        DistInfo(?"Nouvelle commande d'UCT envoyée !");
        )
        |
        (TotalScreenQty < (75/100)*ScrMax;
        curOrder : Order; curOrder -> type := screen; curOrder -> date := today;
                curOrder -> dueDate := today + 20 -- 20 jours pour la livraison
                curOrder -> status := encours; curOrder -> qty := 10; -- par lot de 10
        orderOut(?curOrder);
        DistInfo(?"Nouvelle commande d'imprimante envoyée !");
        )
        |
        (TotalPrinterQty < (80/100)*PrtMax;
        curOrder : Order; curOrder -> type := printer; curOrder -> date := today;
```

```
            curOrder -> dueDate := today + 20 -- 20 jours pour la livraison
            curOrder -> status := encours; curOrder -> qty := 5; -- par lot de 5
        orderOut(?curOrder);
        DistInfo(?"Nouvelle commande d'imprimantes envoyé !");
        )
        |
        (TotalAccQty < (80/100)*AccMax;
        curOrder : Order; curOrder -> type := Accessory; curOrder -> date := today;
            curOrder -> dueDate := today + 15 -- 15 jours pour la livraison
            curOrder -> status := encours; curOrder -> qty := 15; -- par lot de 15
        orderOut(?curOrder);
        DistInfo(?"Nouvelle commande d'accessoires envoyée !");
        ))
        []
        (stock() -- mode gestion des stocks existants
        )
endWhile
endvar
endproc
```

- **Supplier Process**

Le système fournisseur reçoit des ordres de commandes et des notifications venant soit des distributeurs, des entreprises d'assemblage, ou bien directement des clients finaux. Les ordres sont routés par le système et envoyés pour traitement dans les unités concernées, ça peut être des écrans, des unités centrales (UCT), des accessoires, ou bien des imprimantes (Printers). Chaque sous-système gère une liste qui lui permet de stocker temporairement les commandes reçues. Les commandes sont traitées par ordre d'arrivée, en fonction de la disponibilité des articles commandés. Le système fournisseur gère l'ensemble des synchronisations avec les autres services.

```
Process supplier [ OrderIn: Order, ItemOut:Item, cltInfo: Info] is
        Var curOrder,: Order, curItem:Item in
While (? orderIn)
        ((if (OrderIn ->type = screen) then
        OrderIn(?curOrder) ; cltInfo("commande d'ecrans reçue par le fournisseur !");
        screenSupply [scr, ScrInfo]();
        cltInfo(!" Ecran en cours de traitement ..."));
        []
        ((if (OrderIn ->type = ctu) then
        OrderIn(?curOrder) ;
        cltInfo("commande d'uct reçu par le fournisseur !");
        uctSupply [uct, uctInfo]();
        cltInfo(!" Unité centrale en cours de préparation ..."));
```

**[]**

*((**if** (OrderIn ->type = printer) **then***
*OrderIn(?curOrder) ;*
*cltInfo("commande d'imprimantes reçu par le fournisseur !");*
*printerSupply [prt, prtInfo]();*
*cltInfo(!" Imprimantes en cours de préparation ..."));*

**[]**

*((**if** (OrderIn ->type = accesory) **then***
*OrderIn(?curOrder) ;*
*cltInfo("commande d'accessoires reçu par le fournisseur !");*
*accesorySupply [acc, accInfo]();*
*cltInfo(!" Accessoires en cours de préparation ..."));*

***endWhile***
***Endvar***
***Endproc***

- **Shipping company**

La compagnie de transport s'occupe du chargement des marchandises dans les containers. Nous considérons que les containers ont un volume max à ne pas dépasser (volMax). Lorsqu'un article est chargé dans le container, le volume du container est incrémenté du volume de l'article. Si le volume max est atteint, le container est déclaré prêt à être embarqué pour livraison. Ainsi, les clients dont les marchandises ont été chargées seront notifiés automatiquement. La compagnie informe le port d'origine la disponibilité d'un container prêt pour embarquement. De même, le port de destination de la marchandise est informé automatiquement pour se préparer à la réception du container à la date d'arrivée estimée.

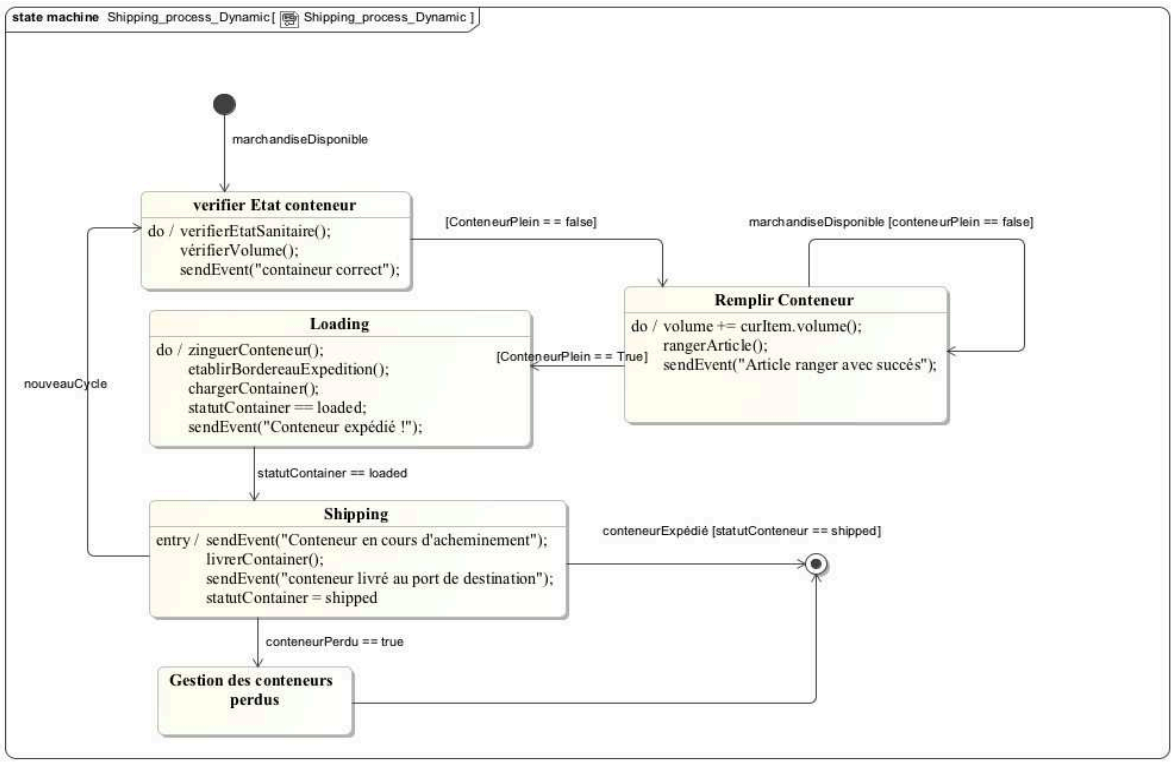


Figure 42. Diagramme d'états transition du processus de shipping

***Process*** *shipping [ ItemIn: Item, ContainerOut: Container, cltInfo:Info]* ***is***
***Var*** *curItem: Item, vol:float, TotalVol: float; MaxVol: float ,* ***List*** *ItemsList: Item,*
*curCont: Container* ***in***
*While (? ItemIn)*
*ItemIn(!curItem);vol:=curItem ->volume;*
*TotalVolume := TotalVolume + vol;*
***if*** *(TotalVolume > MaxVol)* ***then***
*cltInfo("Volume max atteint !"); TotalVolume := TotalVolume - vol;*
*screenSupply [scr, ScrInfo]();*
*cltInfo(!" Ecran en cours de traitement ...");*
***elseIf*** *(*
*ItemList->inssert(curItem); --Insertion dans la liste des Items*
***If*** *(TotalVolume > (75/100)*MaxVol)* ***then***
*curCont -> ItemsList := ItemsList;*
*cltInfo("Container pret pour expedition");*
*ContainerOut(?curCont)*
***Else*** *(* ***shipping*** *[ ItemIn, ContainerOut, cltInfo] ) -- continuer la réception d'items);*
***shipping*** *[ItemIn, ContainerOut, cltInfo]*
***endWhile***
***Endvar***
***Endproc***

- **Computer Assembly**

L’assemblage des ordinateurs est un processus dont la commande est périodique, toutes les semaines. L'assemblage proprement dit consiste à assembler toutes les parties pour former une seule machine à partir d’une unité centrale, un clavier, une souris, un écran. Les autres accessoires sont optionnels et ne seront pas pris en compte pour des besoins de simplicité.

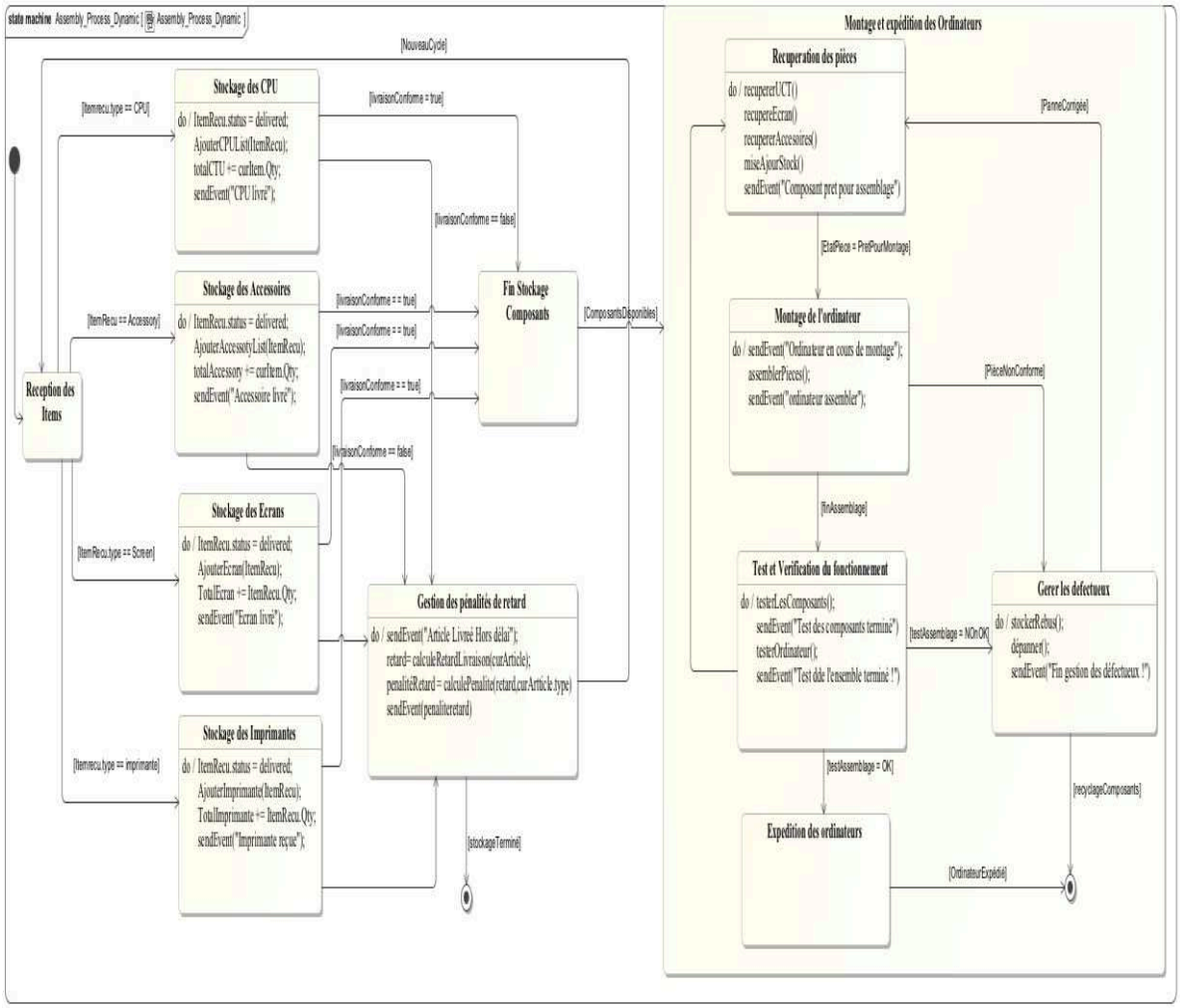


Figure 43. Diagramme d’états transition du système d’assemblage

```
Process Assembling [ componentIn: Item, ComputerOut: Computer, AssInfo:Info] is
        Var curComponent: Item,  List ComponentList: Item, List ComputerList: Computer
                currComputer: Computer in
While (true)
        ((?componentIn)
        componentIn(!curComponent);
        ComponentList ->hed(curComponent);
        TotalComponent := TotalComponent + curComponent ->qty;
        AssInfo(!" Composant reçu , assemblage en cours ...");
        )
        |||
        (curComputer : Computer;
        curComputer -> id := randInt();
        curComputer ->ctu : ComponentList - >ctus ->head();
        curComputer ->screen : ComponentList - >screens ->head();
        curComputer ->keyBoard : ComponentList - >keyBoards ->head();
        curComputer ->mouse : ComponentList - >mouses ->head();
        computerList->add(curComputer);
        ComputerOut(?curComputer) -- mise à disposition sur le marché
        AssInfo(!" Assemblage d'un ordinateur terminé ...");
        )
endWhile
Endvar
Endproc
```

**Définition des structures de données du système avec la spécification E-LOTOS**

Dans le listing ci-dessous, nous donnons la spécification E-LOTOS de quatre structures de données pour géréer respectivement les ordres de commande, les clients, les messages de confirmation et les ports des cannaux de communication.

```
type  Order is
        (cname => string,
        Qty => nat,
        orderDate => string,
        dueDate => string),
type Customer is
        (cname => string,
        portId => nat),
type orderConfirm is
        Send (OK) | ack(200)
Endtype
Type PortCanal is
        (canalId => string,
        canalName => string),
```

### 3.4.2. Introduction à JCSP

JCSP ou Communicating Sequential Processes for Java est une librairie java créée pour le calcul de processus. JCSP est un projet de l'université de Kent (Royaume-Uni), portés par Peter Welch, Neil Brown, James Moores, Kevein Chalmers et Bernard Sputh [110]. Outre l'intégration et l'extension de JCSP, ce cercle de réflexion traite divers aspects du calcul parallèle, de la synchronisation des processus et leur implémentation en Java. Parmi ces réflexions, nous citons "*CSP Networking for java (JCSP.net)"*, "*Communication Processes, components and Scaleable Systems"*, "*A CSP Model for Java Threads*", "*Concurrent Programming in Java*" et "*Communicating Threads for Java (CTJ)*". L'avantage de JCSP est d'être basé sur le langage orienté objet Java, et d'intégrer les concepts fondamentaux du calcul de processus communiquants. Il s'agit de la notion de canal de communication, de processus (Thread), de Timer, de noeud de dérivation (Fork) et de noeud de composition (Join) de processus. La dernière version disponible en date d'écriture de ce rapport est la version 2.1 avec la licence LGPL 2.1 Lesser GNU Public Licence. Cette version est téléchargeable sous forme de fichier .jar et intégrable directement dans un projet sous forme de librairie java.

### 3.4.3. Implémentation du cas d'utilisation avec JCSP

Reprenons l'exemple simple de la figure 33, où un client et son fournisseur communiquent via un bus collaboratif. Le client envoie des ordres de commande au fournisseur via le bus, qui sert de routeur de messages. Lorsque le fournisseur reçoit l'ordre de commande, il prépare la commande et la livre chez le client, en envoyant un message de livraison. Lorsque le client reçoit la marchandise livrée, il confirme la réception et ainsi le processus s'achève. La synchronisation des échnages fournisseur-client via le bus collaboratif est schématisée par le code de la figure 44.

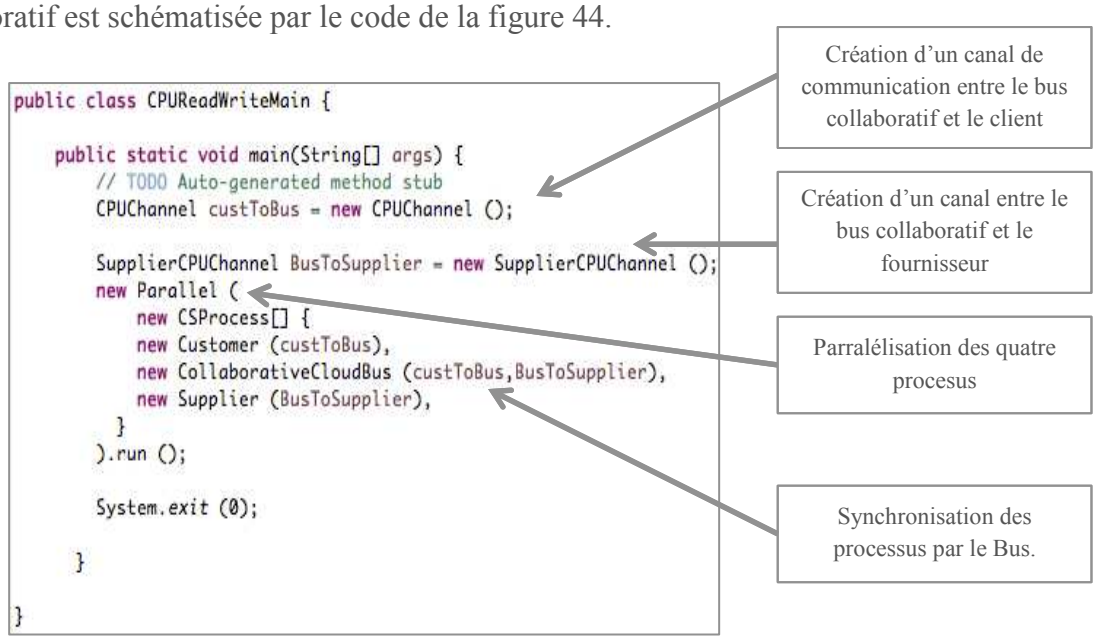


Figure 44. Synchronisation des processus client fournisseur via le bus

Dans la figure 44, le client utilise un canal de communication pour passer les ordres de commandes (*custToBus*) via le bus et de même le fournisseur utilise un canal pour déposer les marchandises à livrer (*BusToSupplier*). Le bus synchronise les échanges pour assurer la cohérence d'ensemble en utilisant les deux canaux (*custToBus, BusToSupplier* ).

Le processus client se contente d'envoyer des ordres de commande et d'attendre la livraison du fournisseur. Une fois les Items livrés, il en confirme la bonne réception. Le processus fournisseur lit les ordres de commandes passées par le client via le bus et donne une réponse. Le fournisseur est composé de deux sous processus : le processus de préparation de commandes et le processus de livraison.

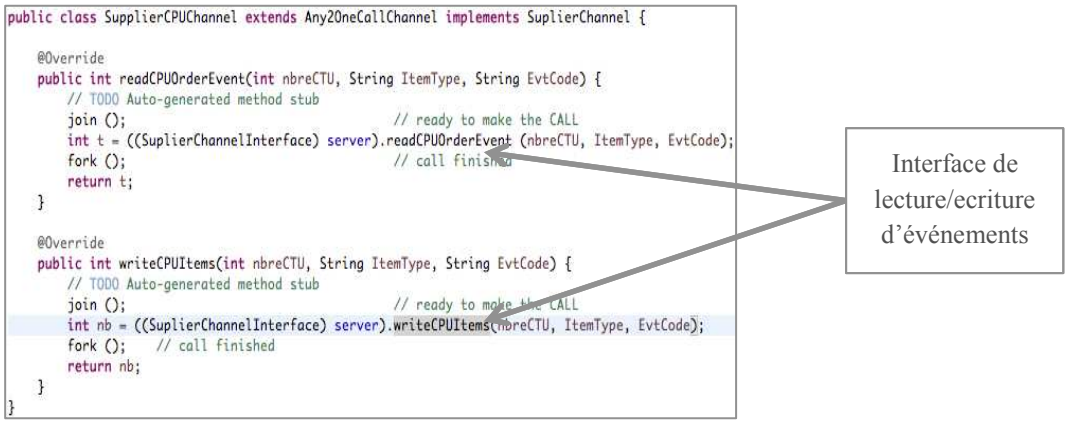


Figure 45. Interface de communication Bus-Fournisseur

Les interfaces de communications permettent de synchroniser les opérations de lecture et d'écritures sur les canaux de communication du bus collaboratif.

```
Time: 8:26:25:1474352786
Customer: ==> request for CPUs ...
Time: 8:26:25:1474352786
Time: 8:26:25:1474352786
Time: 8:26:26:1474352786
Cloud BUS: Receiving customer Item request, Number of CPU
Customer: ==> Number of CPU requested ==>  CPUs:210
Customer: ==> Receiving CPU deliver by Supplier ...
Time: 8:26:26:1474352788
Cloud BUS:  delivering CPU to customer ...
Cloud BUS:  Number CPU: ==> 200; Event code: ==> CPU-200
Time: 8:26:26:1474352788
Supplier:  delivering customer Items to Cloud Bus ...
Time: 1474352788
Supplier:  Number CPU: ==> 200; Event code: ==> CPU-200
Time: 8:26:25:1474352788
Customer: ==> Number of CPU receive ==>  CPUs:200
```

Figure 46. Trace d'exécution

## 3.5. Conclusion

Le but de ce chapitre est d'étudier la dynamique du flux logistique en utilisant la spécification E-LOTOS. Il ressort que les diagrammes d'états transitions d'UML permettent de donner une représentation graphique du comportement dynamique d'un workflow, mais que la spécification E-LOTOS est plus riche et plus adaptée. La puissance de ce formalisme réside dans sa syntaxe étendue et sa grammaire orientée vers les aspects de communication, de synchronisation et de simulation du processus tout en intégrant le temps comme élément principal.

Pour ce même objectif, nous avons proposé un modèle générique de workflow qui est basé sur le standard GSM [124], enrichi par les concepts de E-LOTOS notamment les opérateurs de synchronisation, les canaux de communication et les entrées sorties du Workflow. Enfin les exemples proposés permettent de mieux appréhender l'application de ce formalisme à l'analyse de la dynamique du workflow, afin de deceler les deadlocks, les lifelocks.

# Chapitre 4: Extraction des données et modélisation des connaissances dans un réseau de flux logistique

## 4.1. Introduction

Avec l'avènement de l'Internet des objets et du Cloud computing, le flux logistique et les outils de workflow management sont de plus en plus interconnectés. De la simple palette jusqu'aux acteurs de la chaîne d'approvisionnement, ils aident à mieux gérer le flux d'information généré par le réseau et les entités logistiques interconnectées. Les outils du web sémantique y occupent également une place prépondérante. En particulier, on y trouve les ontologies et les langages d'extraction de connaissances hétérogènes ou peu structurées, dans les environnements distribués à l'instar de SPARQL.

Dans ce chapitre, nous proposons une ontologie web pour la représentation et la classification des concepts dans le domaine du flux logistique. Cette ontologie permet aussi l'extraction de connaissances dans le réseau de flux avec des requêtes pour la gestion et la coordination des entités logistiques. Par ailleurs, cette ontologie peut être utilisée comme format d'échange et de partage d'informations entre les différents acteurs logistiques. Ceci facilite l'interopérabilité des systèmes d'information et la gestion du flux de marchandises.

Ce travail est basé sur trois standards: et la gestion du flux de marchandises.
Ce travail est basé sur trois standards : le langage de définition de processus XPDL (*XML Process Definition Language*) et le BPML *(Business Process Modeling Language)*. Nous utilisons aussi OWL (*Web Ontology Language*), un des standards pour la spécification des ontologies web.

Cette proposition se différencie des travaux précédents en trois points essentiels. Le premier point consiste à considérer le flux logistique comme un réseau d'entités connectées entre elles, qui évoluent dans le temps en fonction des événements *discrets* survenus. Le deuxième point est l'introduction de la notion de contexte dans la gestion du flux, pour mieux appréhender l'ubiquité des objets logistiques et leur changement de statut. Enfin, nous prenons en compte la notion d'interface de communication (*gateway*) entre le flux et son environnement en utilisant le standard *NGSI* (*Next Generation Service Interface*). De plus, nous appliquons le design-pattern Publish/Subscribe pour la souscription au contexte du flux, la publication et le partage d'événements entre les entités et les différents acteurs.

La première partie du chapitre dresse un état de l'art sur les différentes ontologies web en lien avec la modélisation du flux logistique et leurs différents usages. La deuxième présente notre ontologie, en insistant sur des aspects tels que la représentation de la connaissance dans le réseau de flux : nous y abordons le partage de données, l'interopérabilité et l'extraction des connaissances. Par la suite, nous présentons quelques cas d'utilisation de cette ontologie dans la gestion d'un flux de marchandises

conteneurisées. Il est aussi question de l'extraction d'informations pour le suivi et la traçabilité des objets logistiques. Le dernier paragraphe fait une synthèse sur cette approche.

## 4.2. Architecture générale d'intégration du flux logistique dans le Cloud

### 4.2.1. Pourquoi une intégration du flux logistique dans le Cloud

Le problème à résoudre ou l'objectif à atteindre à travers cette intégration du flux logistique dans le Cloud est celui de la collecte et de la gestion des données dans le temps et dans l'espace. Nous soulignons quelques points importants liés à cette problématique comme suit:

- **Collecte et disponibilité des données issues du flux logistique**. A ce niveau, il s'agit de remonter l'information sur le flux en temps réel vers le/les utilisateurs concernés, c'est-à-dire ceux qui gèrent le flux logistique. Le gestionnaire du flux d'approvisionnement a besoin d'avoir des informations sur la position des containers avec leurs contenus, au fur et à mesure que ces entités logistiques changent de position géographique dans le temps.

- **Tracking des données en temps réel**. La question qui se pose ici est celle de savoir ce qui s'est déroulé sur le flux dans le passé. Concrètement, il s'agit de déterminer quelles sont les différentes positions successives occupées par une entité logistique (Where), quelles sont les transformations ou les opérations logistiques subies par cette entité (What), à quel moment ces opérations ont-elles été réalisées sur l'entité logistique (When), qui sont les acteurs logistiques qui ont agi sur cette entité logistique (Whom/Owner).

- **Prévision du flux logistique.** Les questions qui se sont posées sur le flux et l'entité logistiques dans le passé se posent pour le futur proche ou lointain. Ceci pour des besoins de planification à court ou à long terme. Il s'agit de déterminer :
  - ➔ Quelles sont les positions futures ou par quels lieux l'entité logistique passera dans le futur (Where).
  - ➔ Quelles sont les transformations qui vont s'opérer à ces positions ou dans ces lieux (What).
  - ➔ À quel moment l'entité est-elle censé passer par ces lieux (When)
  - ➔ Par quels acteurs logistiques (Whom/Owner).

- **La gestion des droits d'accès aux données collectées**. Il s'agit donc de trouver une politique de gestion des droits d'accès aux données collectées, de façon dynamique, en fonction de l'évolution du flux logistique. La question principale est la suivante: "Qui a accès à telles informations, quels sont ses droits (lecture, ecriture, mise à jour, etc.)". Il s'agit aussi de définir des niveaux hiérarchiques d'accessibilité à l'information en fonction du rôle que l'acteur joue dans le flux logistique (*tracking*

*vertical*). Précisément, le superviseur du flux doit avoir accès à toutes les informations et les vues sur l'ensemble du flux qu'il gère. A l'opposé, un opérateur (cariste) n'aura accès qu'aux informations le concernant telles que le nombre et l'emplacement des palettes ou batchs qu'il remplit dans le container.

- **Extraction et mise à jour des données.** Vu le nombre d'acteurs et la quantité de marchandises impliqués dans le flux de bout en bout, la quantité et le volume d'informations récoltées et partagées entre les acteurs au sein d'un flux peut poser problème. Il s'agit donc d'améliorer l'architecture de stockage pour faciliter l'interrogation des données (query). Tout en gérant les mises à jour, avec les problèmes connexes tels que la cohérence, l'intégrité et la fiabilité des informations dans la plateforme. Il est aussi question de faire des mises à jour sur les données d'un acteur logistique ou d'une organisation sans affecter les données relatives aux autres acteurs ou aux autres flux. Enfin, déterminer la meilleure politique de mise à jour des données : par batchs, à la volée, dès lors qu'une information arrive, ou par flux, utilisation ou non de la technique de catching pour une mise à jour future en batch ; clusterisation ou non des données en fonction de la géographie du flux ; mise à jour par cluster géographique, ou mise à jour par fenêtre temporelle... Autant de questions auxquelles nous tentons de répondre.

### 4.2.2. Architecture générale

Pour résoudre les différents problèmes que nous venons d'énoncer, nous proposons une démarche architecturale et la méthode de résolution basée sur la programmation multi-agents. La première analyse consiste à décrire la structure de données que nous utilisons et les raisons de ce choix, ses avantages et ses inconvénients comparés à d'autres structures de stockage existantes. Ensuite, nous décrivons l'architecture qui nous permet d'intégrer les données du flux logistique dans la plateforme Cloud. La dernière analyse explique comment nous utilisons la programmation multi-agents pour résoudre ces.

L'architecture d'intégration du flux que nous proposons est subdivisée en quatre niveaux ou couches, en plus de la couche multi-agents (figure 47). La ***couche physique*** représente le flux réel de marchandises. À ce niveau nous retrouvons les conteneurs, les palettes, les camions de transport et les acteurs logistiques impliqués dans le flux. La ***couche Capteurs*** (Sensors Network) est constituée du réseau de capteurs qui relèvent les informations sur les entités physiques du flux. Ces informations sont la position de chaque entité logique, la température, les événements qui surviennent sur ces entités. La ***couche service Cloud*** est constituée de la plateforme de stockage et de traitement des informations issues du réseau de capteurs. Cette couche offre des solutions logicielles consommables à la demande en tant que service (SaaS), basées sur le modèle SOA (*Service Oriented Architecture*). Parmi ces services nous pouvons citer le stockage, la mise à jour, la gestion des droits d'accès et la gestion des événements complexes (*Complex Event Processing*). La ***couche End-User Device*** fournit une interface à l'utilisateur final. Nous reviendrons sur chacune des couches pour une spécification détaillée. Dans ce chapitre, nous décrivons la

***couche Multi-agents***, l'architecture du réseau multi-agents constitué et spécifions la manière dont les agents communiquent entre eux pour faciliter la coordination du flux et le partage d'informations.

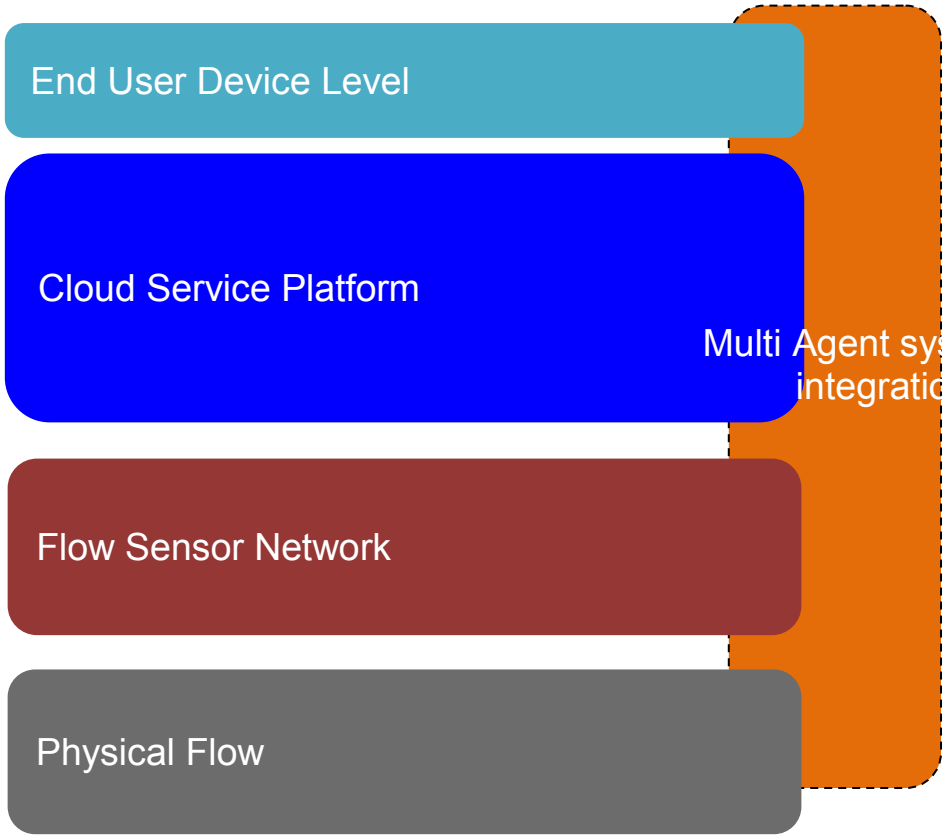


Figure 47. Architecture d'intégration du flux logistique dans le Cloud

Le paragraphe suivant explique de manière détaillée la structure et le fonctionnement des deux premières couches de cette architecture (niveau physique et réseau de capteurs).

### 4.2.3. Niveau physique du flux logistique et réseaux de capteurs

- Couche physique

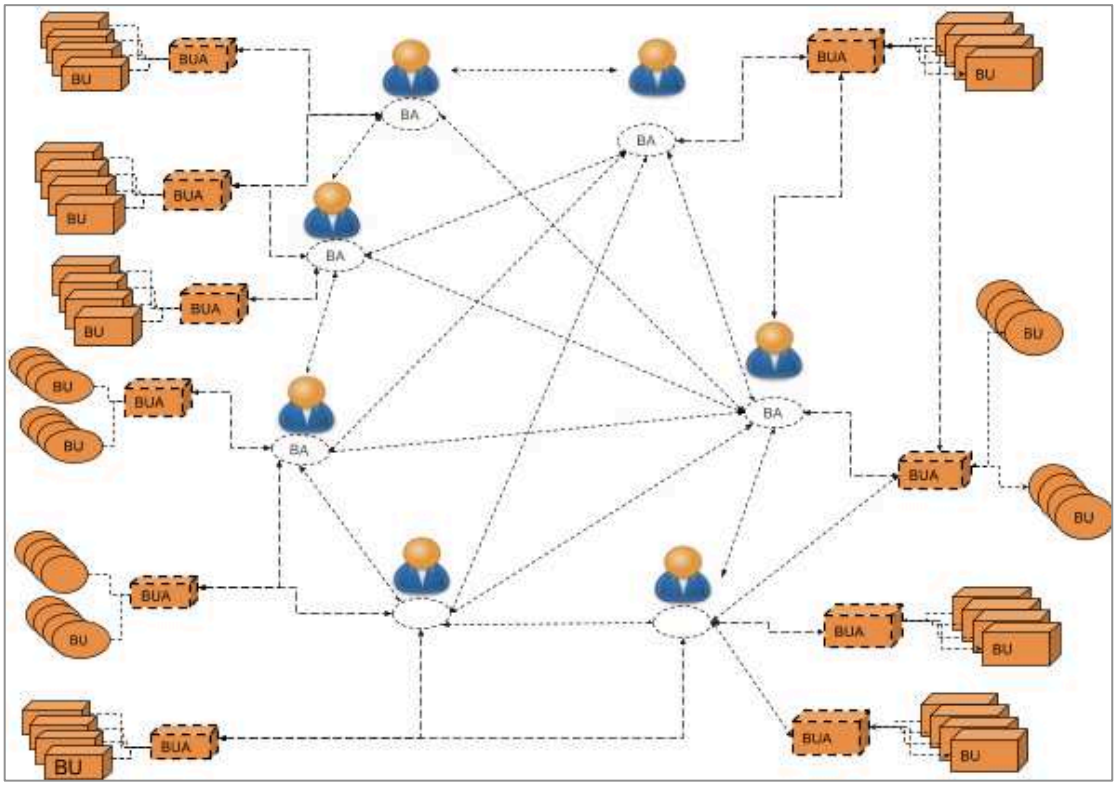


Figure 48. Réseau de flux physique

La couche physique du flux est constituée des entités logistiques (***BU-Business Unit***), des moyens de transport, des infrastructures de transit du flux de marchandises. Par entités logistiques nous entendons les marchandises conteneurisées ou non, mobiles ou fixes (robot de la chaîne de manufacture). Les entités logistiques peuvent être regroupées pour former un seul bloc, c'est l'exemple des conteneurs qui sont des regroupements de palettes ou de produits. Ces regroupements constituent un agregat de marchandises (***BUA-Business Unit Agregator***). Les relations entre un Business Unit et un Business Unit Agregator peuvent prendre plusieurs formes aux sémantiques différentes, comme discuté au paragraphe dédié à l'ontologie. Les agregats d'unités logistiques ont administrés par des acteurs logistiques ***Business Actor*** (***BA)***. Il s'agit à ce niveau des opérateurs qui réalisent des activités sur le flux. Il s'agit aussi de services logistiques qui génèrent des événements sur l'unité logistique ("conteneur arrivé à destination", "palette livrée", ...). Les agents communiquent entre eux en partageant les informations sur le flux via la plateforme. Ce réseau d'acteurs est clusterisé comme nous le décrivons par la suite. Le diagramme UML suivant représente le niveau physique du flux de marchandises.

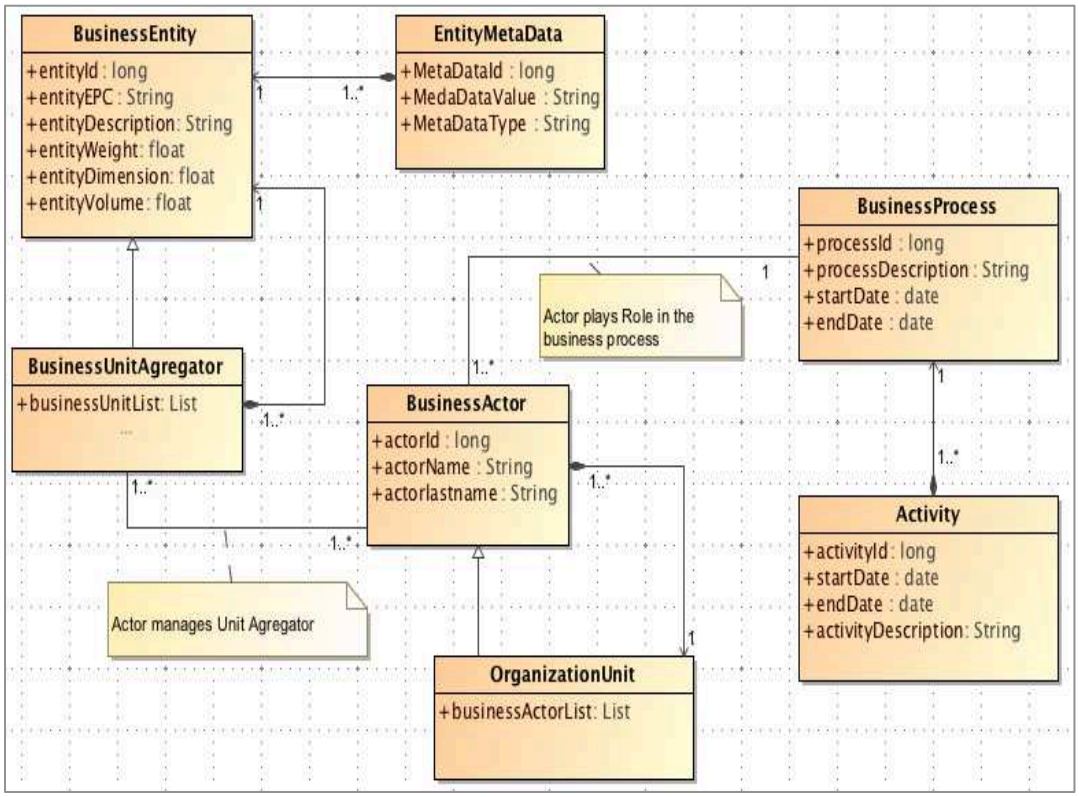


Figure 49. Diagramme UML du niveau physique du flux de marchandises

- **Réseau de capteurs**

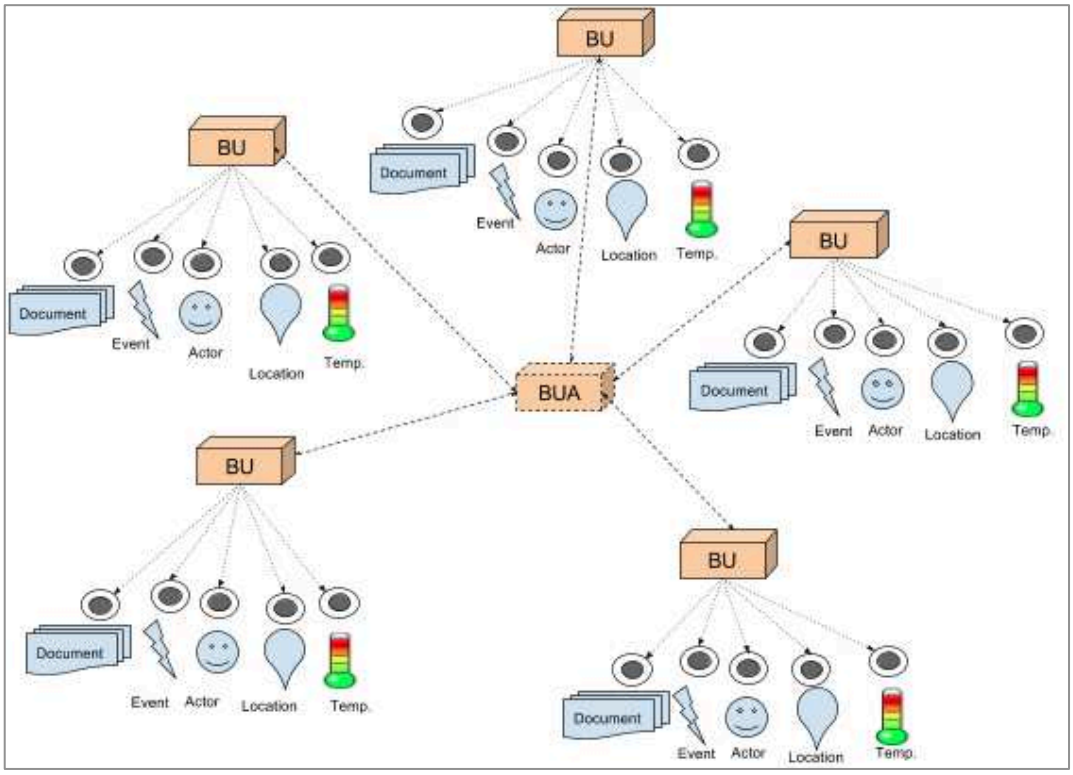


Figure 50. Réseau des entités logistiques avec le réseau de capteurs associé

Nous considérons l'unité logistique comme une entité communicante, équipée de code à barres, de capteurs et/ou d'actuateurs, éventuellement muni d'une capacité de traitement de l'information. Le Business Unit est donc équipé de plusieurs capteurs en fonction des usages (marchandises conteneurisées, cartons, produits en vrac, ...), ou des circonstances (flux de fret, flux de marchandises liquides, ...). Ces capteurs forment un ensemble de noeuds dans un réseau de flux connecté. Un ***noeud*** (***SensorNode***) étant l'unité fondamentale. Un ensemble de noeuds peuvent être agrégés pour former un réseau de noeuds coordonnés par l'agrégateur ***BUA***. Les agrégats de noeuds forment à leur tour un réseau plus large, géré par les coordinateurs du réseau (*Coordinator*).

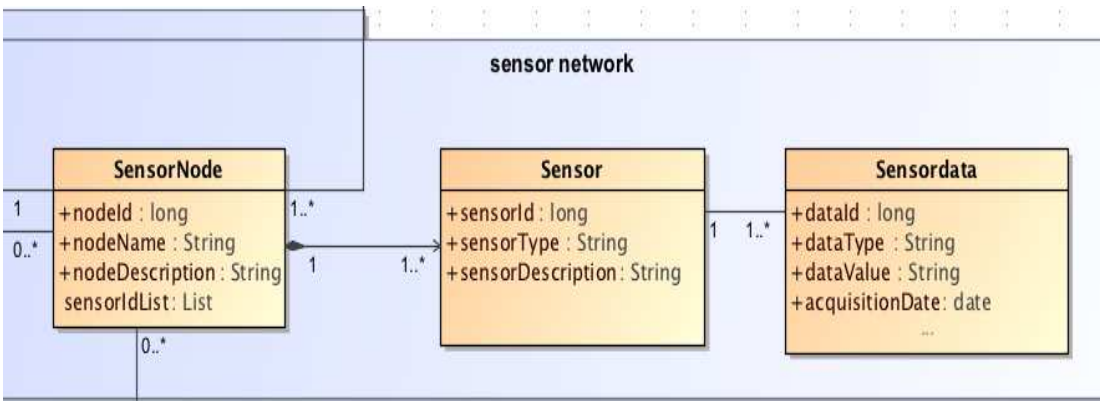


Figure 51. Schéma UML du réseau de capteurs

### 4.2.4. Le système d'intégration multi-agents

- **Architecture du système multi-agents**

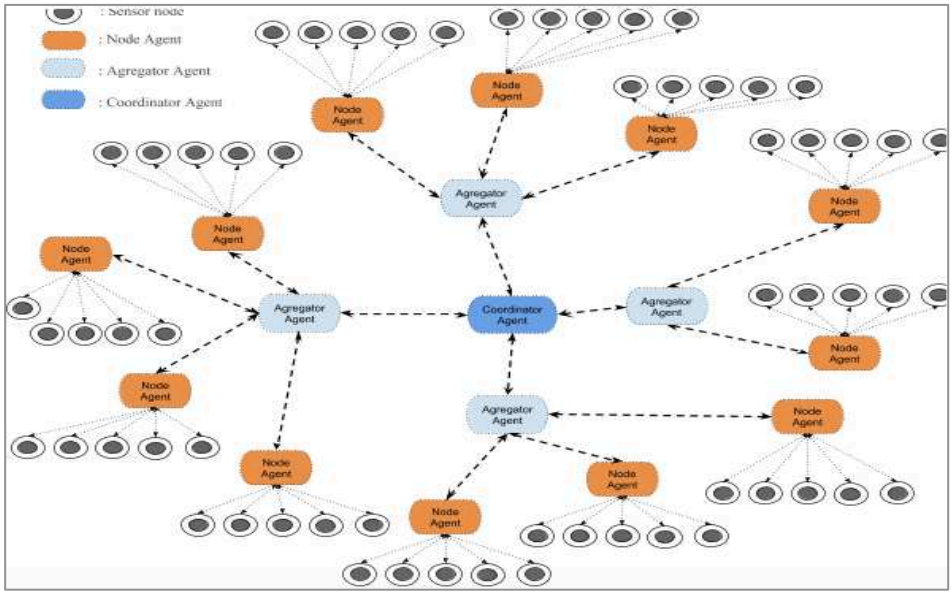


Figure 52. Architecture du système multi-agents d'intégration dans le réseau du flux physique et du réseau de capteurs

A chaque BU, nous associons un agent logiciel qui se charge de l'intégration avec le Cloud et le réseau de capteurs. L'agent logiciel gère ainsi l'ensemble des capteurs qui équipent l'unité logistique. Nous l'appellerons *Node Agent* ou agent dédié à un noeud. Les noeuds peuvent être regroupés en une structure de petits réseaux associés à un agrégat d'unités logistiques (Un camion transportant des conteneurs par exemple). Pour cette agrégation, nous utilisons un agent agrégateur (***Agregator Agent)***. Celui-ci a pour fonction de router l'information des différents noeuds vers la plateforme et de jouer le rôle d'interface. Il gère aussi les droits d'accès aux informations des différents noeuds. Les agrégateurs peuvent se réunir pour former un réseau plus large qui nécessite les services d'un coordinateur. Le coordinateur gère ainsi les agrégateurs de flux en étant lui-même un agrégateur de flux d'informations dans le réseau ainsi constitué. Nous pouvons voir le réseau d'un coordinateur comme étant une flotte ***(Coordinator Agent)*** de véhicules (***Agregator Agent***), transportant chacun un ensemble de conteneurs (***Node Agent***). Cette description nous conduit au schéma de la figure 53.

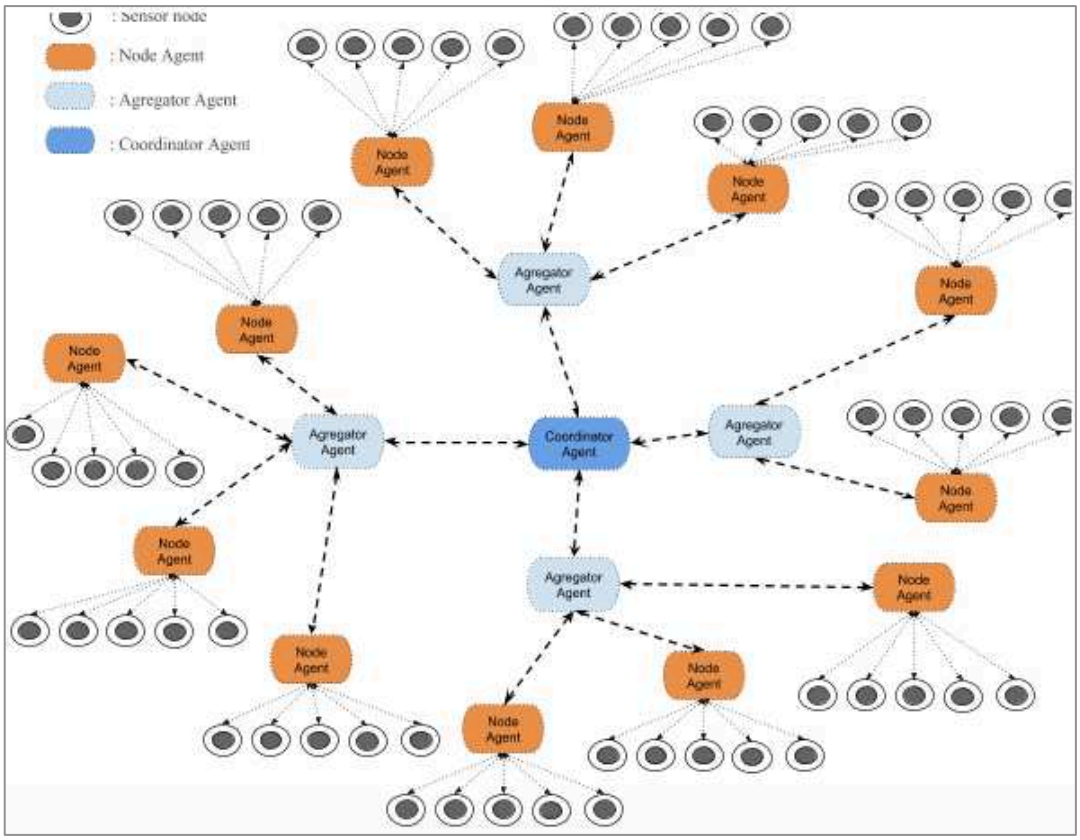


Figure 53. Réseau multi-agents

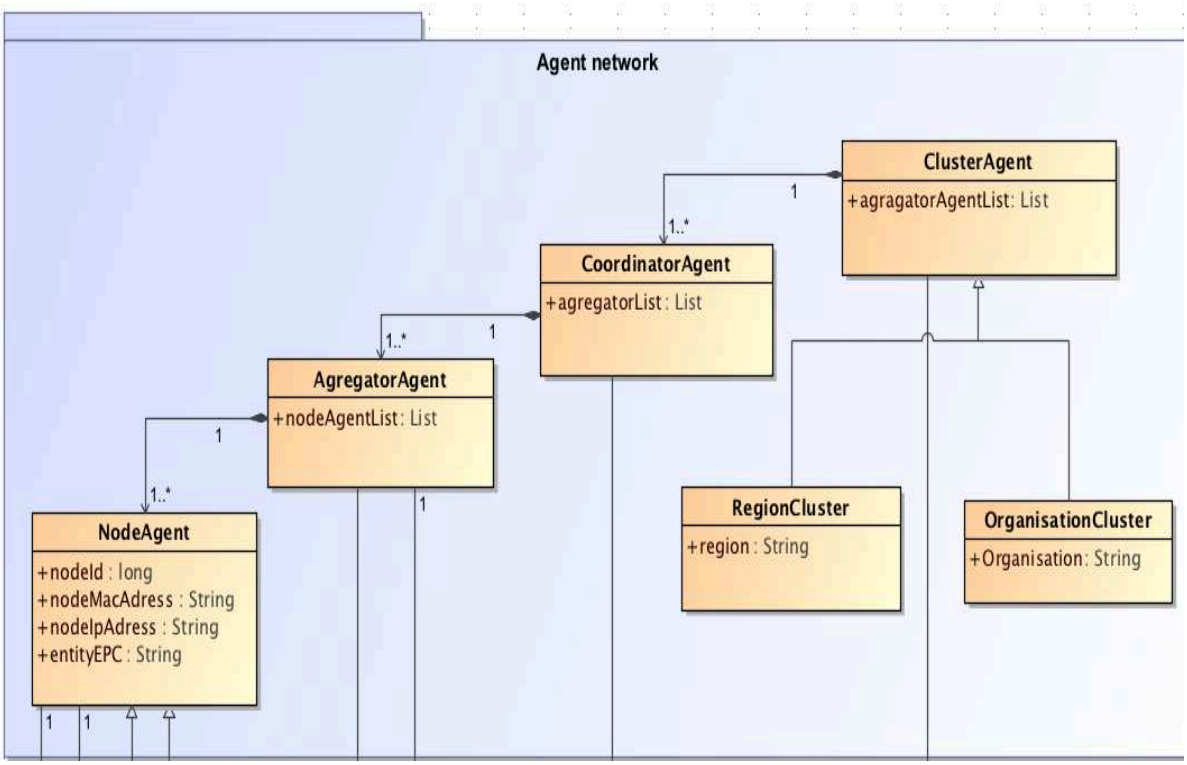


Figure 54. Diagramme UML du réseau d'agents

- **Cluster et régionalisation des agents**

Nous introduisons la notion de cluster d'agents (figure 55) pour répondre au besoin de régionalisation des entités logistiques. En effet, les entités physiques peuvent être regroupées selon leur zone géographique ou selon l'organisation propriétaire du flux. Nous définissons ainsi deux types de cluster, le cluster ***Region*** et le cluster ***Organization***. Le cluster ***Region*** est un ensemble de flux d'entités logistiques liées par leur appartenance à la même région géographique (une ville, une province, un pays, un continent, ...). Nous savons que le flux est contraint de changer de localisation dans la plupart des cas. Il est donc intéressant de clusteriser le flux dans la région vers laquelle il se déplace. Par exemple un flux de marchandises qui reste dans un même pays. La régionalisation du flux ou des entités joue deux rôles importants. En premier lieu, elle permet de faciliter le stockage des informations dans la plateforme et de faciliter les recherches et les mises à jour du contexte des entités logistiques. Le cluster ***Organization*** permet de regrouper les flux selon les organisations propriétaires : une ***organisation*** pourrait ainsi créer plusieurs clusters selon ses unités logistiques. Une ***firme*** constituée de plusieurs ***branches*** peut ainsi créer plusieurs clusters associés à chacune de ses branches et déléguer la gestion du flux régional à chacune des branches concernées. Cette façon de regrouper le flux logistique est davantage liée à l'organisation des ressources humaines et matérielles et aux services logistiques qui interagissent avec le flux d'entités.

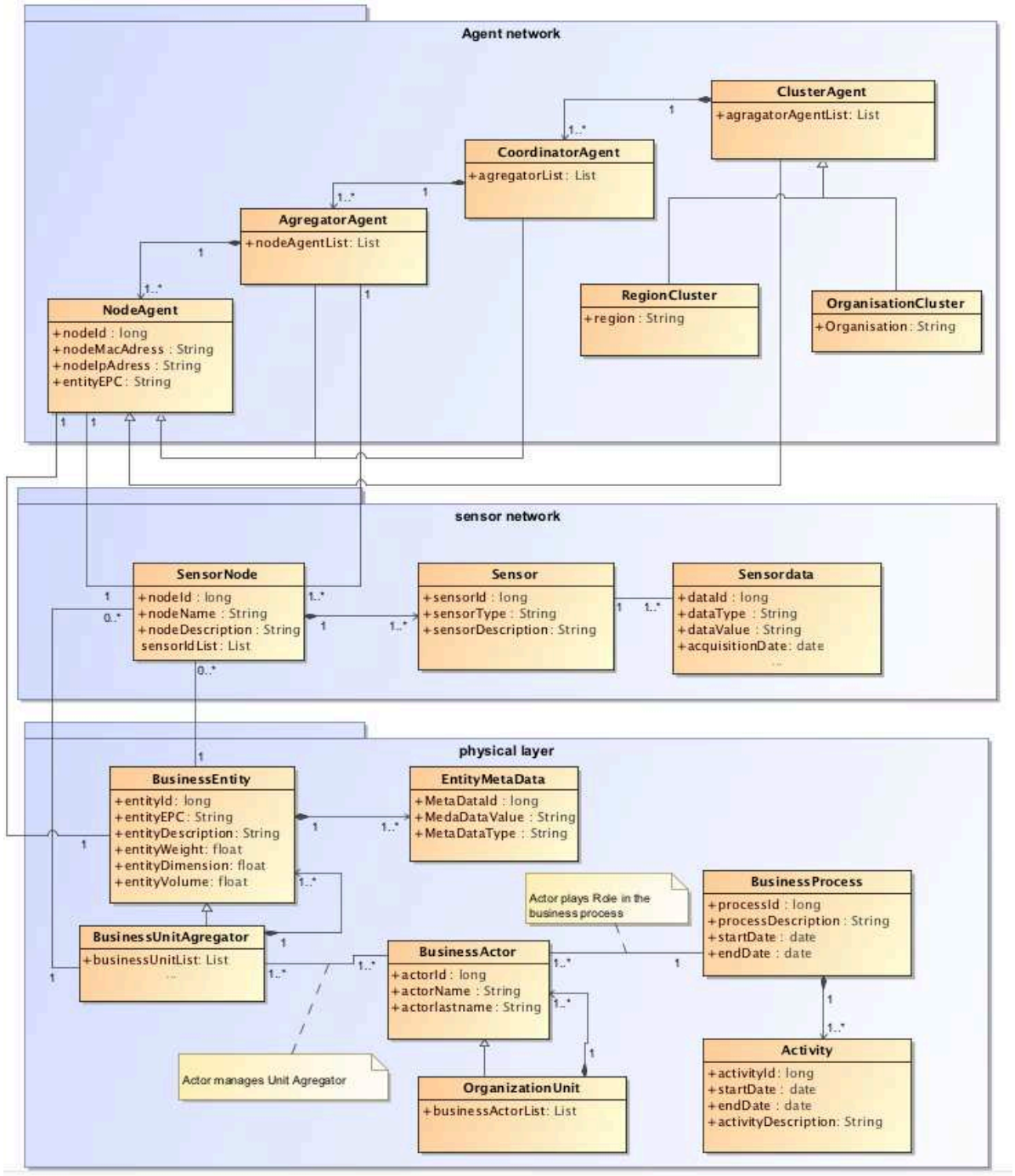


Figure 55. Schéma UML de l'architecture multi-agents

**Spécification des services et des interfaces de communication des agents logiciels**

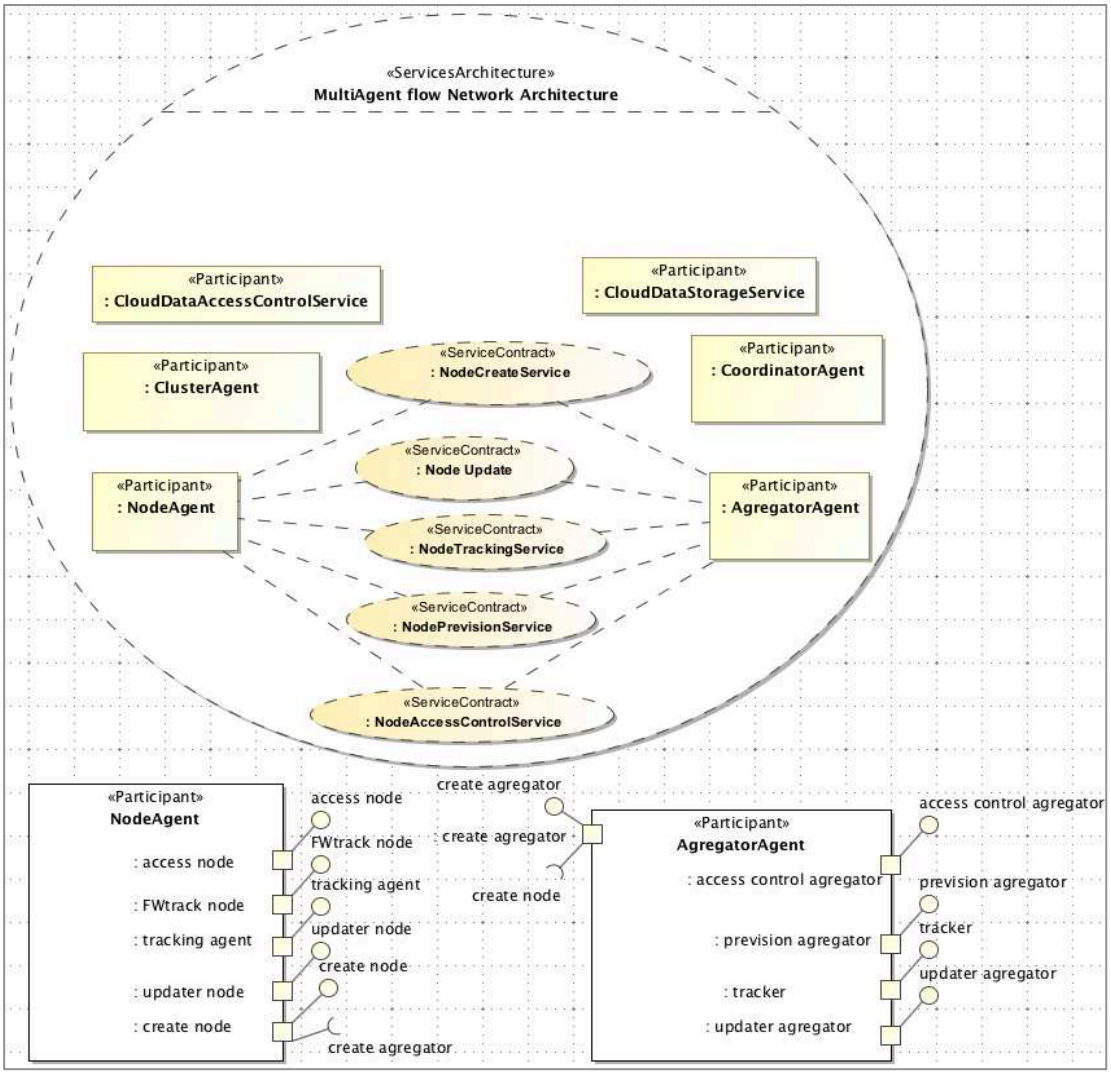


Figure 56. Architecture Orientée Service du système

L'agent associé à un noeud est représenté par le composant ***NodeAgent***. Ce composant communique avec l'agrégrégateur *via* cinq ports reliés à cinq services:

- Le service de création des noeuds (***NodeCreateService***)
- Le service de mise à jour des noeuds (***NodeUpdateService***). Ce serivce met à jour les informations relatives au noeud dans la base de données de la plateforme.
- Le service de tracking des noeuds (***NodeTrackingservice***). Ce serivce est solicité pour avoir la position et les dernières informations sur le nœud.
- Le service de prévision (***NodePrevisionService***) qui permet de prévoir les déplacements futurs du noeud, ainsi que les opérations logistiques qui vont être effectuées dessus dans un futur proche.
- Le service de gestion des droits d'accés aux noeuds (***NodeAccessControlService***).

Nous considérons dans la spécification que le noeud agrégateur sert d'intermédiaire et de routeur entre le noeud Agent et la plateforme. Ainsi, toutes les requêtes adressées à la plateforme par le noeud passent par le noeud agrégateur associé.

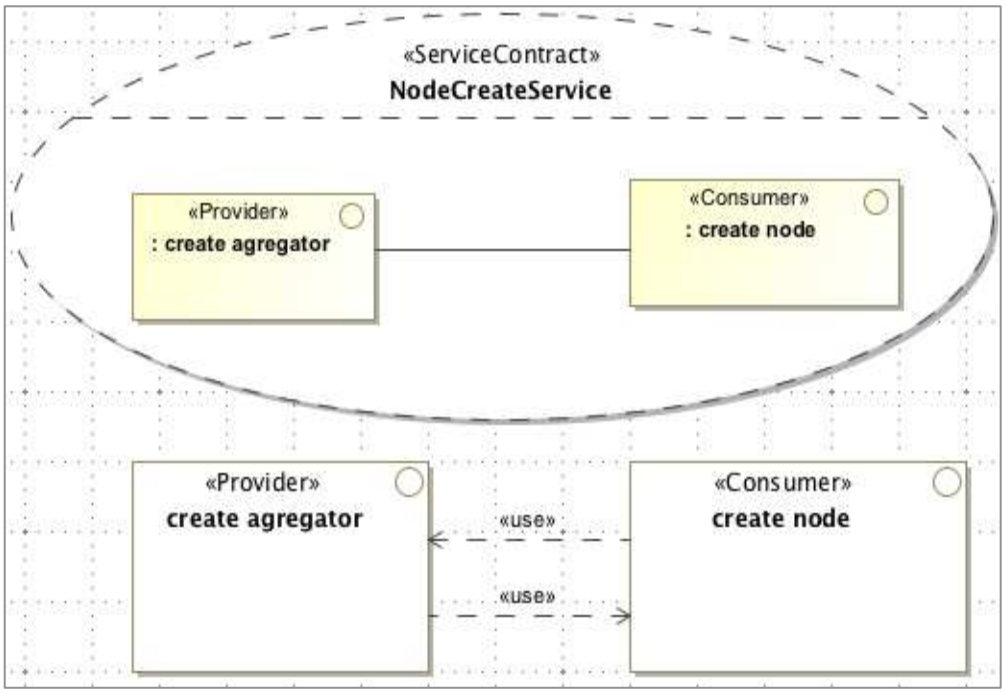


Figure 57. Service de création d'un NodeAgent

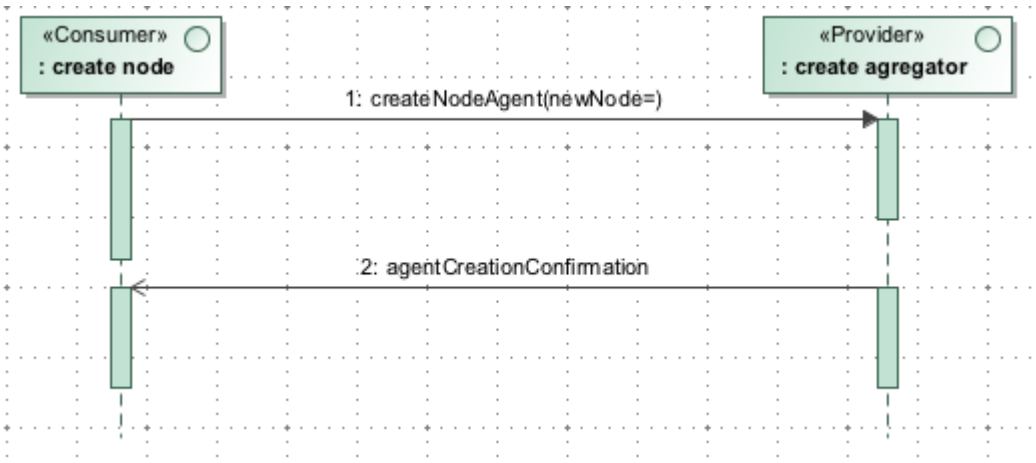


Figure 58. Diagramme d'interaction lors de la création d'un NodeAgent

## 4.3. Le web sémantique

### 4.3.1. Généralités et définitions

Malgré la mise en place du langage HTML pour standardiser le web, nous remarquons que les ressources demeurent hétérogènes de par leurs formats de représentation, de

stockage et de présentation. Le Web sémantique a donc donné naissance à plusieurs langages et outils de formalisation des connaissances du web. Ceci afin de faciliter la structuration des connaissances, l'inférence, l'automatisation et l'échange des savoirs dans le web. Parmi ces langages nous pouvons citer le Resource Description Language (*RDF*), le eXtensible Markup Language (*XML*), le Javascript Object Notation (*JSON*) et le Web Ontology Language (*OWL*). RDF se donne pour objectif de proposer un langage de structuration des données et des métadonnées sous forme de triplets (Sujet-Prédicat-Objet). Il permet de fournir des informations sur les ressources web, l'interopérabilité des automates qui utilisent ou se partagent ces ressources. Un manque du langage RDF est de ne pas pouvoir raisonner sur les données ainsi structurées, d'où l'apport de OWL qui permet de faire des raisonnements et d'automatiser les traitements sur les données structurées par RDF.

Nicolas Guarino [118] définit l'ontologie comme étant l'ensemble des primitives de connaissances, le sens des concepts du niveau épistémologique. Selon Guarino le niveau épistémologique est celui de la structuration des concepts, alors que le niveau ontologique est celui de la signification de ces concepts, ce qui conduit Guarino à situer le niveau ontologique entre le niveau épistémologique et le niveau logique. Au niveau ontologique, les primitives de connaissances satisfont les postulats de sens formel, ce qui permet de limiter l'interprétation d'une théorie logique sur la base de l'ontologie formelle. Quant à Gruber [87], il définit une ontologie comme étant une conceptualisation, c'est à dire un ensemble de concepts (entités, attributs, processus), leur définition et leur interrelation. Cet ensemble de concepts permet d'exprimer et d'extraire de la connaissance dans un système, ainsi qu'à plusieurs systèmes et de s'échanger des connaissances dans un format unique.

Les ontologies peuvent être utilisées pour plusieurs finalités. La représentation de la connaissance, l'extraction de la connaissance, le partage d'information et l'interopérabilité des systèmes communicants. De plus, les ontologies génériques permettent d'organiser et catégoriser les concepts ou la connaissance d'un domaine précis, à des fins de réutilisabilité. C'est le cas de SUA (*Standard Upper Ontology* [119]) qui a mis en place SUMO (*Suggested Upper Merged Ontology*) pour la généricité des grandes catégories d'objets et de pensées.

Dans la litératture récente, plusieurs travaux mentionnent l'usage des ontologies à diverses finalités. Ainsi, Stefan Poslad [120] proposent en 2015 une sémantique IoT pour un système d'alerte de désastre environnementaux et de gestion de crises naturelles. Cette sémantique permet l'acquisition et le partage des données des capteurs, facilite le plug-and play des sources de données, ainsi que l'analyse des méta-données pour l'interopérabilité et l'orchestration des services.

### 4.3.2. Rôle et importance des ontologies

De par leur diversité, il est difficile de faire une classification universelle des ontologies. Néanmoins, nous pouvons les catégoriser selon qu'elles sont génériques ou spécifiques à un domaine d'application (Ontologie du domaine). Nous pouvons aussi les classer selon qu'elles servent de représentation des concepts d'un domaine (ontologie de

représentation) ou qu'elles permettent d'efectuer des raisonnements (ontologie d'inférence). Les ontologies ont des usages multiples, parmi lesquelles nous pouvons citer :

- **Modèle de représentation et classification des concepts d'un domaine:** dans ce rôle, les ontologies sont utilisées pour identifier, représenter, définir et classifier les concepts relatifs à un domaine de connaissance. Aussi, elles permettent d'exprimer la manière de combiner les différents concepts pour produire de nouvelles connaissances à partir de connaissances existantes. C'est le cas de règles d'inférence d'un Système à Base de Connaissance (*SBC*).

- **Extraction de connaissances, raisonnement ou inférence:** Si nous reprenons la philosophie de Guarino, l'un des avantages d'une ontologie est de donner un sens à un ensemble de concepts, d'entités, de relations entre ces entités. L'ontologie permet ainsi de déduire d'autres connaissances à partir des connaissances de base, des déductions logiques dans un SBC.

- **Echange de données et méta-données hétérogènes de sources diverses dans le web sémantique:** l'un des avantages d'une ontologie est de jouer le rôle de médiateur au niveau du format d'échange de données, des ressources, des connaissances provenant de sources hétérogènes, formalisées à l'aide de Data-Type-Definition (*DTD*) par exemple. Dans ce rôle de schéma médiateur, l'ontologie permet à plusieurs systèmes de nature différente de partager une même sémantique des termes du domaine, indispensable à l'interopérabilité des systèmes.

### 4.3.3. Concepts fondamentaux et langages de mise en oeuvre

**Concepts fondamentaux d'une ontologie**

La mise en place d'une ontologie web nécessite la définition de plusieurs concepts, parmi lesquels:

- **La notion de classe**: les classes dans une ontologie représentent un ensemble d'entités ayant des mêmes propriétés, les mêmes caractéristiques. Nous pouvons dire par exemple que la classe *Container* désigne l'ensemble des conteneurs dans le flux logistique. De même, nous pouvons désigner l'ensemble des palettes par le concept de *Palette*, et les événements par la classe *Event*.
- **La notion d'individu**: les individus constituent les éléments du domaine à représenter. Un individu peut être considéré comme une instance d'une classe.
- **Le concept de propriété**: une propriété est un attribut qui permet de décrire les individus d'une classe. La propriété permet de rattacher à un individu une certaine caractéristique. Par exemple, un container est caractérisé par une dimension. La propriéte dimension du conteneur varie en fonction de son type. Ainsi, on distingue des conteneurs de 10 pieds, 20 pieds, et 30 pieds. De même une entité logistique a un identifiant, un nom et une localisation.

- **Le concept de relation**: les relations sont de plusieurs types dans une ontologie. Les relations d'héritage exprimées par "*is-a*", permettent de structurer et catégoriser les individus et les différents concepts de l'ontologie. Cette relation permet aussi à un individu enfant d'hériter des propriétés de l'individu parent, de manière transitive. Par exemple, un conteneur est une ressource. Toute les ressources ont un nom et un identifiant (attributs) donc le container a un nom et un identifiant.

**Langages de mise en oeuvre**

Il existe dans la litérature plusieurs langages de mise en oeuvre des ontologies, entre autres OWL, pour la réalisation des ontologies web ; DAML (*DARPA Agent Markup Language*) ; LOOM pour la construction d'application intelligentes, OKBC (*Open Knowledge Base Connectivity*) pour l'accès à des bases de connaissances ; KIF (*Knowledge Interchange Format*) pour l'échange de connaissances entre systèmes hétérogènes. Parmi ces langages, OWL a la particularité d'être un standard du W3C. OWL se base sur le langage RDF (*Resource Description Framework*) pour la représentation des ressources, tandis que RDF-S (*RDF-Schema*) est utilisé pour la structuration et le partage de ressources dans le web. Par ailleurs, OWL offre un module qui permet de vérifier si la hiérarchie des classes est correcte (le *reasoner*), ainsi que plusieurs opérateurs algébriques tels que l'intersection, l'union, la négation au niveau des classes, la reflexivité et la transitivité dans l'expression des relations.

## 4.4. Mise en place d'une ontologie web pour le workflow connecté

Dans cette section, nous proposons une ontologie web de modélisation du workflow connecté. Pour la représentation des concepts du flux logistique, nous utilisons les dictionnaires du *XML Process Definition Language* (*XPDL*) pour la terminologie liée au workflow, le *Guard Stage Milestone* pour la gestion des événements et des triggers, et d'autres termes empruntés du web sémantique et de l'Internet des Objets. Les objectifs que nous visons par cette ontologie sont les suivants:

- Proposer une sémantique pour la spécification, la classification des entités logistiques et des interrelations entre ces entités.
- Proposer une sémantique de représentation du workflow, des activités, des tâches et des relations de coordination inter-processus.
- Partager les informations entre workflows, pour assurer l'interopérabilité des acteurs logistiques et le workflow en utilisant les canaux de communication (gateways) et le pattern Publish/Subscribe.
- Faciliter la découverte des entités business et la mise à jour de leur contexte par des événements émis par les acteurs logistiques à l'aide des capteurs, des terminaux mobiles ou des systèmes de gestion de flux de marchandises.
- Servir de base pour l'extraction de connaissances dans un réseau de flux logistique, à des fins de traçabilité, de suivi ou de mise en place de tableaux de bords et KPI pour la prise de décision.

### 4.4.1. Représentation des concepts du workflow et leur relation

- **Entity**

Nous utilisons le terme *Entity* ici pour désigner ce qui peut être engagé dans un processus de workflow. L'arbre de classification de cette classe est illustré par la figure 59. Les entités désignent les produits, les ressources humaines, le matériel et les logiciels utilisés, l'équipement pour l'emballage ou la containerisation pour l'expédition. Les entités désignent aussi les robots de la chaîne de manufacturation, les moyens de transports, de manipulation ou de stockage de marchandises. Au vu de cette diversité, nous pouvons définir deux sous-classes de l'entité, la *Resource* et l'*Item*.

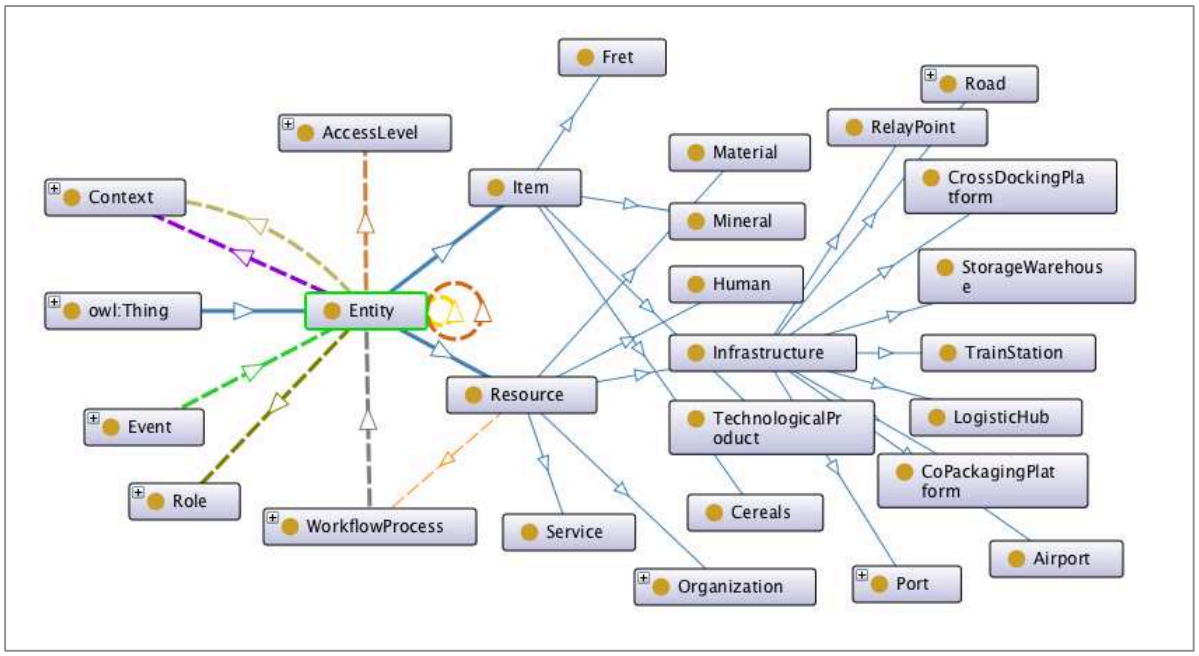


Figure 59. Arbre de classification de la classe Entity

- **Resource**

Le concept Ressource désigne tout moyen utilisé par le workflow pour atteindre ses objectifs. Ressource peut être une ressource matérielle, humaine, logicielle, ou tout autre moyen mis à la disposition du workflow pour son accomplissement. Ressource inclut tout moyen logistique qui permet de déplacer le flux d'un point de départ appeler origine (From) vers un point d'arrivée appelé destination (To). La classe des ressources peut à son tour être subdivisée en quatre catégories:

➔ les infrastructures de transport ou de transit (route, voies ferrées, voies de navigation, Hub de transit, Plateforme de co-packaging, etc.);

➔ les ressources humaines, qui sont des acteurs logistiques humains qui interagissent avec le flux logistique;

➔ les ressources matérielles que sont les moyens de transport, les conteneurs, les palettes, ou tout autre objet qui sert à l'accompagnement du flux de marchandises;

➔ les services, qui sont des services logistiques effectués par les opérateurs 3PL ou 4PL sur le flux de marchandises.

- **Item**

Un Item est l'entité sur laquelle le processus est réalisé. Dans un processus d'expédition, l'Item désigne la marchandise à livrer. Dans un processus de conditionnement à façon (*custom-packaging*), l'Item désigne la matière reconditionnée. Il est intéressant de catégoriser les marchandises selon leur nature : *fret* (viandes, poissons frais, produits laitier), *minéraux* (fer, charbon, cuivre, or, aluminium), *céréales* (riz, blé, orge, sarrasin, mil, etc.). Les produits de technologie comme les ordinateurs, les téléphones cellulaires sont classés dans la catégorie *TechnologicalProduct*. Ceci permet de raffiner les recherches dans une base de données.

- **Workflow process**

Le terme *WorkflowProcess* se refère à la description de l'évolution du flux de marchandises dans le temps, de son point d'origine à sa destination finale (figure 60). Il décrit le cycle de vie des entités business engagées dans le flux. Concrètement, il s'agit d'une séquence d'activités et de tâches, de transformations, d'opérations de transport, qui sont réalisées sur le flux au cours de son cycle de vie. Il décrit également le flux d'informations et d'événements qui déclenchent les tâches, qu'elles soient manuelles ou automatiques. Notre étude étant basée sur des architectures orientées données (*Data-Driven Architecture-DDA*) et des architectures orientées évènements (*Event-Driven Architecture-EDA*), notre principal objectif est de capturer la bonne information au bon moment et de la partager entre les acteurs logistiques pour la coordination du flux. Ainsi, nous définissons trois vues pour mieux décrire chaque aspect du flux de marchandises. Ces vues que nous appelons workflow level seront détaillées plus bas.

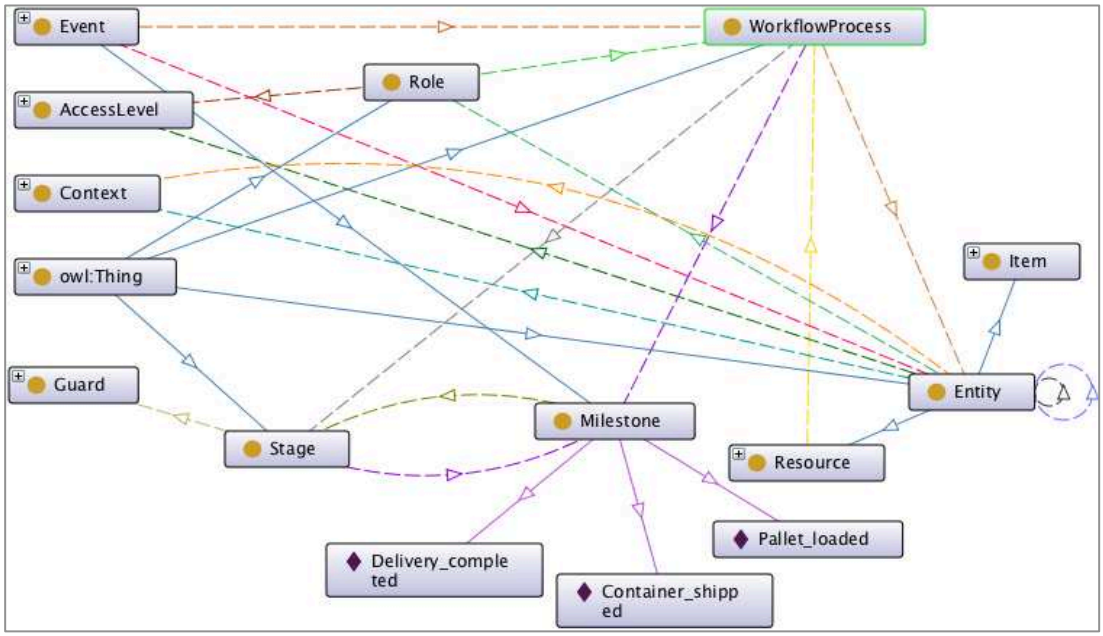


Figure 60. Ontologie du workflowProcess

- **Context**

Le contexte (***Context***) d'une entité logistique est le scope dans lequel il agit dans le workflow (figure 61). Le contexte concerne ce qui entoure le *business entity*, l'environnement dans lequel il évolue et communique. La notion de contexte est étroitement liée au rôle joué par l'entité dans le processus, ou dans le workflow. Dans cette étude, nous distinguons ce qui relève de la description statique de l'entité et de son contexte. La description statique réunit l'ensemble des caractéristiques qui ne peuvent pas évoluer dans le temps. Le contexte est l'ensemble des variables qui sont potentiellement soumises au changement, dont les valeurs changent dans le temps en fonction de l'évolution du workflow. Le contexte d'une entité peut changer en fonction du type de workflow, ou de son implication dans le processus. Parmi ces attributs, nous pouvons citer la position de l'entité, l'horodatage, l'opération logistique ou la tâche en cours d'exécution, l'acteur logistique qui réalise l'opération, etc. De ce fait, nous employons le terme *ContextVariable* pour désigner les caractéristiques qui changent en fonction du temps.

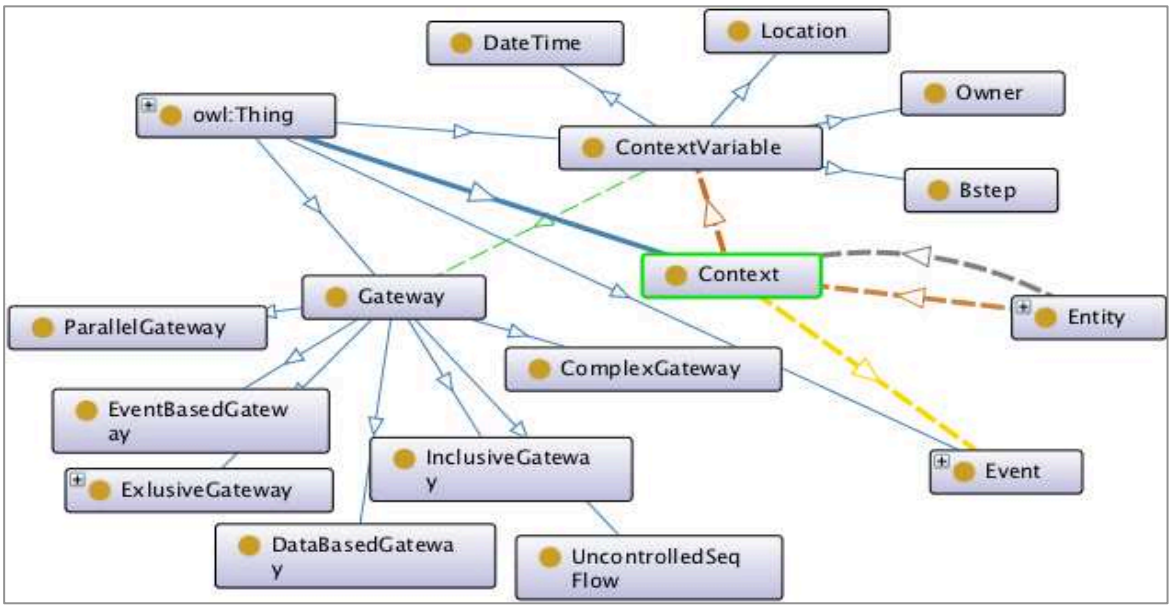


Figure 61. Ontologie du contexte du Business Entity

Les variables de contexte d'une entité sont mises à jour à l'aide d'événements qui agissent comme des stimulus qui transportent un message. En fonction de ces événements, la valeur de la variable change.

- **Event**

Un événement (***Event***) est toute cause qui peut perturber le contexte d'une entité et changer l'état d'une activité ou celui du workflow (figure 62). Par exemple: disponilbité d'une palette livraison d'un conteneur, fin d'un conditionnement. Les événements peuvent être classifiés en plusieurs catégories, les événements qui déclenchent une activité ou une tâche du workflow (*Guard*) ou condition de déclenchement, les événements qui marquent la

fin d'une activité ou d'un processus (*Milestone*). Ces événements agissent tous comme des triggers.

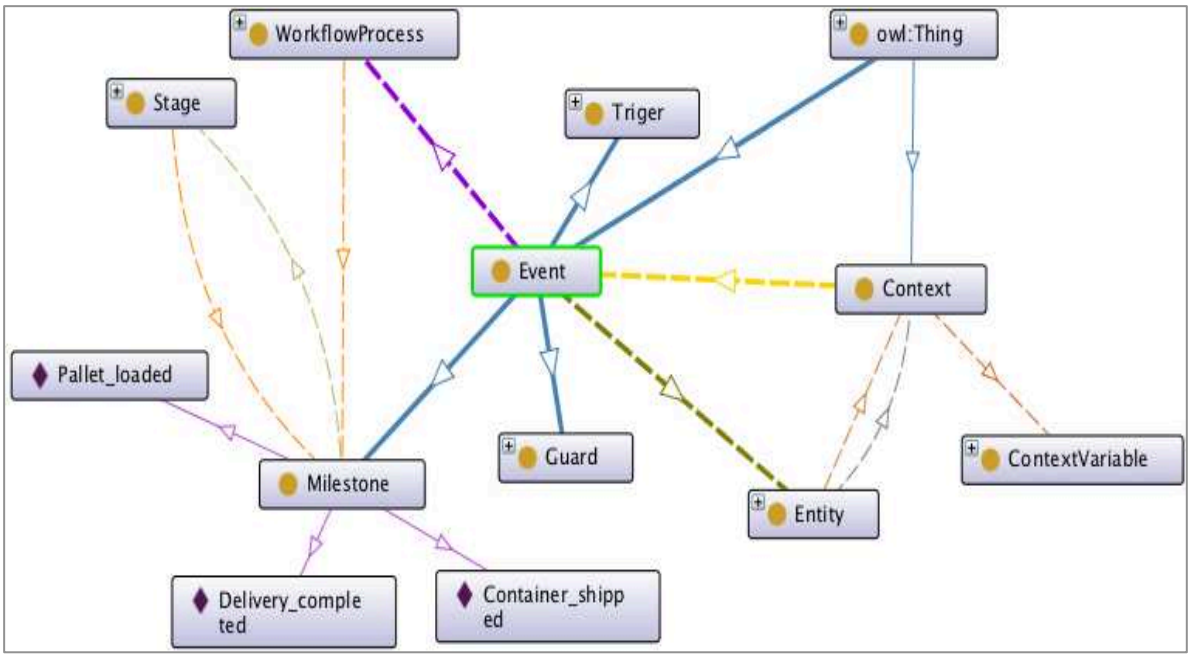


Figure 62. Ontologie des événements (Event)

- **Trigger**

Un trigger est un événement particulier qui opère le déclenchement d'une action, d'une tâche, d'une activité ou d'un processus (figure 63). Un Trigger peut servir aussi d'événement déclencheur à plusieurs activités simultanées, ou à une séquence d'activités. Parmi les Triggers couramment utilisés dans le business process, nous pouvons citer : les signaux (***Signal***), les messages (***Message***), les timers (***Timer***), les Triggers conditionés (***Conditional***), les points de compensation de flux (***Compensation***), les points d'exécution parallèle (***ParrallelMultiple***) et les erreurs d'exécution du workflow process (***Error***).

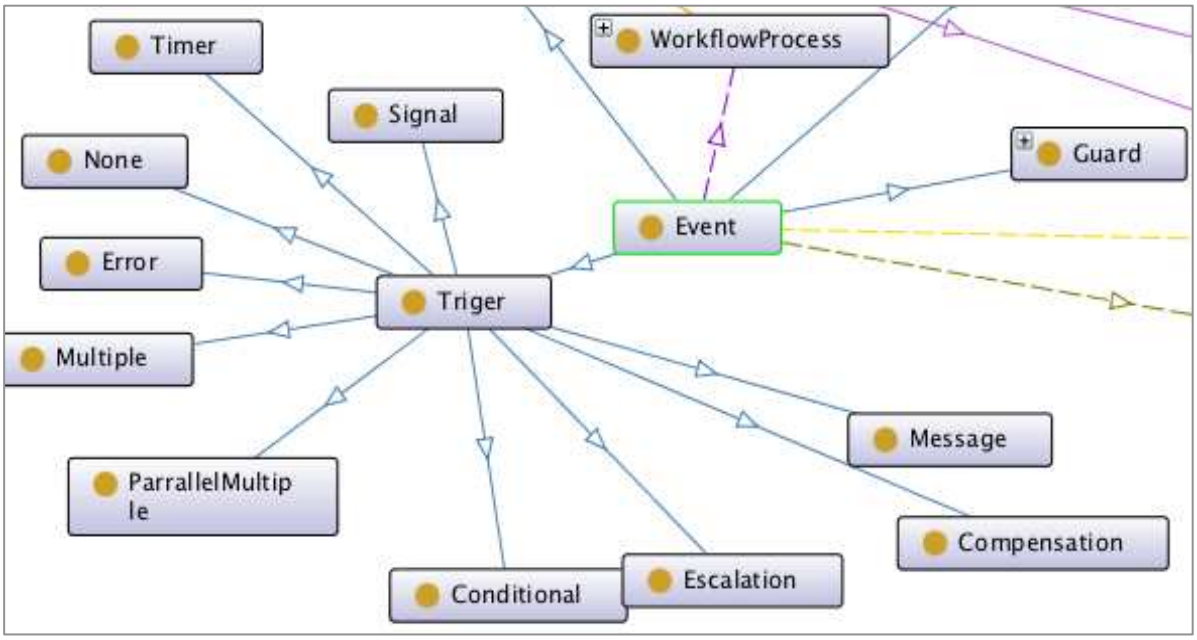


Figure 63. Ontologie des Trigers

- **Rôle**

Un Rôle dénote la fonction occupée par une entité dans le flux logistique (figure 64). Par exemple : Paul (Entity) joue le rôle de transporteur (Role) dans un flux d'approvisionnement (workflow) en matière première. De même, le *camion (Entity-Resource)* et le *cargo (Entity-Resource)* jouent le rôle de *moyen de transport (**Role**)*. Cette connaissance peut être necessaire pour vérifier si le moyen de transport selectionné est adapté au workflow en question. En fait, pour un flux d'approvisionnement de 100 palettes, la question est de savoir si le camion a la capacité suffisante (volume) pour transporter les 100 palettes. De même, nous pouvons faire les mêmes vérifications automatiquement pour le transport des conteneurs de freight, qui doit se dérouler sous certaines conditions de température et de pression. Ainsi, une fois les entités et les workflow décrits, le système pourra proposer une affectation optimale des ressources aux processus en analysant sa base de connaissance. Une entité peut jouer plusieurs rôles en fonction du contexte et du workflow. La relation ***hasRole*** qui lie une entité à un Role permet de définir un ou plusieurs rôles pour la même ressource. Le rôle définit la relation impliquée dans (***PlayIn)*** qui lie le rôle au workflow et permet d'assigner un rôle particulier au processus.

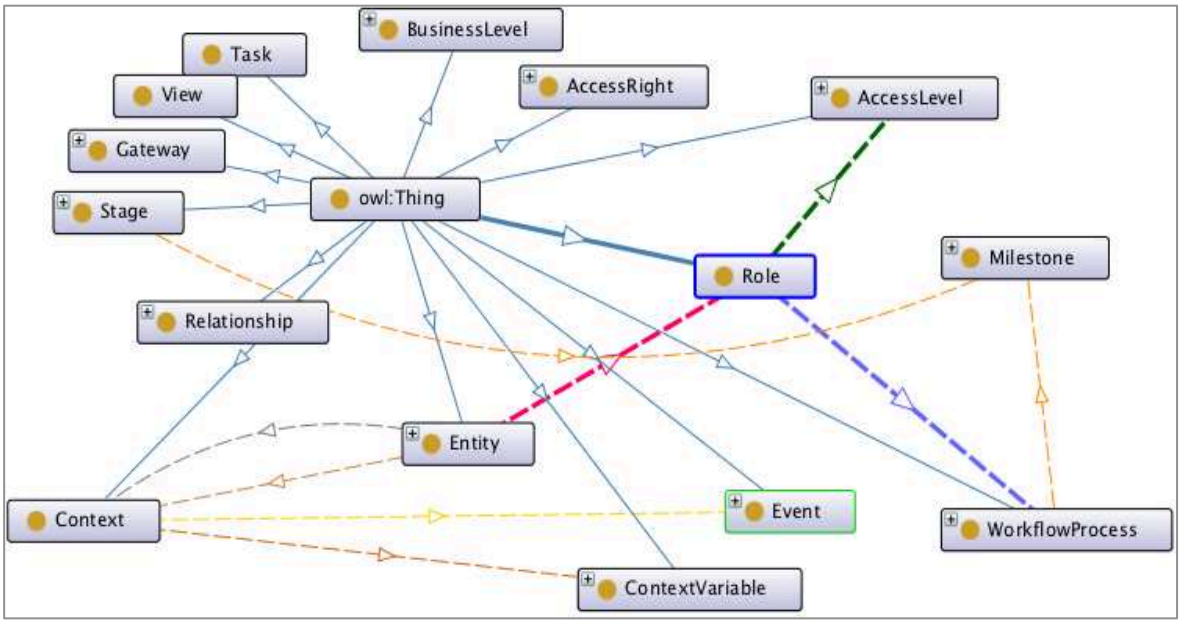


Figure 64. Ontologie du rôle du BU (Rôle)

- **Gateway**

Une gateway est une porte qui permet aux processus de communiquer (figure 65). Les processus peuvent donc envoyer et recevoir des messages à travers cette porte de communication. La similitude peut être faite à l'aide du réseau internet, dans lequel des machines communiquent à travers des passerelles. Un processus peut donc écrire sur une gateway ou lire à partir de cellle-ci. Un processus peut s'enregistrer au flux d'événements en provenance d'une gateway. Cette souscription permet au workflow d'écouter les événements émis par un autre workflow afin de faciliter les opérations de synchronisation. Il existe plusieurs types de gateways : ***InclusifGateway***, ***ExclusifGateway***, ***ParallelGateway***, ***EventBasedgateway***, ***ComplexGateway***, ***DatabasedGateway***, ***UncontroledSeqFlow***. Ces gateways sont des points de synchronisation du flux.

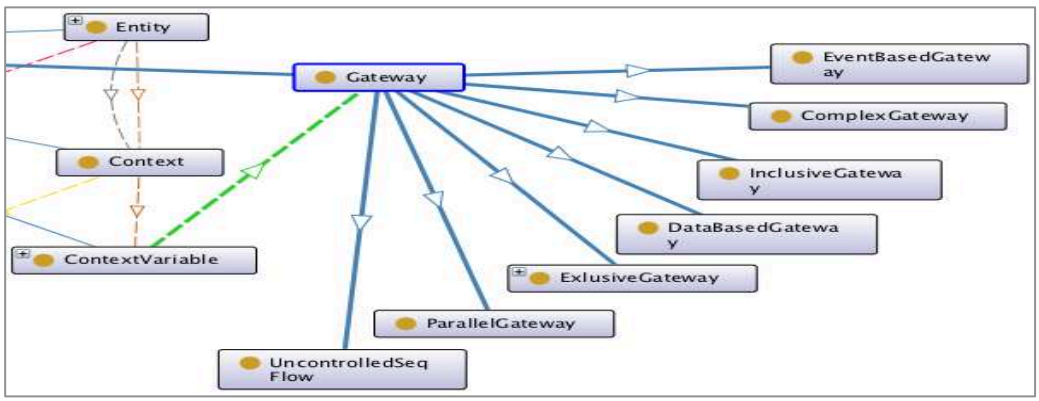


Figure 65. Ontologie du Gateway

- **AccessLevel**

Le concept niveau d'accès (*Accesslevel*) permet d'attribuer des droits d'accès à un

acteur ou à une entité logistique par rapport à son degré d'implication dans le flux. En réalité, il s'agit d'une vue qu'a l'acteur ou l'entité sur tout ou partie du flux, et plus précisément les informations liées au flux. Une vue est constituée d'un ensemble d'attributs du workflow qui peuvent être soit lus (*read*), mis à jour (*update*) ou permettre l'exécution du flux par déclenchement d'évènements (*execute*). Chacune de ces actions est associée à un droit d'accès correspondant.

**Représentation des relations entre les entités logistiques**

**Relationship**

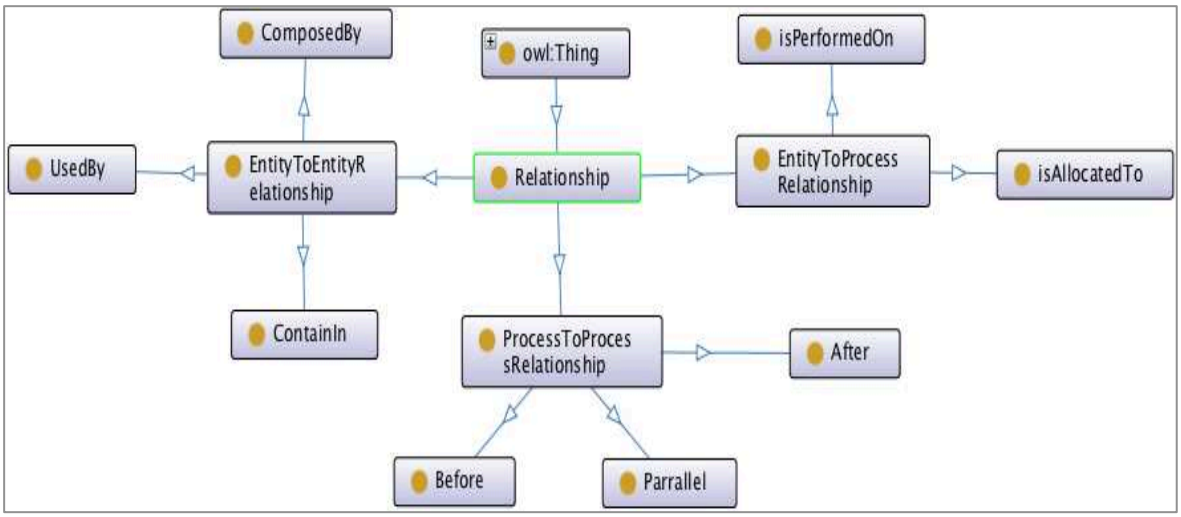


Figure 66. Ontologie des relations d'inter-dépendance (Relationship)

La notion de *Relationship* désigne un lien entre deux entités (*Entity-To-Entity Relationship*). Ce concept peut être utilisé pour désigner la relation qui lie une entité logistique et le flux (Entity To Process Relationship). L'autre catégorie de relations est celles entre deux tâches ou sous-processus du workflow process (*Process-To-Process Relationship*). Le concept de relationship est repris du modèle des réseaux sociaux dans lequel, les entités sont liées par des relations multiformes. Le shéma de la figure 66 illustre ce concept de Relationship.

Dans le flux logistique, les éléments du flux sont liés par des relations muti-formes. Nous pouvons ajouter à une relation des caractéristiques pour préciser le poids de la relation, la durée de celle-ci, ou tout simplement les dates de début et de fin de la relation. Les relations entre entités logistiques sont subdivisées en trois sous-familles. Les relations dont la sématique dénote un lien d'usage (***UsedBy***), par exemple : un camion (Entity) de transport qui utilise 20 litres de gazoil (Resource) pour transporter une marchandise (Process). Nous dirons donc que le gazoil a été utilisé (*usedBy*) par la resource camion dans le cadre du processus de transport des marchandises.

Nous utilisons le concept ***ComposedBy*** pour représenter le deuxième type de relations qui expriment un lien de composition entre les entités. Par exemple, un conteneur est composé des palettes qui sont composiées à leur tour des cartons. La dernière catégorie de relations est celles où l'on veut dénomer une relation de contenu-contenant. Pour cette

sémantique nous utilisons le terme ***ContainIn***, qui exprime le fait qu'une entité logistique soit contenue dans une autre entité. Cette dernière classe de relations permet de mettre à jour le contexte du contenu en fonction du contexte du contenant. Par exemple : les palettes sont contenues dans le camion. Les conteneurs sont contenus dans le cargo.

## 4.5. Conclusion

Dans ce chapitre nous avons fixé comme objectif de proposer un modèle de connaissance pour le flux logistique, en particulier pour la gestion du contexte des entités impliqués dans le flux.

L'ontologie proposée avec sa syntaxe et sa sémantique riche nous permet d'atteindre trois objectifs spécifiques. Il s'agit de l'uniformisation des termes dans le contexte d'une collaboration entre acteurs logistiques, la mise en place d'un outil qui facilite l'interopérabilité des systèmes qui manipulent les données et les documents sur les entités du flux, et enfin d'avoir un outil d'extraction de connaissances dans le réseau d'entités et de flux logistiques. L'intégration avec des systèmes multi-agents apporte une spécificité dans la mesure où des agents logiciels se préoccupent de tâches élémentaires comme la mise à jour des données et la synchronisation au niveau des clusters d'entités logistiques.

Dans le prochain chapitre, nous proposons une plateforme de collaboration basée sur les technologies de l'IoT. Nous expliquons dans les détails la mise en place du réseau de capteurs et le stockage des données dans les plateformes Cloud à des fins de partage et d'interopérabilité.

# Chapitre 5: Plateforme collaborative basée sur l'Internet des Objets

## 5.1. Introduction

Dans ce chapitre nous traitons le suivi et le tracking des objets logistiques, le traitement et le partage des données récoltées, la gestion des droits d'accès et les interactions entre tous les acteurs impliqués dans le flux [126, 123]. Les plateformes existantes ont des difficultés à résoudre ce problème sur certains points clés : recueillir des données directement à partir de capteurs intégrés dans les objets logistiques pour le traitement en temps réel et la notification ; définir une politique commune et un protocole de communication pour toutes les parties prenantes ; gérer l'interopérabilité entre les acteurs qui utilisent des technologies d'Internet hétérogènes.

La valeur ajoutée de l'architecture proposée est principalement l'intégration des différentes couches de l'Internet des Objets : la couche capteurs, la couche de transmission de données, la couche de stockage dans le Cloud et enfin la publication des données et événements recueillis vers les utilisateurs. Cette architecture a pour objectifs de faciliter le partage d'informations sur les flux logistiques pour la traçabilité et la collaboration entre les différents acteurs de la chaîne d'approvisionnement. Ces exigences sont les principaux défis pour les entreprises dans le domaine de la gestion des flux, la chaîne d'approvisionnement collaborative et la Business Intelligence (BI). En outre, pour chaque couche du système nous proposons un modèle d'interaction basé sur l'approche Guard-Stage-Milestone (GSM).

Le reste du chapitre est divisé en trois parties. La première expose un cas d'utilisation où nous décrivons un scénario dans lequel des entreprises doivent collaborer pour répondre à un besoin d'approvisionnement. La deuxième partie présente l'architecture de la plateforme avec des détails sur ses composants et leurs interactions. La troisième partie montre quelques résultats expérimentaux obtenus à partir des simulations effectuées.

## 5.2. Architecture générique et fonctionnement de la plateforme collaborative

### 5.2.1. Architecture de la plateforme collaborative

Cette section décrit plus en détails les composants et services de la plateforme, ainsi que les technologies d'implémentation mises en œuvre. Nous nous focalisons particulièrement sur les tâches principales du middleware comme l'identification des produits, le géo-positionnement, le tracking et la traçabilité, les protocoles de communication et la transmission des données récoltées. La figure 67 présente l'architecture globale de la solution proposée.

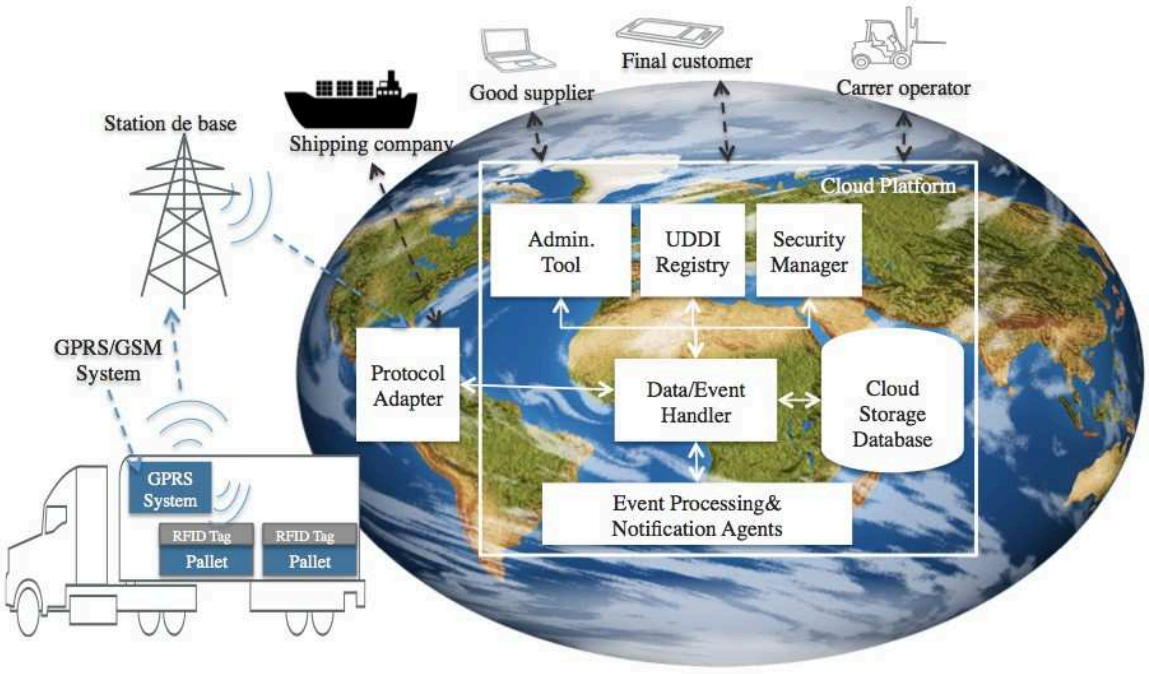


Figure 67. Architecture globale de la plateforme collaborative

Dans les paragraphes suivants, nous présentons la mise en œuvre des principaux composants de cette architecture.

### 5.2.2. Identification et tracking des objets logistiques

Les informations que nous voulons récolter sur le produits sont classés selon quatre points, Quoi, Où, Quand, Pourquoi (*What, Where, When, Why).* Ces quatre questions résument les principales attentes des acteurs, et les informations dont ils ont besoin pour piloter le flux. Comme cela est illustré dans la figure 67, nous supposons que la palette est équipée d'une puce RFID contenant des informations sur le produit: la marchandise contenue dans la palette, l'origine et la destination de la palette, le poids, la description du contenu de la palette, sa dangerosité, les produits incompatibles. Lorsque la palette est prête

pour le chargement, le lecteur RFID lit les informations et les envoie à une plateforme Cloud à l'aide d'un système de transmission tel que le *GPRS / GSM* ou le réseau *SIGFOX* [111].

Le modèle GSM de figure 68 décrit ce processus d'identification. L'activité principale (Stage) nommé *"RFID tag reading and transfert"* est activée par la combinaison de deux événements ou Guards *"Palet ready for loading" et "Transportation car arrived"*. Lorsque cette étape est activée, les tâches internes *"Read RFID tag information"* et *"Send information through GPRS"* sont activées automatiquement avec les mêmes Guards. L'accomplissement des deux tâches déclenche l'exécution du deuxième stage *"Platform verification and confirmation"*. Cette activité est déclenchée lorsque le bus reçoit une requête du lecteur RFID signalant que les informations sont disponibles. L'activité est subdivisée en deux tâches complémentaires: *"Information checking"* pour le prétraitement et la vérification des données, et *"Users Notification"* pour la notification aux utilisateurs de ces événements. L'activité principale atteint ainsi ses objectifs (Milestones) lorsque le Milestone ou *"pallet identified and users notified"* est atteint. Après l'acquisition des données, l'information doit être stockée pour des usages variés. La section suivante montre comment nous organisons ce processus de traitement et de stockage.

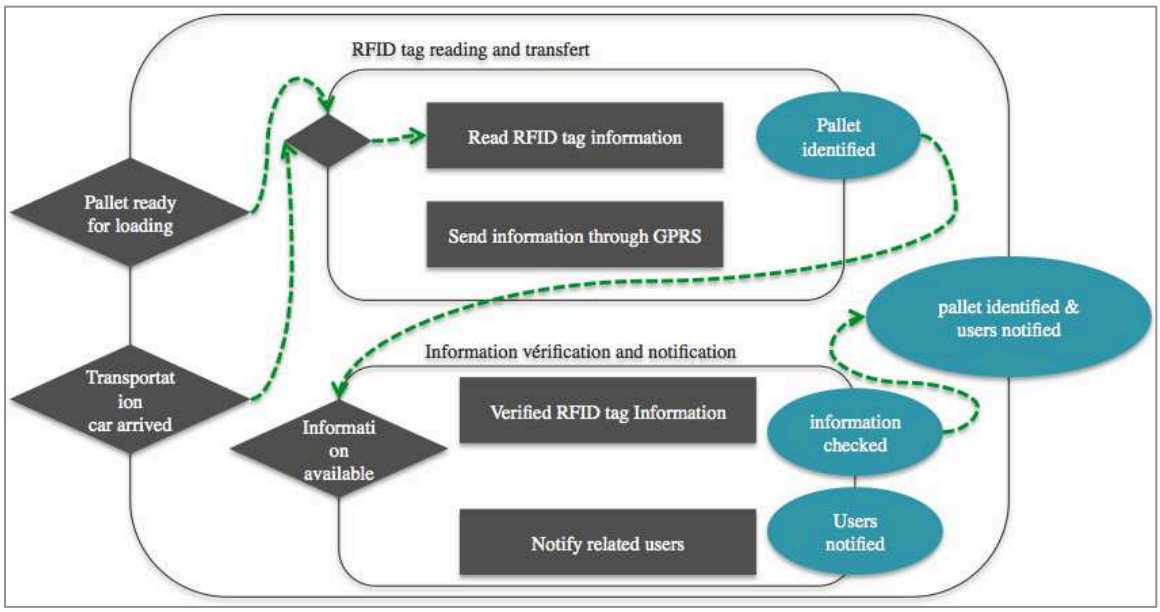


Figure 68. Modèle GSM du processus d'identification de la palette

### 5.2.3. Stockage de données et traitement d'événements complexes

L'architecture de la figure 69, que nous proposons pour la plateforme de stockage est orientée événement *(EDA, Event-Driven-Architecture)*. C'est une extension des Architectures Orientées Services *SOA (Service-Oriented-Architecture)* [112] par l'ajout d'une couche de gestion et de traitement d'événements *(Event processing)*. La plateforme est constituée de cinq principaux composants :

- Le module *Data/Event* handler responsable de la capture des événements et messages venant de sources diverses et hétérogènes (lecteur *RFID*, tablette, smartphone, PC, etc.).
- Le module *Cloud Storage Database,* qui sert de plateforme de stockage des données issues des tags RFID ou des utilisateurs.
- Le module *Event Processing & Notification* qui joue le rôle d'un gestionnaire d'événement complexes, par combinaison de plusieurs sources d'événements. Il avertit également les utilisateurs abonnés dès qu'un nouvel événement est arrivé.
- Le module *UDDI Register* joue le rôle d'un registre de services disponibles sur la plateforme. Il permet aux utilisateurs de découvrir les services disponibles et de s'abonner aux les listes de diffusion d'événements.
- Le module de sécurité *Security manager*. qui gère les droits d'accès aux données et aux composants de la plateforme. Il est responsable de filtrer l'accès aux données en fonction du profil de l'utilisateur et garantie la confidentialité.

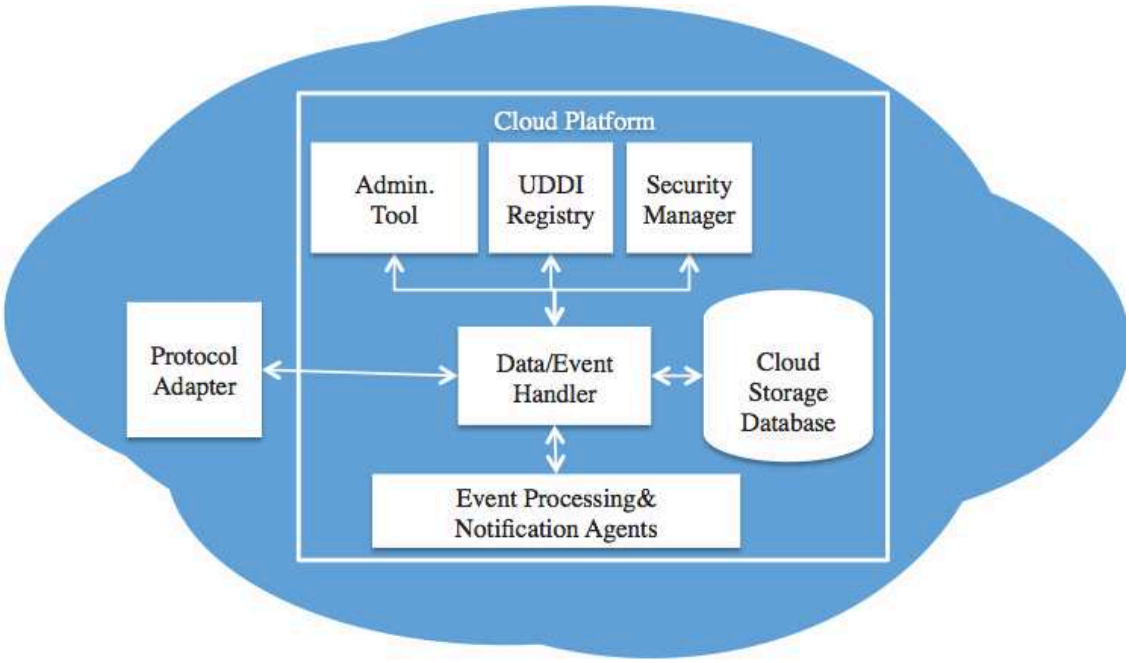


Figure 69. Architecture de la plateforme de stockage Cloud

- **Data/Event Handler**

Le gestionnaire de données et d'événements se compose de deux parties comme illustré par la figure 70: le *Request Handler* chargé des requêtes, et le *Data Processing* chargé du traitement des données. Le *Request Handler* capture les événements (Lecture de tags RFID ou requête utilisateur) et transfère ces messages au module *Data Processing*. Les tâches principales de ce module sont : 1) déterminer le type de requête. 2) Extraire les données des enveloppes 3) vérifier le format et la cohérence des données 4) enregistrer les données dans la base de données partagée. Après ces opérations, le service de notification est appelé automatiquement pour informer les utilisateurs du changement de contexte.

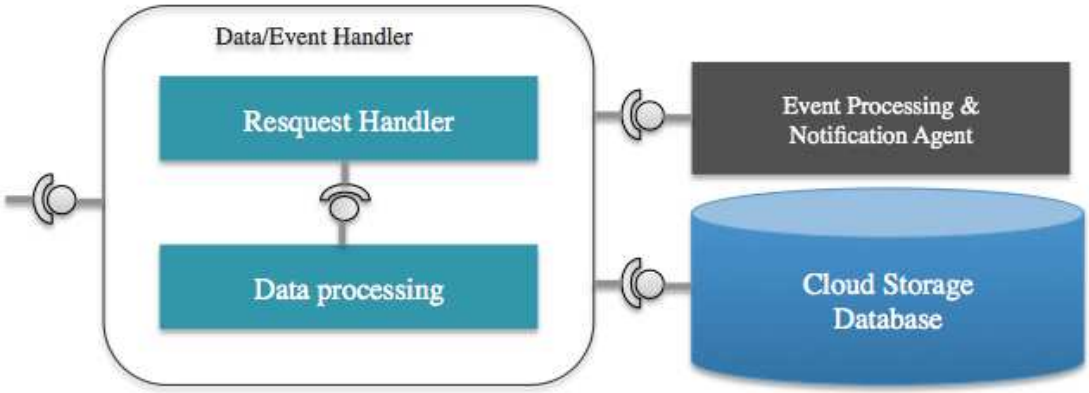


Figure 70. Architecture du composant Data/Event handler

- **Cloud storage database**

Comme mentionné précédemment, le stockage en mode Cloud est recommandé ici pour ses critères *d'évolutivité*, de *disponibilité*, de *reconfigurabilité* et *accessibilité* sur n'importe quelle plateforme à partir de n'importe quel terminal (*anywhere, anyhow*). Le Cloud a aussi cette capacité de stocker de grand volume de données (Big data). Comme nous l'avons mentionné dans les chapitres précédents, les systèmes de gestion traditionnels possèdent des données disséminées dans plusieurs bases plus ou moins distribuées, gerées par des *SGBD*. L'approche *Cloud*, bien que recommandée, a des défis à relever parmi lesquels la sécurisation des données, (contrôle d'accès, authentification, autorisation), l'hétérogénéité des protocoles et la forte volumétrie des données. Pour réaliser ses tâches, la plateforme doit stocker des informations sur des entités business (BE), les événements liés à ces BE et les utilisateurs concernés.

Nous proposons un schéma de données NoSQL pour stocker les données de ces entités. En effet, les données provenant des objets logistiques n'ont pas une forte relation d'interdépendance si on raisonne à l'échelle de l'objet. En revanche, les entités impliquées dans le même flux peuvent avoir une interdépendance forte selon le contexte, ainsi que les données associées. Nous avons choisi Google Cloud Datastore pour stocker les entités. Cette plateforme assure la réplication, la gestion de grands volumes de données, la

performance et la disponibilité des données[3]. Les différents packages du modèle des entités ont été discutés dans le chapitre sur la vue statique du flux logistique. Dans ce chapitre nous présentons les parties propres au middleware.

Les événements issus du flux sont des signaux envoyés par les acteurs logistiques en utilisant des canaux différents (lecteur RFID, smartphones, etc). Un événement donné est lié à une ou plusieurs BE ou à des opérateurs impliqués dans le pilotage du flux. Un *Business User* est une organisation ou une personne partageant des données ou des services pour la collaboration ou l'interopérabilité des systèmes qui pilotent le flux. Cet utilisateur peut aussi être un opérateur impliqué, qui a besoin d'information pour le suivi du flux ou de l'objet.

Une structure *Hashmap* est utilisée par la plateforme pour stocker les URI des services mis à disposition par les opérateurs. Ainsi, pour chaque opérateur, la plateforme stocke son identifiant unique comme clé de la hashmap et une liste de codes électroniques représentant les BE, dont il peut accéder au contexte. Une autre difficulté de l'intégration est l'hétérogénéité des formats d'échanges des acteurs ayant des sources de données diverses *(Oracle, DB2, MySQL, etc.)*. Pour répondre à cette problématique, la plateforme joue le rôle de bus orienté service avec une fonction de routage. Nous proposons pour cela d'utiliser le même format d'échange sur la plateforme (JSON). Pour le pooling des données, nous utilisons la technique de compartimentation pour séparer les données de chaque acteur de sorte à éviter les conflits et pertes de données. Cette technique permet également à chaque partenaire de définir sa propre politique de contrôle d'accès et de sécurité sur les données qu'il partage via la plateforme. Techniquement, la réalisation est faite par la plateforme Cloud de Google.

- **Registre UDDI pour l'abonnement et la découverte des services web (UDDI)**

*UDDI* (*Universal Description Discovery and Integration)* est un annuaire pour l'enregistrement et la découverte de services web. Il permet de localiser un service donné sur Internet. L'usage d'un annuaire de services est dû au choix architectural (SOA). Ainsi, les services de la plateforme s'enregistrent dans un annuaire pour faciliter leur découverte par les autres acteurs du système. Ce composant du middleware permet l'enregistrement des services web. Il spécifie la manière dont les services et les autres applications propriétaires (portails d'entreprise) peuvent s'abonner au contexte des entités afin d'être notifiés automatiquement lorsque le contexte change.

- **Traitement d'événements complexes (CEP)**

Quand un événement est déclenché, (lecture d'une étiquette RFID, commentaires ajoutés à une palette, mise à jour les coordonnées GPS, changement de température, etc.) ce module informe les utilisateurs et les parties prenantes concernées. Ainsi, les acteurs de la chaîne d'approvisionnement qui collaborent ensemble pour piloter la marchandise peuvent prendre des décisions importantes en temps réel, en particulier en cas de problèmes (conteneur bloqué, palettes non conformes à la réglementation, etc.). Le traitement d'événements complexes permet de combiner plusieurs sources d'événements pour générer

[3] *https://Cloud.google.com/storage*

de nouveaux événements, afin d'améliorer la prise de décision. Ainsi, si les conteneurs arrivent en retard à la plateforme portuaire par rapport à la date initialement prévue, il est inutile de mobiliser les remorqueurs pour attendre leur arrivée, par rapport au planning prévisionnel. La plateforme peut donc préconiser aux remorqueurs de retarder leur arrivée à la plateforme portuaire.

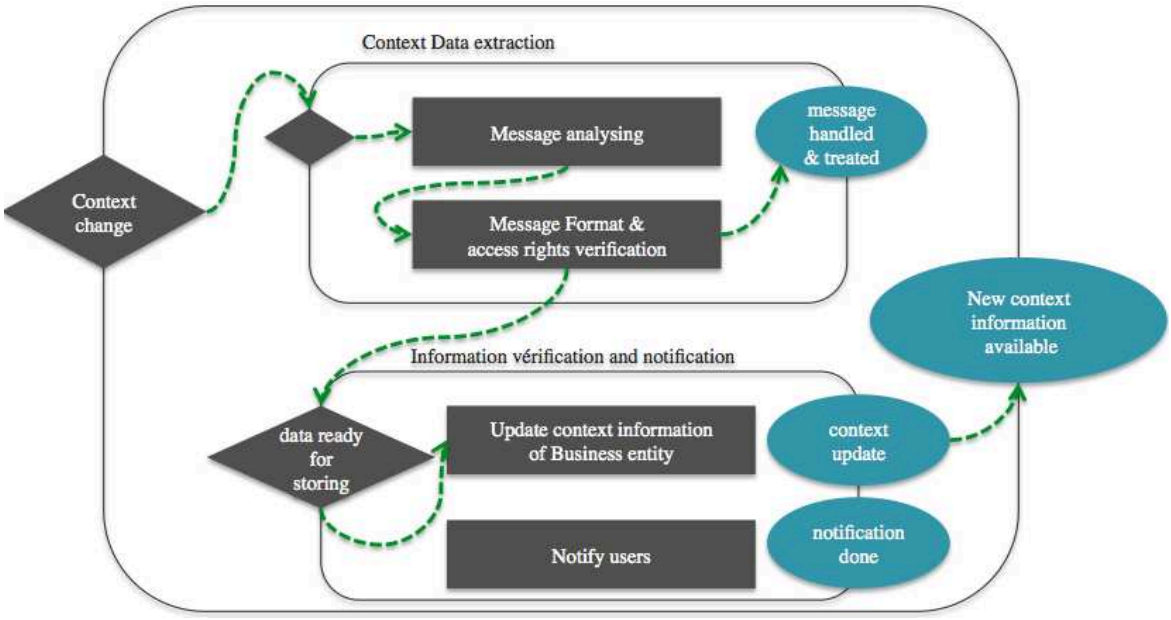


Figure 71. Traitement d'évènements et notification

Le diagramme GSM de la figure 71 décrit le fonctionnement dans le cas d'une notification. Les données passent par le réseau avant d'atteindre la plateforme Cloud. La section suivante explique le fonctionnement de cette transmission réseau.

Pour la publication d'événements, nous utilisons le pattern *Publish/Subscribe* qui permet à des utilisateurs de s'abonner à des sources d'événements (web services, ...) et d'être notifié automatiquement par des alertes quand le contexte de l'entité change. Le pattern *Publish/Subscribe* diffère des autres patterns d'interopérabilité EDA. En effet, une seule souscription permet de recevoir plusieurs notifications sans avoir à envoyer une nouvelle requête au service producteur d'événements. De ce fait, le pattern Publish/subscribe semble être le candidat approprié pour la gestion des événements venant de sources multiples, et consommer par des utilisateurs multiples et hétérogènes [113]. L'usage de ce pattern est aussi mentionné dans le *Business Process management (BPM)* où des services clients peuvent souscrire aux services fournisseurs et publier des ordres de commandes. Le service fournisseur enverra donc des événements de notifications aux clients pour le suivi de leurs commandes [114]. La section suivante explique comment cette transmission réseau fonctionne et introduit les protocoles de communication utilisés.

### 5.2.4. Protocole de communication et de transmission de données

De nombreuses technologies de communication sont développées dans la littérature [72, 115]. Dans ce qui suit, nous nous concentrons sur la transmission de l'information en utilisant deux technologies de communication adaptée à la mobilité : les réseaux cellulaires GPRS et SIGFOX, ce dernier étant particulièrement dédié à l'Internet des objets. Une étude comparative de ces technologies est présentée à la fin du paragraphe.

- **Transmission par le réseau GPRS**

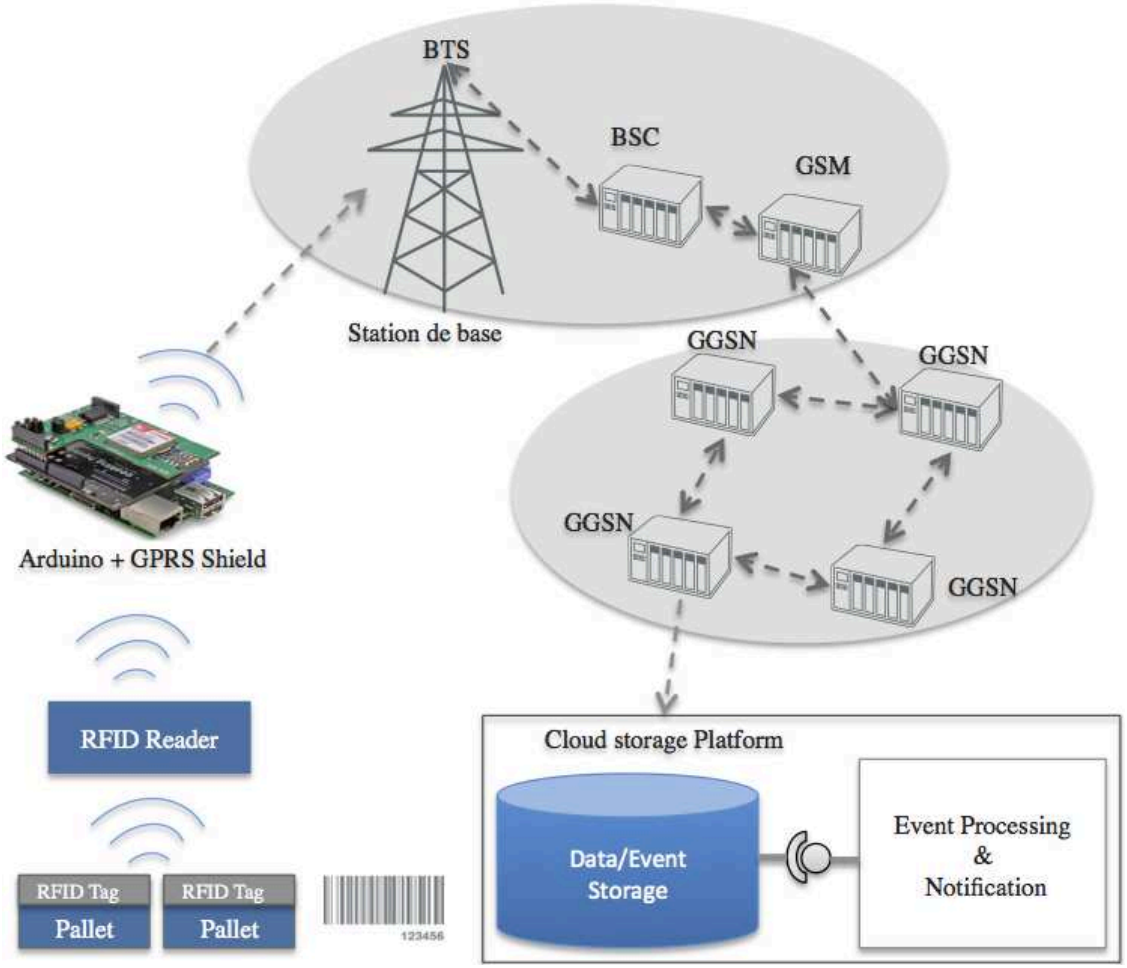


Figure 72. Transmission de données via GPRS

Comme le montre la figure 72, une communication bidirectionnelle est créee entre la carte Arduino équipée d'un module de communication *GPRS* et la plateforme Cloud à travers le réseau *GPRS*. Les informations lues à partir de la carte *RFID* sont transmises à la carte Arduino. Le message est ensuite envoyé à travers le réseau *GSM / GPRS* à la plateforme.

Étant donné que les modules d'acquisition et de traitement utilisés ont une charge de batterie faible et de même que leur espace mémoire *(Arduino mega: 256 Ko de mémoire flash)*, il est nécessaire de définir un encodage raisonnable de l'informations à transférer. Le schéma de la figure 73 montre le format de la trame que nous proposons. Elle se compose de trois parties : la première fournit des informations sur les données EPC. Pour cette première partie, nous devons respecter le standard EPC. La deuxième partie de la trame est

utilisée pour envoyer des informations sur l'environnement de la palette comme la température, la pression et l'humidité. La troisième partie de la trame contient des informations sur la position GPS de l'entité logistique, sa longitude et sa latitude.

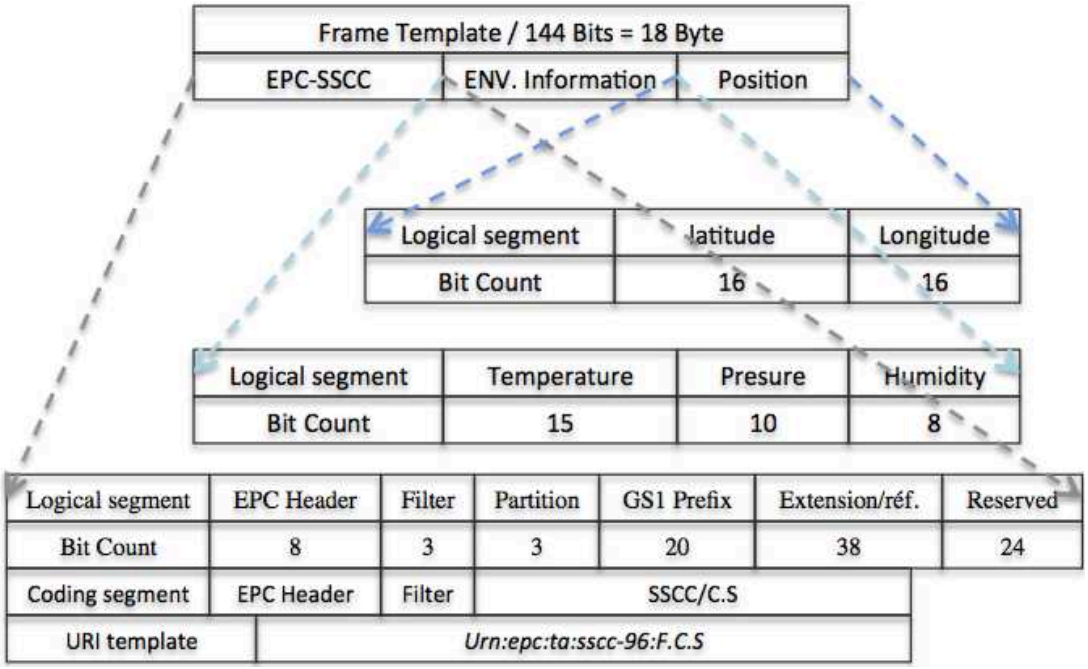


Figure 73. Format de la trame

- **Transmission par le réseau cellulaire Sigfox**

Une autre voie pour la transmission de données est l'utilisation du réseau cellulaire SIGFOX. Sur la figure 74, nous présentons l'architecture du réseau SIGFOX. Pour nos expérimentations, seules les cartes de développement *Telecom Design Development Board (TD)* sont capables de communiquer directement avec le réseau *SIGFOX* et la plateforme Cloud de Telecom Design *(réseau de services web pour les capteurs)* [111]. Il nous est donc impossible au moment de nos travaux de connecter un autre type de carte *(Arduino, Galileo, Raspberry PI, etc.)* directement avec le réseau SIGFOX actuel. Cependant, il nous est possible d'envoyer des commandes standard AT vers une carte Telecom Design (TD1208, TD1202, TD1204 et TD1207) pour s'en servir comme passerelle, comme le montre la figure 75. Côté Cloud, la plateforme SIGFOX dispose d'un service de *call back* pour récupérer automatiquement les données enregistrées sur la plateforme par ses utilisateurs.

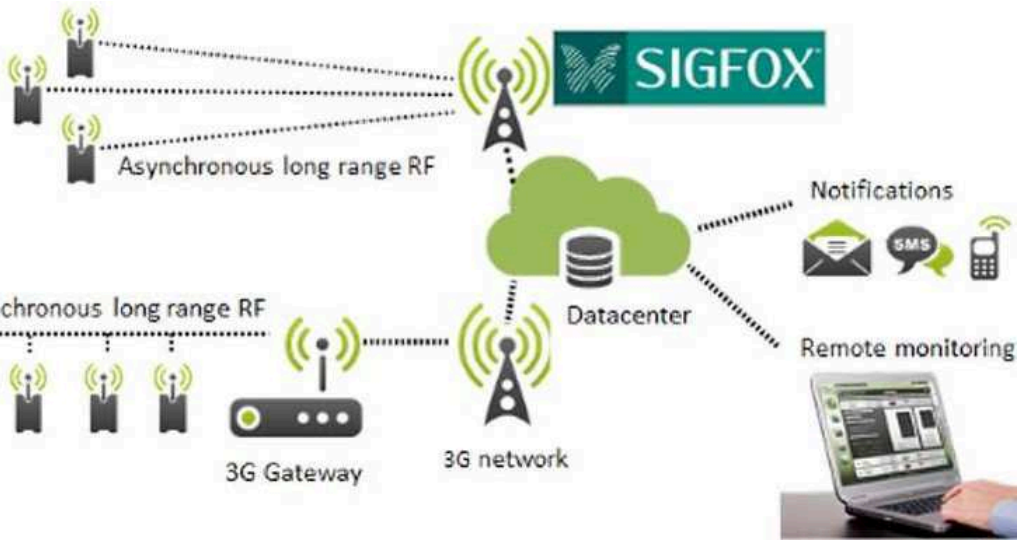


Figure 74. Architecture du réseau Sigfox[4]

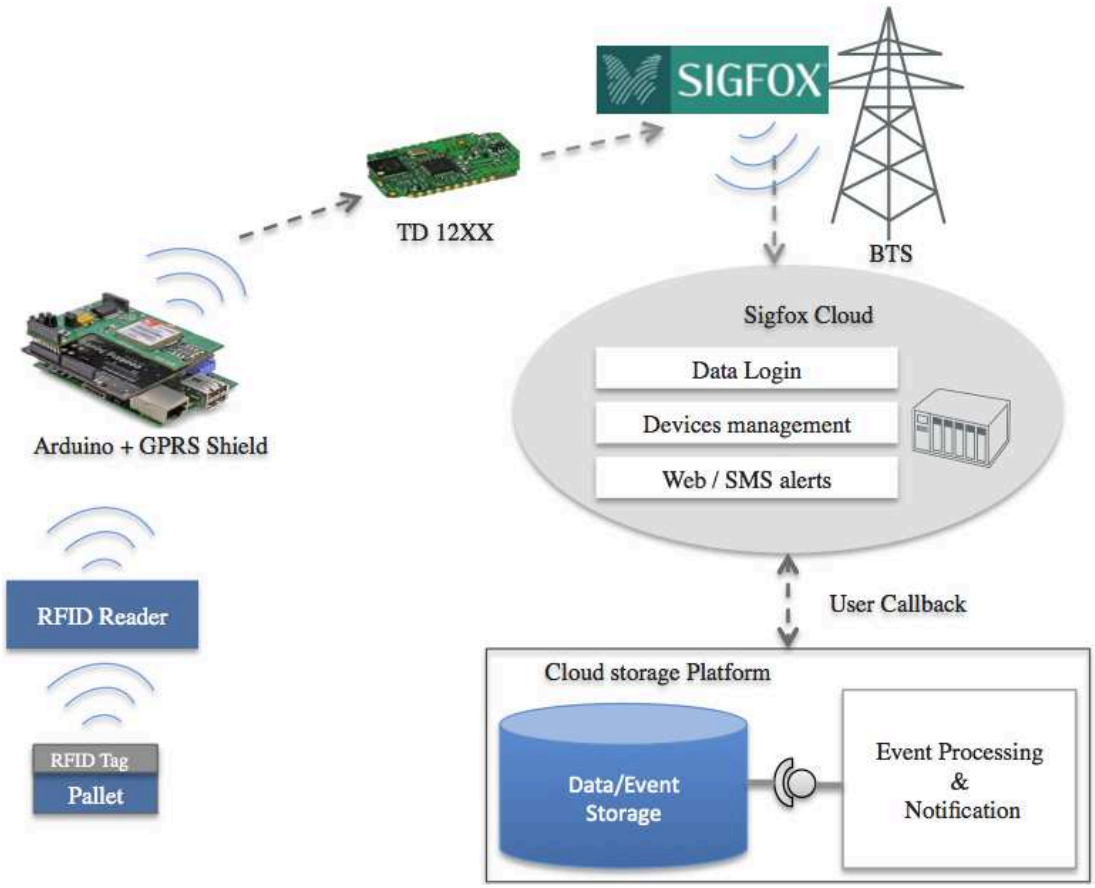


Figure 75. Transmission de données utilisant le réseau Sigfox et un module Arduino

*NB : Snootlab Akeru a développé un prototype de carte incorporant un module SIGFOX sur Arduino*

[4] *http://makers.sigfox.com/about/*

- **Comparaison technique des deux réseaux de transmission**

Le tableau 4 montre que l'avantage principal de GPRS est la possibilité d'envoyer un nombre illimité de messages, d'une taille légèrement supérieure à ceux de SIGFOX. Le coût annuel d'un abonnement GPRS est cependant très élevé par rapport à celui proposé par SIGFOX. Cette comparaison montre que SIGFOX est mieux que le GPRS dans le cas où il est destiné à connecter plusieurs objets dépassant les centaines. Cet avantage est renforcé par une plateforme de nuage que SIGFOX met à la disposition de ses clients.

**Tableau 4: Comparaison entre Sigfox et GPRS**

| Critère | GPRS | Sigfox |
|---|---|---|
| Messages/Jour | ∞ | 143 |
| Taille max d'un message | < 1500 Byte | <10 Byte |
| Plateforme Cloud intégrée | Inexistante | Offerte |
| Coût/objet | < 65€ | < 15€ |
| Coût abonnement/unité/année | < 240€ | < 155€ |
| Couverture réseau | Mondiale | Nationale |

- **Utilisation du GPS pour le tracking des conteneurs**

Le suivi de la position d'une marchandise est très important, surtout en cas de perte ou de vol. Le système de navigation par satellite (*GNSS- Global Navigation Satellite Systems*) permet le suivi des marchandises en temps réel dans la logistique de transport.
L'incorporation d'un capteur *GPS* permet ainsi d'avertir le propriétaire et ses collaborateurs sur la bonne position de l'entité logistique, dès le chargement de la marchandise jusqu'à la livraison au client final. A cet effet, nous utilisons le format de données du système géodésique[5] WGS 84 (en degrés décimaux) pour les coordonnées *latitude* et *longitude*.

[5] *http://www.ign.fr/sites/all/files/geodesie_systemes.pdf*

### 5.2.5. Cas d’utilisation

Pour mieux comprendre l’usage d’une telle plateforme, nous présentons dans ce paragraphe un cas générique de collaboration entre les acteurs de la chaîne d’approvisionnement (voir la figure 76), semblable à ceux décrits dans le cadre du projet support à nos travaux.

Nous considérons le scénario dans lequel un conteneur doit être rempli et expédié. Un opérateur 4PL a un conteneur à remplir avec une liste de produits. Lorsque les marchandises sont toutes remplies dans le conteneur, il est galvanisé et transféré au port. Il est ensuite transporté par bateau et livré au centre de distribution. Lorsque le conteneur arrive au centre de distribution, les produits sont dégroupés et expédiés à des clients finaux via l'opérateur de transport terrestre.

Nous considérons que les produits proviennent de différents fournisseurs. Les opérateurs de transports peuvent être tout aussi différents que les clients finaux et ne sont pas nécessairement dans la même zone géographique. Cet exemple montre la complexité de coordination des activités sur des flux logistiques et la chaîne d'approvisionnement. Notre objectif est d'appliquer notre middleware de services, orienté Cloud et *EDA* pour gérer et partager des informations entre les acteurs concernés. La plateforme est également en charge de la gestion de tous les événements émis par le flux de marchandises et les acteurs en temps réel. Le bus est également responsable de la notification automatique à tous les acteurs. Cela conduit à faciliter la coordination, le suivi et le pilotage du flux de marchandises.

L'opérateur 4PL peut suivre l’évolution des différentes phases du processus via la plateforme : remplissage du conteneur, expédition, distribution des produits au client final. Le scénario décrit est résumé dans le diagramme BPMN de la figure 76, dans lequel nous pouvons observer la capture d’événements par la plateforme et la notification automatique aux acteurs impliqués dans le processus.

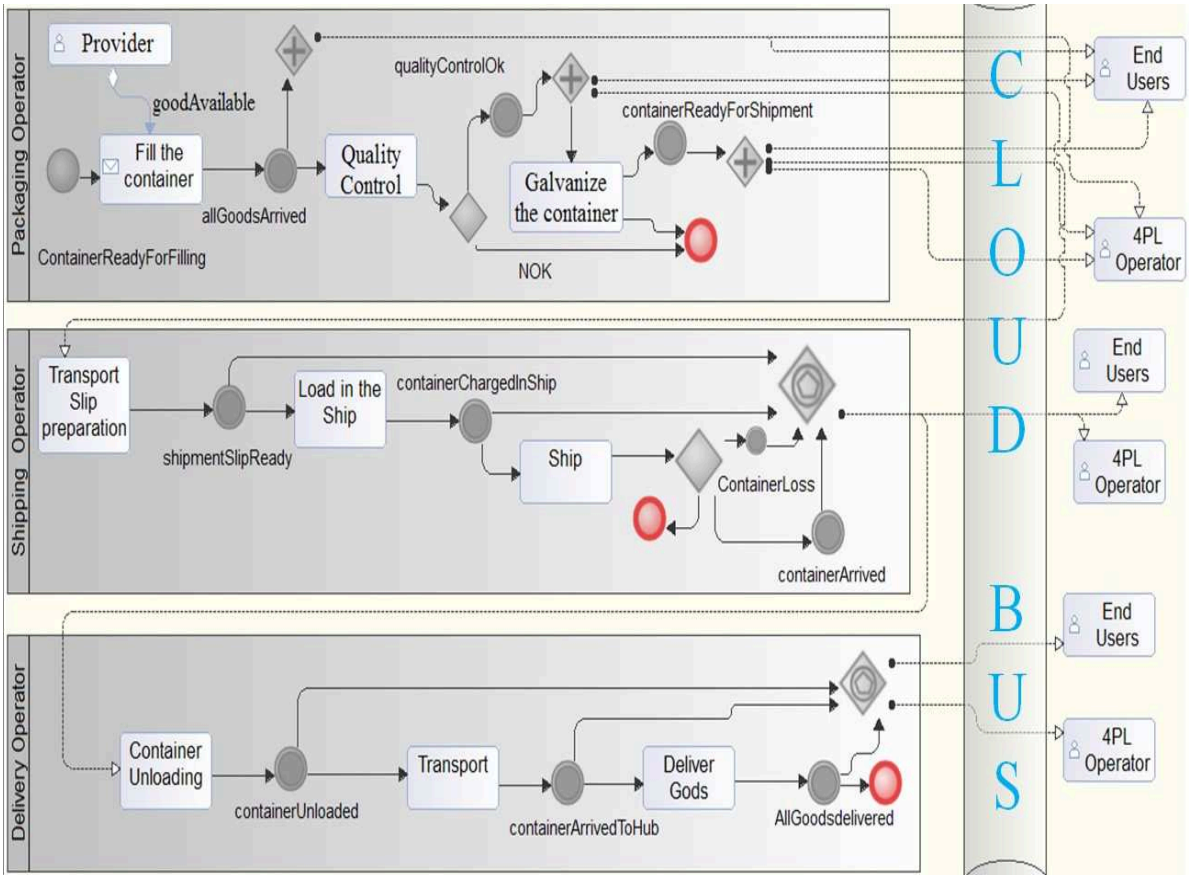


Figure 76. Diagramme BPMN du cas d'utilisation

## 5.3. Mise en œuvre de la plateforme collaborative et expérimentation

### 5.3.1. Premières expérimentations avec Firebase et Arduino

Pour cette première phase du prototype, nous simulons le réseau d'entités logistiques. Pour chaque entité, nous nous intéressons à ses variables de contexte à leur évolution dans le temps. A titre d'exemple, nous choisissons comme entité un conteneur de fret et comme variables de contexte sa température, l'humidité, la géolocalisation du conteneur. Les données récoltées par la carte d'acquisition permettent de tracer l'évolution du contexte du conteneur pendant le processus sous forme de courbes, à chaque point de mesure. Nous clustérisons aussi les BU dans des régions afin d'émettre des statistiques (nombre de BU déjà livrées, ou en attente de livraison).

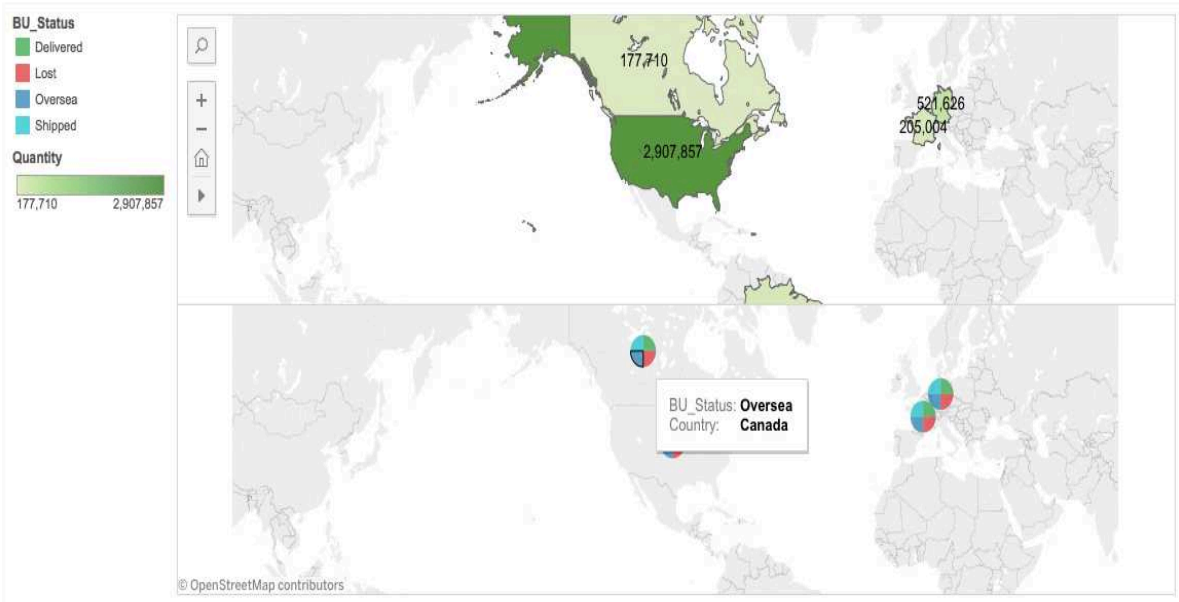


Figure 77. BU groupé par pays et par statut

Dans la figure 77, un BU peut avoir l'un des quatre statuts suivants : *livré (delivered)*, *perdu* (Lost), *en mer (oversea)*, ou *expédié (shipped)*. Ces statuts permettent de grouper les BU et faire des statistiques de base. Pour le Canada nous pouvons par exmple observer qu'un quart de conteneur est livré, et qu'un quart reste en mer. Si nous nous intéressons au changement de contexte dans le temps, la figure 78 présente l'acquisition des données stockées dans la base de données Firebase.

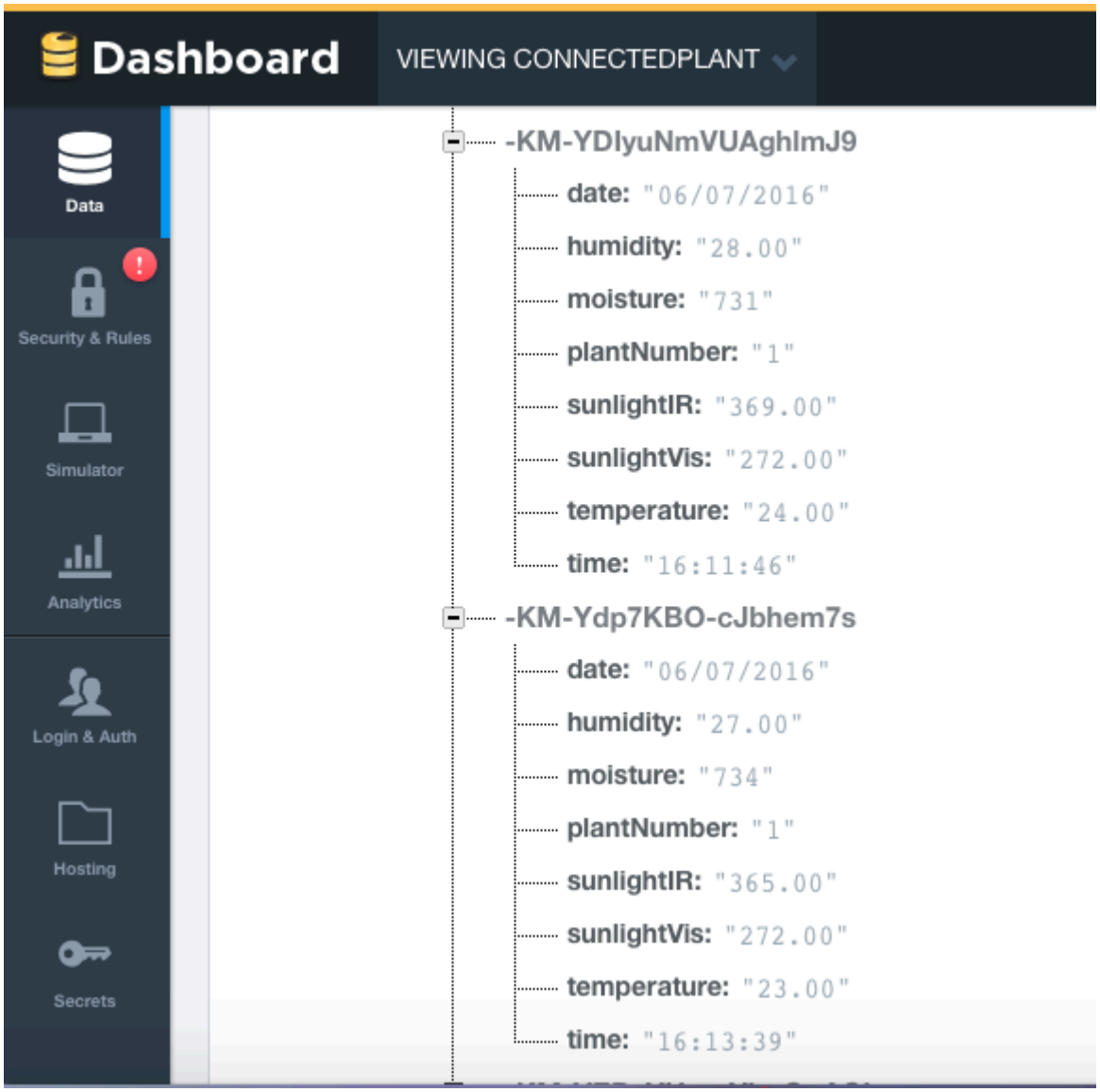


Figure 78. Acquisition des données sur le contexte du BU et stockage dans Firebase

Sur la figure 78, nous faisons une télémétrie des données sur les variables de contexte du conteneur, accompagnée d'un horodatage. Chaque prise de mesure est considérée comme un événement capturé par le bus et qui peut changer le contexte du BU. Nous représentons par la suite ces données sous la forme de courbes associées à chacune des variables de contexte (voir la figure 79).

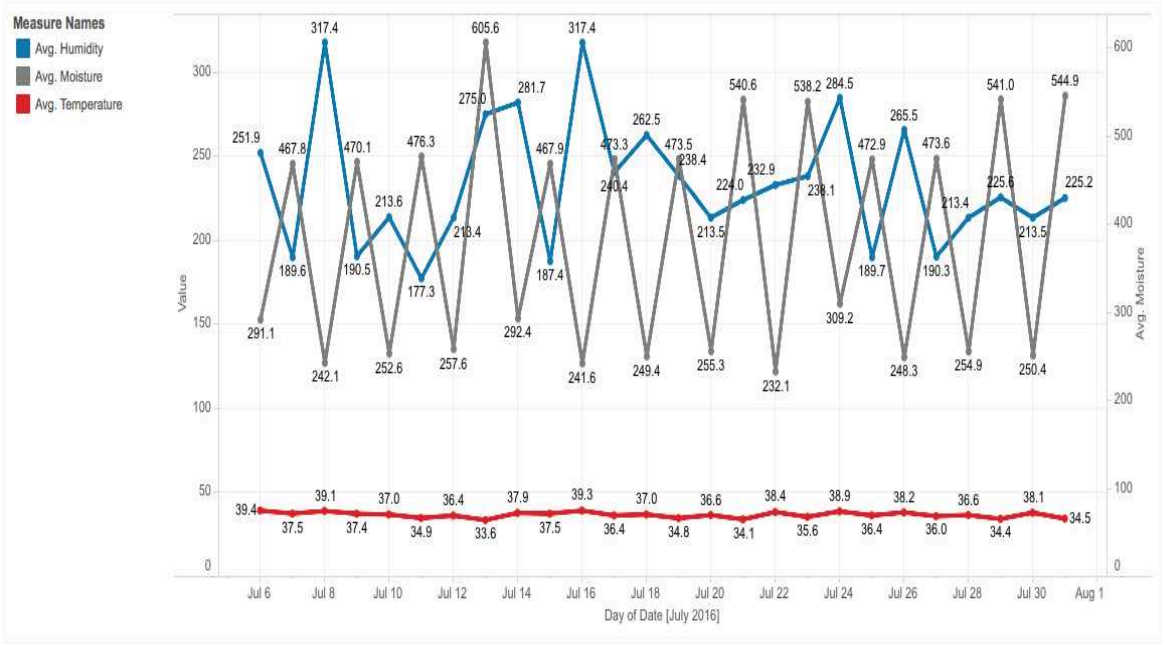


Figure 79. Courbe d'évolution du contexte du BU

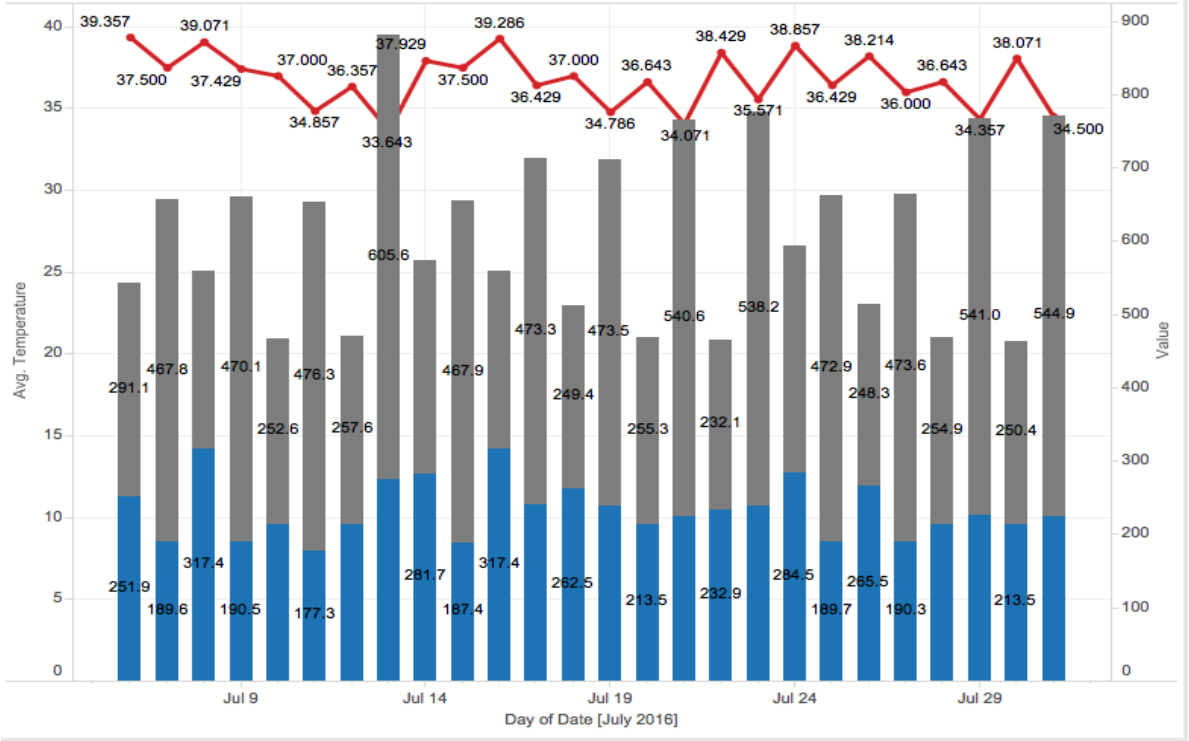


Figure 80. Evolution des variables du contexte du BU sous forme de diagramme à bande

### 5.3.2. Architecture Cloud de Google et son positionnement dans le Cloud Computing

L'architecture Cloud facilite l'exécution d'applications sur des infrastructures virtuelles. Il s'agit d'un ensemble de ressources et de services mis à la disposition du développeur pour atteindre ses objectifs. Ces ressources peuvent être des infrastructures (IaaS), des plateformes (PaaS), des services (SaaS) ou des données (DaaS). Ce paragraphe décrit l'architecture globale de l'offre Cloud de Google (figure 81) et son positionnement par rapport aux autres plateformes et services.

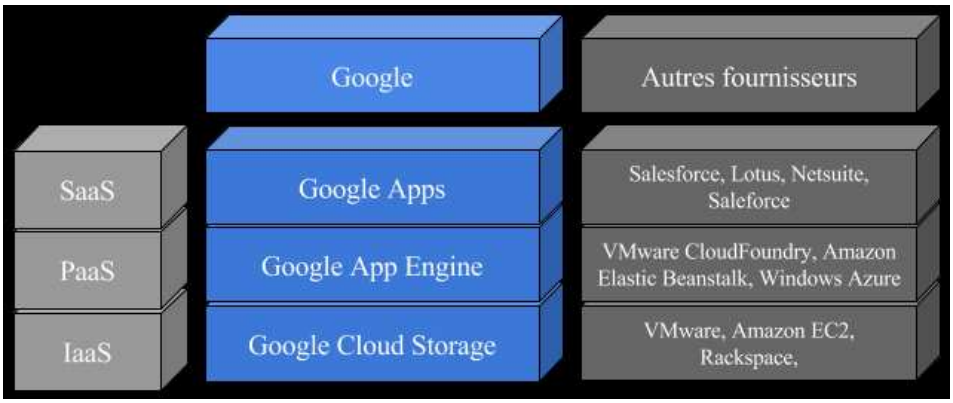


Figure 81. Architecture Cloud de Google et son positionnement dans le Cloud computing

L'offre Cloud de Google peut être subdivisée en cinq groupes :

- Hosting & compute:

Cette composante contient App Engine et Compute Engine.
Google App Engine offre un développement et un déploiement rapides des applications. Elle permet de créer et d'héberger des applications sur la plateforme de Google, de gérer leur administration, des corrections, des sauvegardes, ainsi que l'évolutivité des applications. App Engine sert aussi de créer des machines virtuelles. Google Compute Engine est une machine virtuelle qui offre des capacités de calcul dans un environnement flexible et évolutif dans le Cloud. Elle permet de résoudre des problèmes de traitements et d'analyse de données à grande échelle, ainsi que le réseautage de Google.

- Storage: Cloud Storage, Cloud Bigtable, Cloud Datastore et Cloud SQL:

Google Cloud Bigtable est un service de base de données NoSQL hautement évolutif qui sert à la collecte et le stockage de données de 1TO à plus de 100 PB. Le Cloud Storage est un service Restfull permettant de stocker et d'accéder aux données sur l'infrastructure de Google avec la sécurité et le partage. Le Cloud Datastore quant à lui est une base de données non relationnelle qui gère un ensemble de requêtes (Google Query Language-GQL) et les montées en charge, ayant la possibilité de s'adapter à plus de 1000 utilisateurs, voire

10 millions d'utilisateurs de manière synchrone. Le Cloud SQL est un service web qui permet de créer et configurer des bases de données relationnelles, c'est l'offre SQL de Google en mode Cloud.

- Big Data: Google BigQuery Service

Google BigQuery Service permet l'analyse et le stockage de grand volume de données pouvant atteindre des centaines de téraoctets avec la possibilité de faire des requêtes ad hoc.

- Networking: la gestion des réseaux Google propose trois APIs, le Load Balancing, Interconnect et DNS.

- Services: Cloud Endpoints, Translate API, Prediction API, etc.

Google Cloud Endpoints est un outil qui facilite le développement et le déploiement d'API, de sécuriser et administrer des APIS sur Google Cloud Platform.

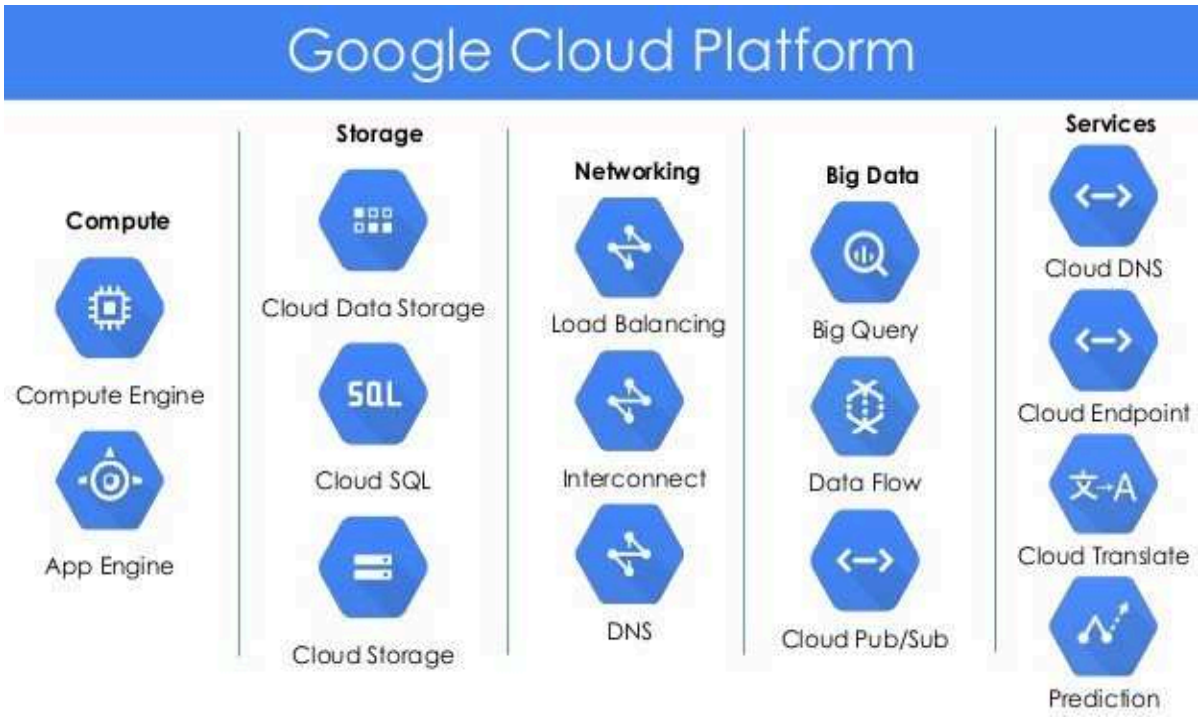


Figure 82. Présentation de la plateforme Cloud de Google

### 5.3.3. Réalisation de la plateforme

L'un des objectifs de ces travaux est la réalisation d'une API REST consommable en mode SaaS pour la publication des données et des événements sur les objets logistiques, ainsi que les informations sur les acteurs qui y interviennent. Dans les paragraphes qui suivent nous expliquons la mise en œuvre de cette API et les critères de mesure de ses performances.

- Relation entre l'architecture générique proposée et l'architecture d'implémentation

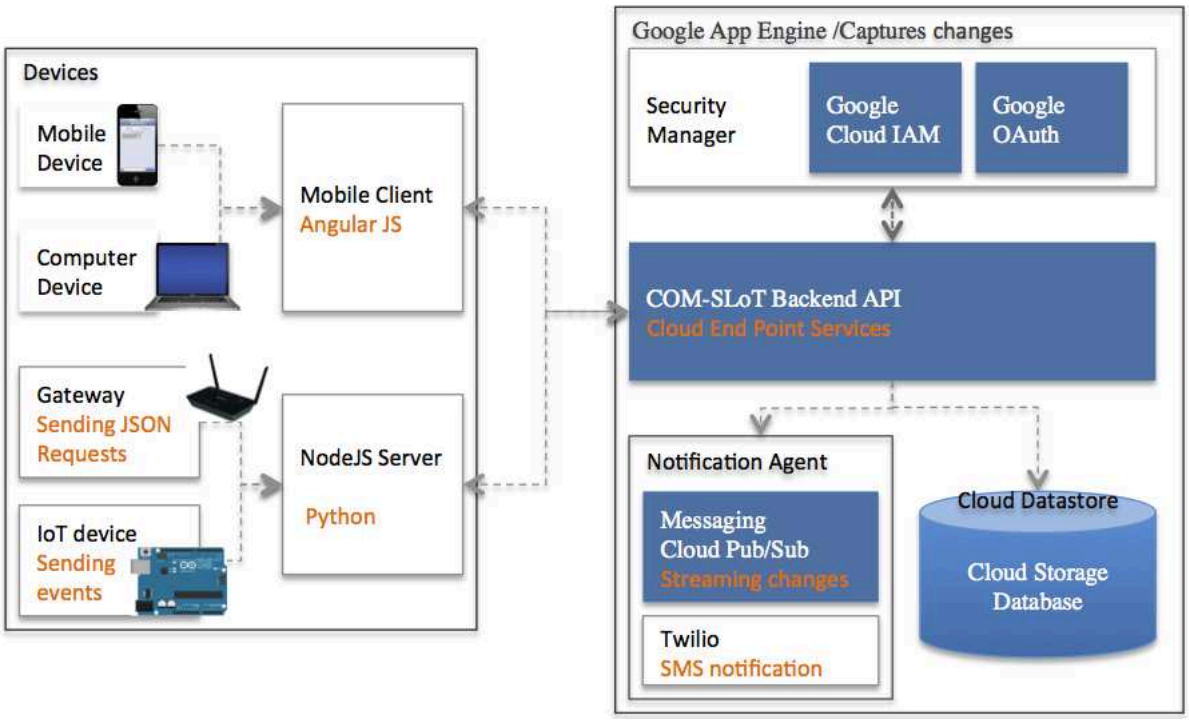


Figure 83. Architecture d'implémentation de la plateforme dans Google Cloud Platform

Le module principal de cette architecture est l'API développé pour servir de backend à la plateforme (COM-SLoT Backend API) (figure 83). Cette API a été développée sous forme de Cloud EndPoint en se basant sur la technologie Google Cloud EndPoint qui permet d'implémenter et déployer rapidement des services web REST. Cette API implémente les fonctionnalités du Data/Event Handler telles qu'elles ont été énoncé dans sa description générique, nous y reviendrons dans les paragraphes qui suivent.

Au niveau du terminal client, nous avons développé une application AngularJS pour les mobiles et Desktop, et une API de récolte et transfert de données en Arduino + Python pour les équipements IoT (IoT devices) qui peuvent être des cartes d'acquisition (Arduino, Raspberry Pi, Galileo, etc. ).

Le module de sécurité est programmé en s'appuyant sur deux technologies principales: Google Cloud IAM pour la gestion dynamique des rôles et des canaux, et

OAuth pour l'automatisation de la stratégie d'authentification. Nous proposons une implémentation du Google Cloud Pub/Sub pour le streaming du flux de données et d'événements sur les Business Units et les workflows. Cette implémentation du Cloud Pub/Sub est complétée par un gestionnaire de notification (Twilio) qui informe en temps réel les utilisateurs par SMS sur tout changement du contexte des objets et du workflow. Ces deux modules implémentent la logique de notre agent de notification. Enfin pour stocker l'ensemble des données sur le flux et ses ressources nous nous appuyons sur Cloud Datastore.

### 5.3.4. Module de publication des événements

Le module de publication des événements (donné dans la figure 84) est un service permettant de réaliser huit opérations ou micro-services. L'ajout d'un évènement survenu sur le flux d'un Item, la recherche des évènements concernant un acteur logistique ainsi que la recherche des événements survenus sur un flux dans une région géographique donnée. Il permet également de produire des états sur les évènements selon les organisations impliquées dans le flux et connectées à la plateforme.

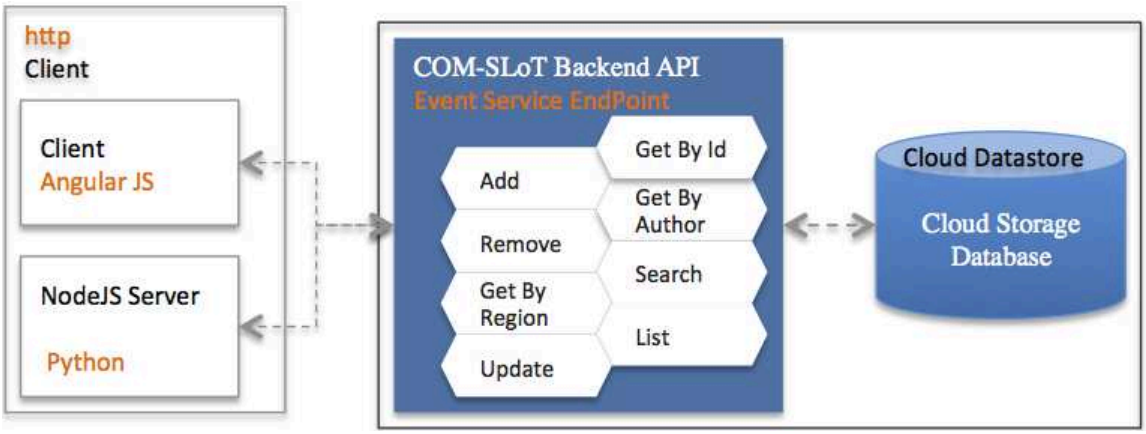


Figure 84. Composant du service de gestion et publication d'événements

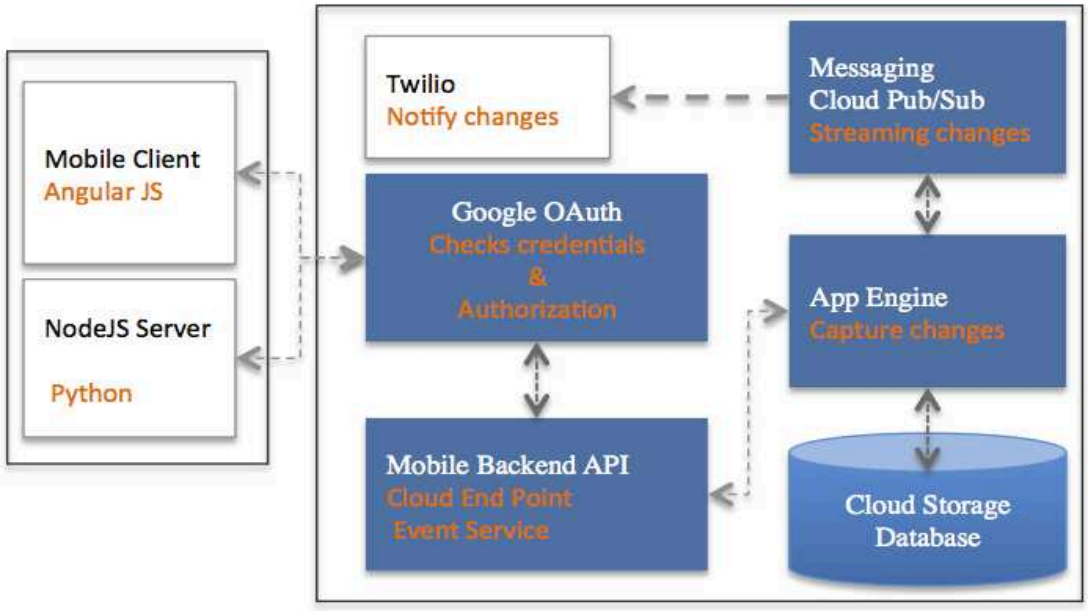


Figure 85. Architecture fonctionnelle du module de publication des événements

Les requêtes de publication, consultation ou de mise à jour des événements arrivent dans le module ***Event Service*** (figure 85). Il s'agit du endpoint mise en place pour gérer tous les événements de la plateforme. Toutes les requêtes sont préalablement vérifiées par le module de gestion de droit d'accès qui utilise le service ***OAuth***. Ce service s'assure que l'utilisateur est bien enregistré dans la plateforme et qu'il possède un niveau suffisant pour accéder à ce flux ou à cet Item. Le service que nous avons déployé vérifie les paramètres de la requête et effectue quelques traitements, enregistre les données de la requête dans le ***Cloud Data*** *Store* et démarre le module ***Cloud Pub/Sub*** pour la diffusion des notifications.

Le module ***Twilio*** est un gestionnaire de notification par SMS facilement intégrable avec Google Cloud Plateform (figure 85).

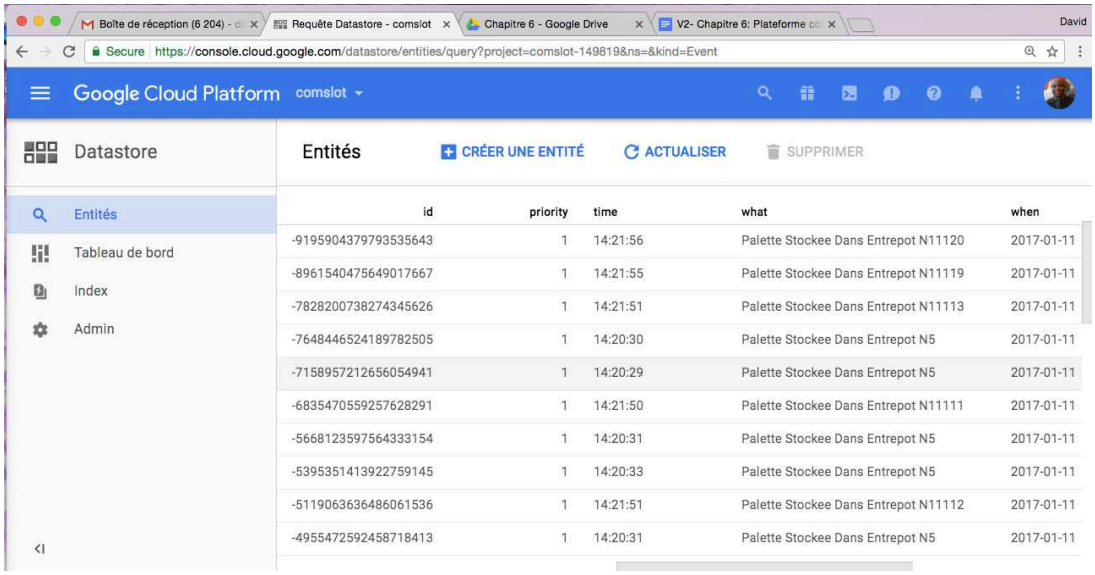


Figure 86. Exemple de stockage des événements dans le Datastore.

Nous pouvons comparer le processus de publication d'événements sur la plateforme au processus d'ingestion de la plateforme ***Google Cloud Platform (GCP)***. L'ingestion est le processus d'importation des informations provenant des périphériques connectés vers GCP. Selon le type d'information, Google fournit un service d'ingestion différent en fonction des données importées : données télémétriques ou données opérationnelles venant des services ou infrastructure IoT.

Nous utilisons ici le processus d'ingestion et plus précisément le module ***Cloud Pub/Sub*** pour la publication d'événements sur le flux logistique. Le Cloud Pub/Sub permet de créer des rubriques (***topics***) ou des canaux (***channels***) sur les flux d'événements publiés, donne la possibilité d'activer ou désactiver les canaux, de s'abonner à un ou plusieurs canaux au même moment pour recevoir des événements publiés sur ce canal. Cette façon de procéder est très intéressante dans la mesure ou un acteur logique peut n'être intéressé qu'à une partie du processus (bStep) et donc ne s'abonner qu'à cette étape du workflow pour recevoir des événements publiés par les autres intervenants durant cette étape. Nous pouvons faire le parallèle avec la zone géographique couverte par le flux, et s'abonner à un canal selon la région géographique qui nous intéresse (par exemple, uniquement les containers qui transitent par l'Ile de France).

La possibilité de connecter le Cloud Sub/Sub à d'autres services Cloud ou d'autres sources de données permet d'étendre ses capacités et gérer des pipelines de données et systèmes de stockage. Enfin les modules Cloud Pub/Sub gèrent de gros volumes de données en provenance des objets logistiques et plus généralement des capteurs ayant des capacités de stockages réduits. En terme de protocole, le Cloud Pub/Sub prend en charge l'API HTTP REST standard et le Framework ***gRPC*** dont le débit est plus élevé pour les applications de télémétrie en IoT au format binaire.

La figure 87 montre le service de publication des événements tel qu'il est implémenté actuellement avec Google Cloud Endpoint. Nous avons fait abstraction des paramètres de méthodes des micro-services pour des besoins de lisibilité.

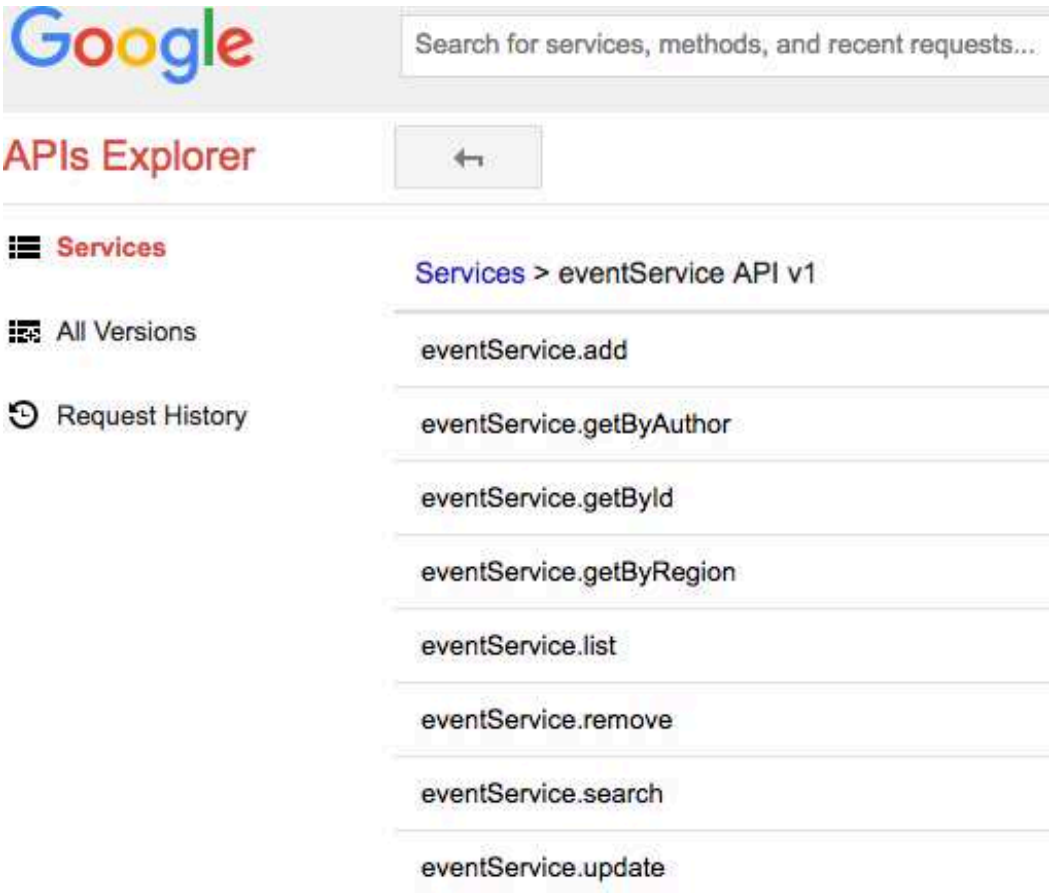


Figure 87. Module de publication des évènements

L'interface de publication et de consultation d'événements que nous avons développée en Angular JS est présentée dans la figure suivante.

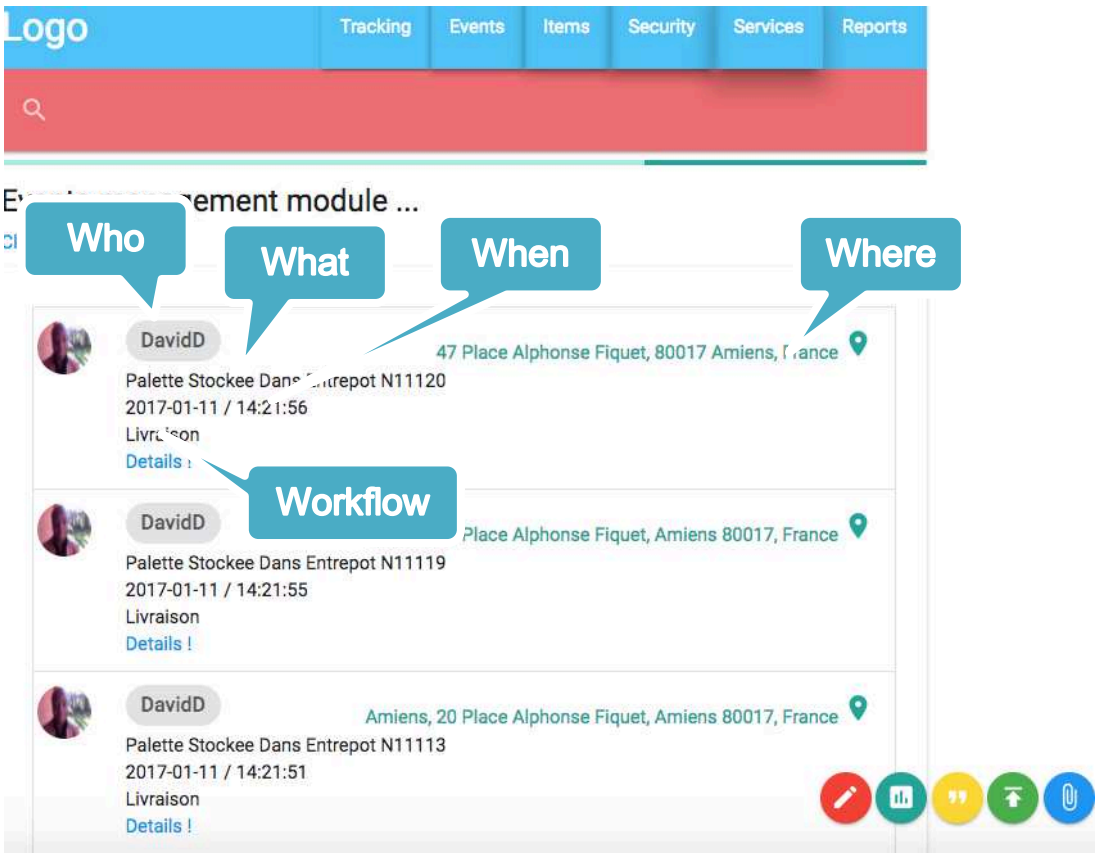


Figure 88. Interface de publication et consultation des évènements

Comme illustré sur la figure 88, nous présentons à l'utilisateur cinq informations clés dans le pilotage du flux : *What*, *Who, Where, When*, Workflow. Cette syntaxe a été commentée dans les chapitres précédents où nous avons présenté le modèle.

Le menu principal présente les différents modules/services de la plateforme. Ce module est dédié à la gestion et la publication des événements. La liste des événements est classée du plus récent au plus ancien. Un clic sur le lieu de l'événement mène sur le flux en question et permet de faire le tracking de celui-ci sur la carte (Map). Un clic sur Détail permet d'avoir des informations complémentaires sur cet événement comme l'étape du processus logistique, l'Item concerné, ainsi que d'autre méta-données du flux. Un menu flottant permet d'accéder directement à la fenêtre d'ajout d'un nouvel événement, aux indicateurs de performances (KPI's), aux commentaires qui ont suivi la publication de cet événement, à télécharger la liste des événements et à joindre un fichier.

Il existe deux modes de visualisation des événements : le mode liste et le mode tracking. Le mode tracking permet de faire une cartographie des événements comme le montre la figure 89.

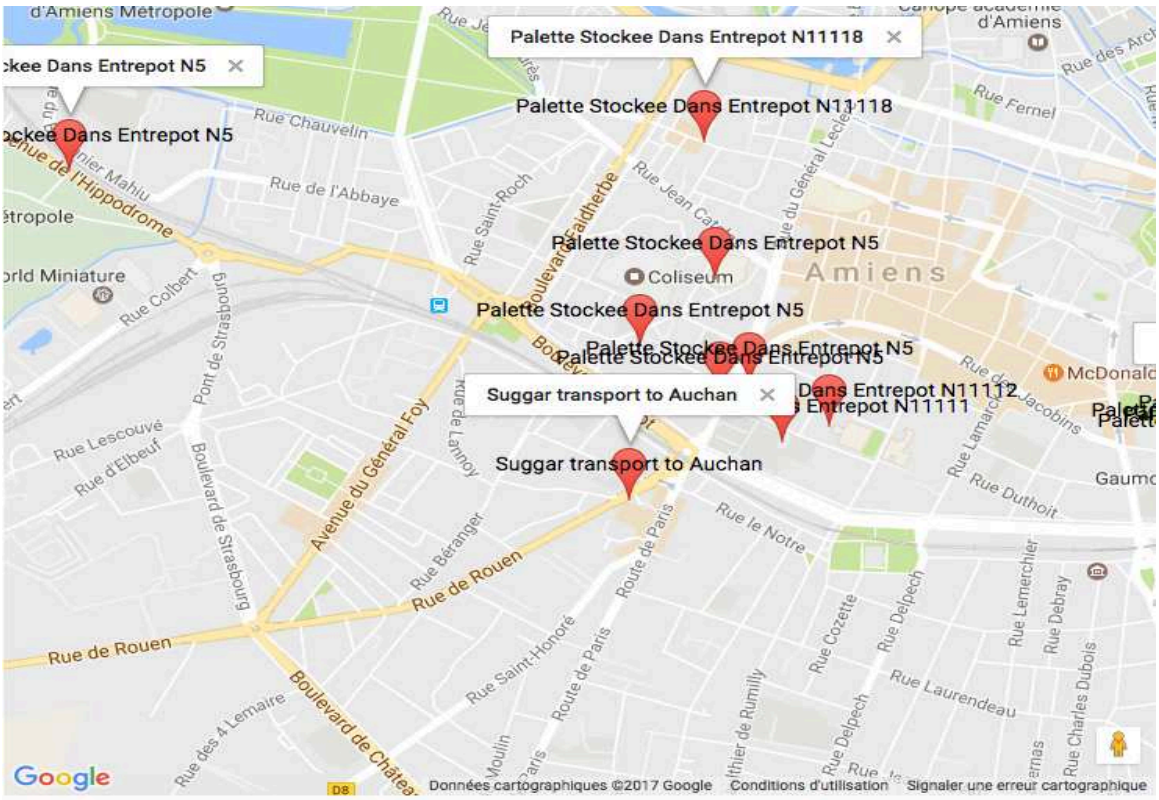


Figure 89. Cartographie des événements du flux

### 5.3.5. Module de gestion des objets logistiques (Items)

Ce service permet la publication des objets logistiques, quelle que soit leur nature. Il dispose de sept micro-services qui permettent d'ajouter un nouvel objet dans la base des objets, de mettre à jour leurs propriétés, de rechercher la liste des objets appartenant à un acteur ou à une région géographique donnée du flux. La figure 90 présente les différents modules du service de gestion des objets logistiques.

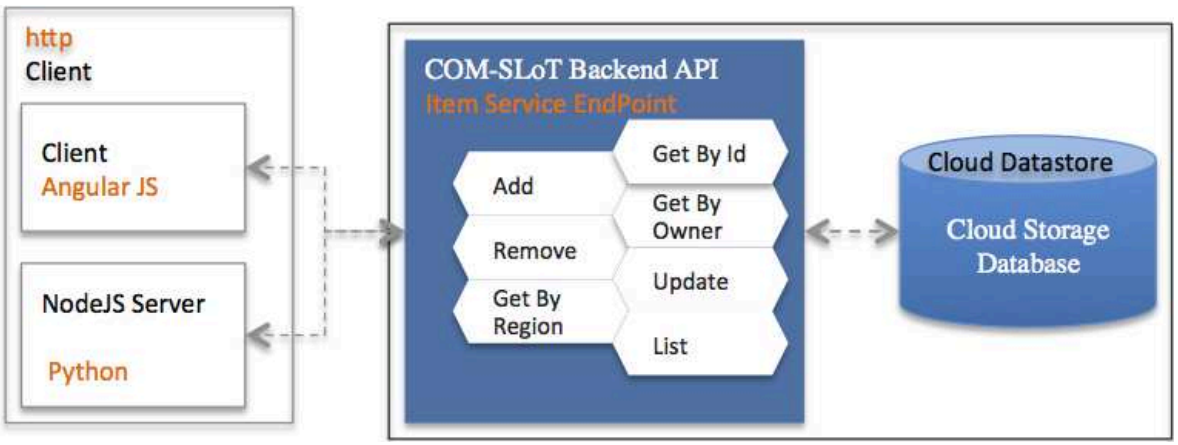


Figure 90. Service de publication des Items

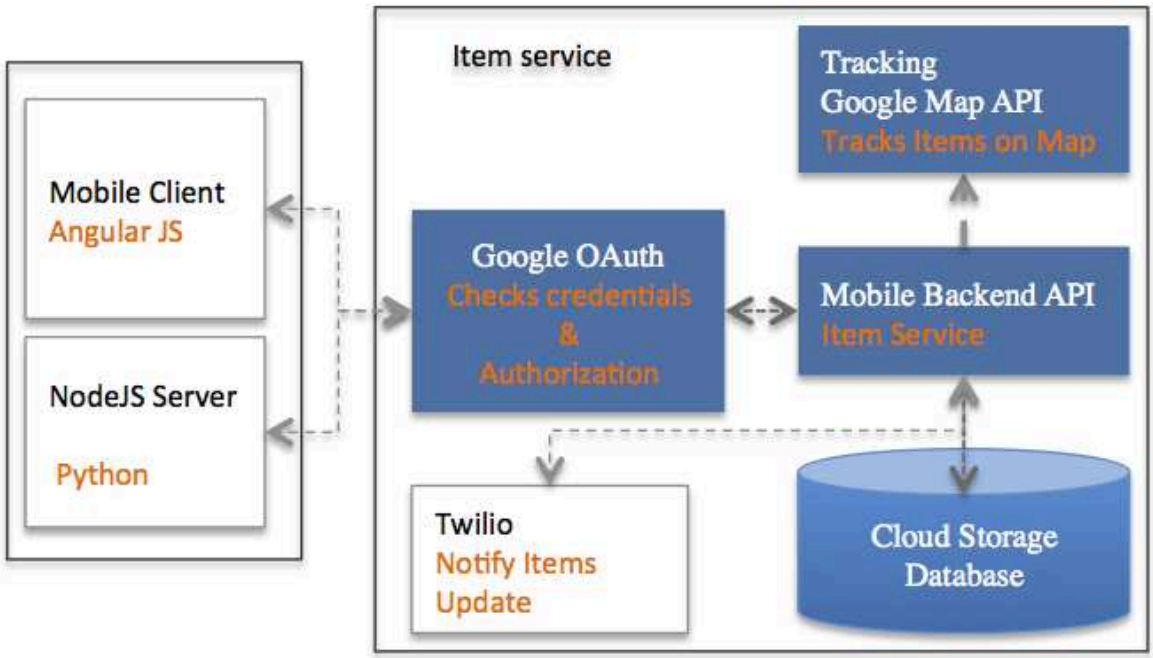


Figure 91. Architecture fonctionnelle du module de publication des objets logistiques

Une notification automatique est envoyée aux utilisateurs connectés à cet Item pour les informer des mises à jour relatives à cet objet. Pour ce faire nous utilisons le service de messagerie Twilio pour l'envoie des SMS de notification.

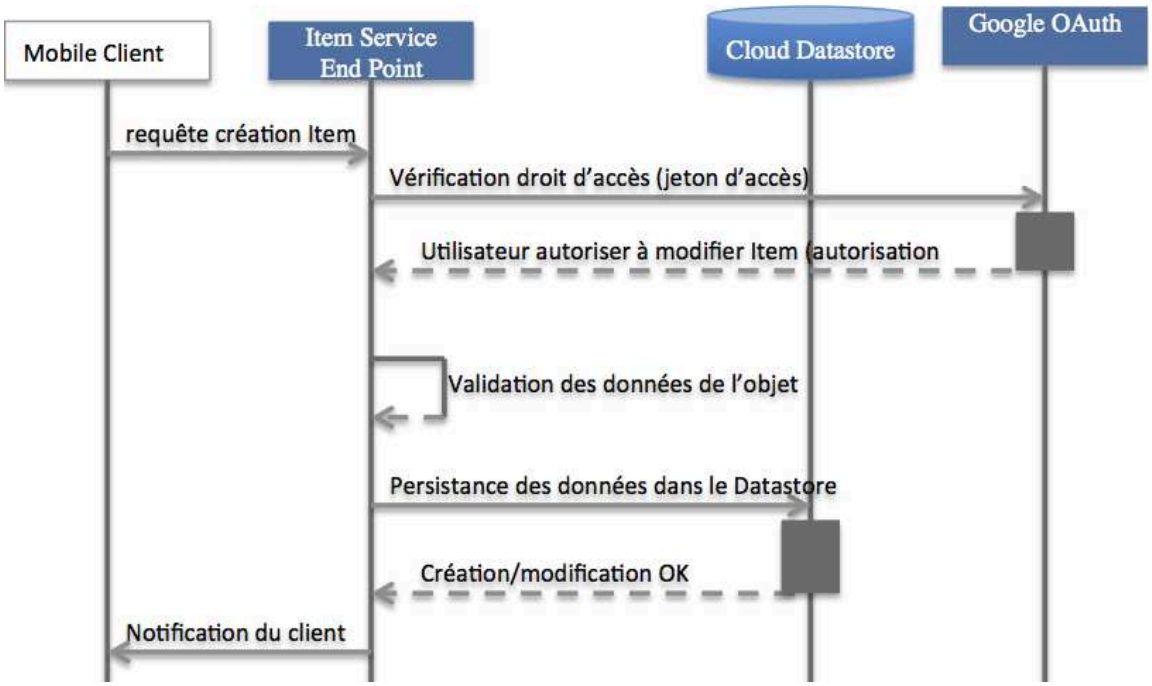


Figure 92. Diagramme de séquence de la publication /mise a jour d'un objet

Le diagramme de séquence de la figure 92 présente la séquence des événements pour une opération de création d'un objet dans la plateforme. Comme mentionné sur le schéma,

l'API communique avec le service d'authentification afin de vérifier les permissions de l'utilisateur ayant demandé la requête en utilisant la clé de sécurité. Ensuite, les données sont validées par l'API avant l'opération d'insertion dans le Datastore. L'utilisateur est enfin notifié de la réussite de l'opération.

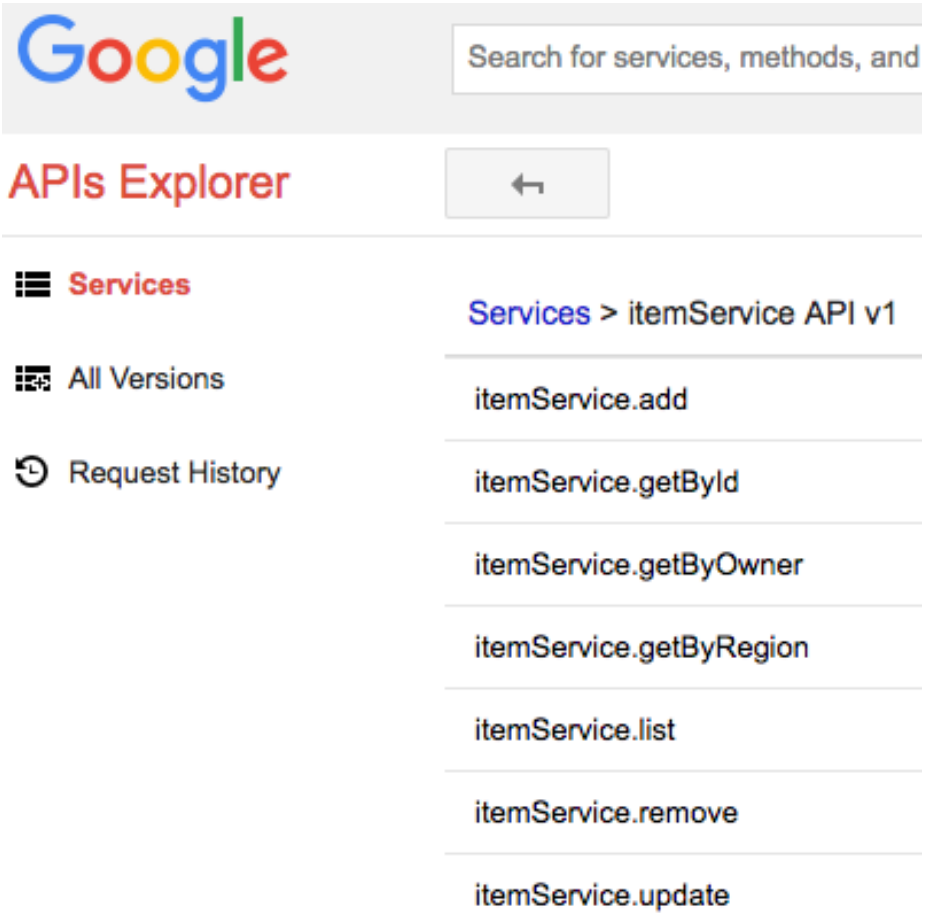


Figure 93. Service de publication des objets logistiques dans le Cloud Endpoint

### 5.3.6. Module de gestion des utilisateurs, des rôles et des droits d'accès

Le service de définition des acteurs logistiques ainsi que les organisations (figure 94) permet de créer des profils d'utilisateurs avec les entreprises associées, de définir les droits d'accès de ces utilisateurs aux différents flux (Workflows) pilotés par l'entreprise. De faire des recherche: rechercher les acteurs logistiques dans la même organisation, par région, afin de faciliter les opérations automatiques de notification, en ciblant exactement les personnes concernées.

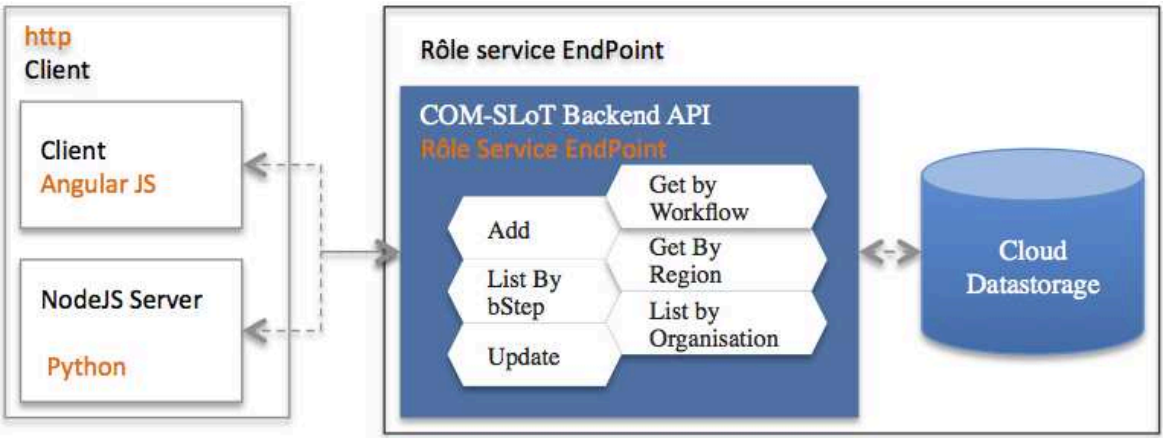


Figure 94. EndPoint du service de gestion des Acteurs et des organisations

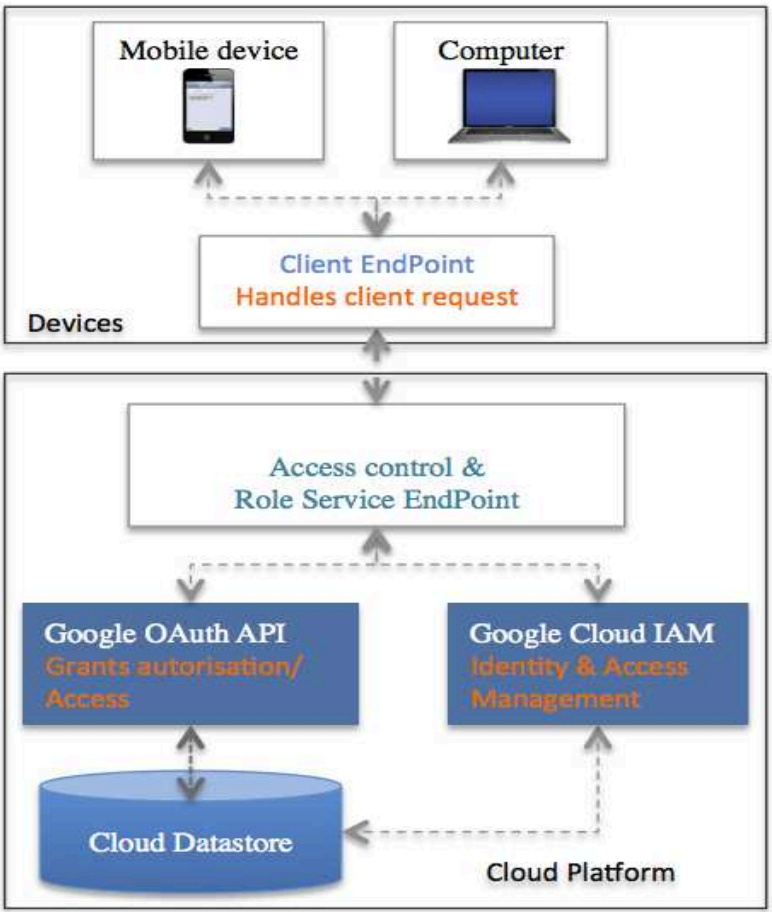


Figure 95. Architecture fonctionnelle du service de contrôle d'accès et de gestion des rôles

Nous utilisons le contrôle d'accès pour filtrer l'accès aux données du flux en fonction du rôle joué par l'acteur logistique. Pour vérifier les informations de logging des utilisateurs de la plateforme. Les deux modules ***OAuth*** et ***Cloud IAM*** sont utilisés pour atteindre ces objectifs. Le service Google ***Cloud Identity & ACCESS Management (Cloud IAM)*** permet de donner des permissions à un niveau de granularité plus fin (niveau ressource) plutôt que de se limiter à des niveaux projet (***Project***) et groupes d'utilisateurs

(***Work group***). L'idée d'utiliser ce service pour administrer les accès dans la plateforme est de pouvoir attribuer dynamiquement des rôles et des droits d'accès à un utilisateur ou un groupe, ou une organisation selon le flux. Nous pouvons ainsi donner le rôle de souscripteur à un utilisateur pour un canal bien défini. Rappelons que les canaux sont créés sur des Workflows, les stages ou les Items. De même que précédemment, nous utilisons le service de stockage Cloud Datastore pour stocker les informations sur les utilisateurs de la plateforme, ainsi que sur les rôles qu'ils occupent dans les différents workflows.

Le diagramme de la figure 96 décrit le processus d'authentification, de sécurité et de contrôle d'accès via OAuth.

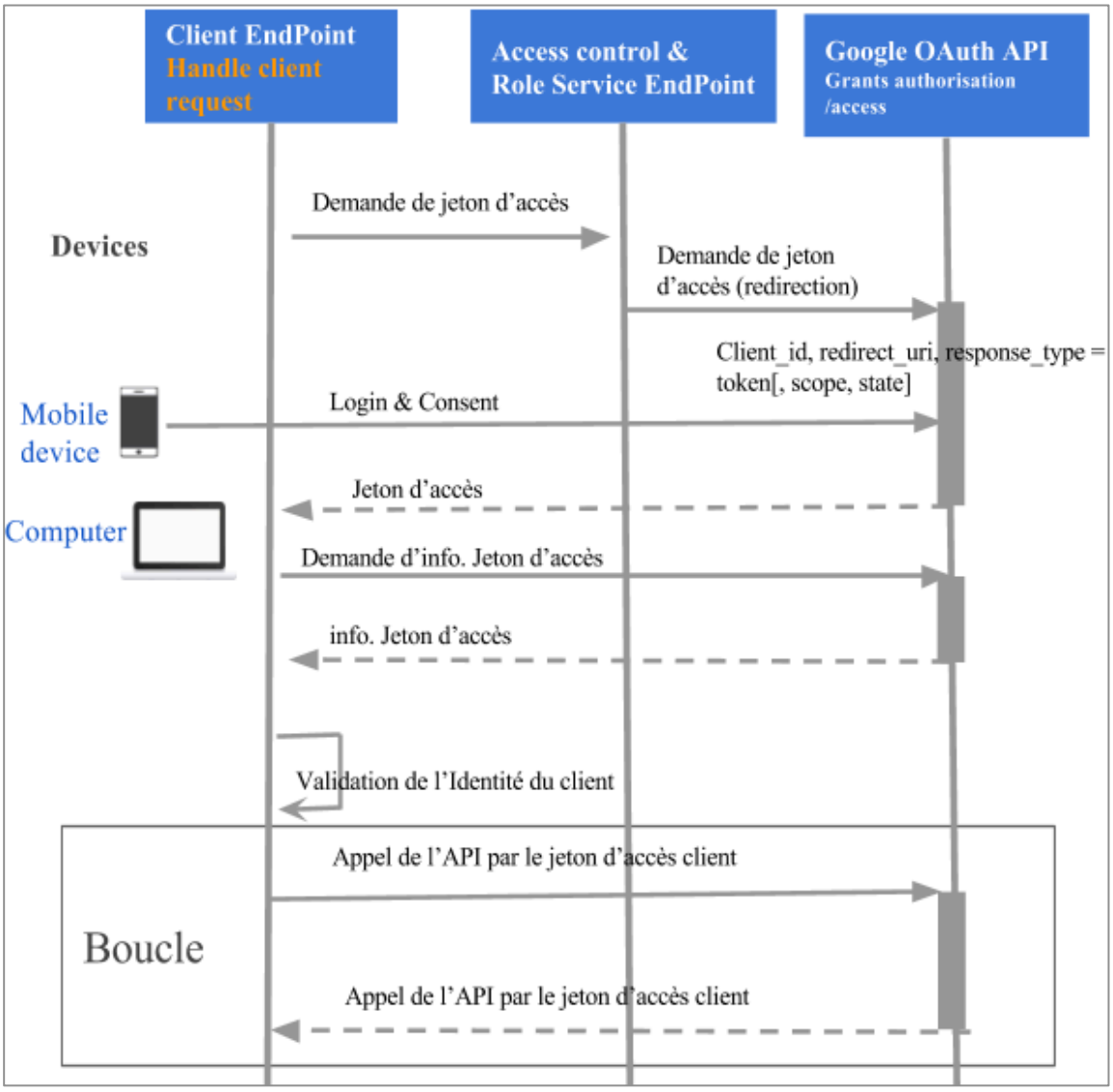


Figure 96. Séquence de fonctionnement du contrôle d'accès via Google OAuth (Cas Implicite Grant Flow)

La figure 97 présente le service de gestion des rôles et ses sous-services tel qu'il a été implémenté dans Google Cloud Endpoint.

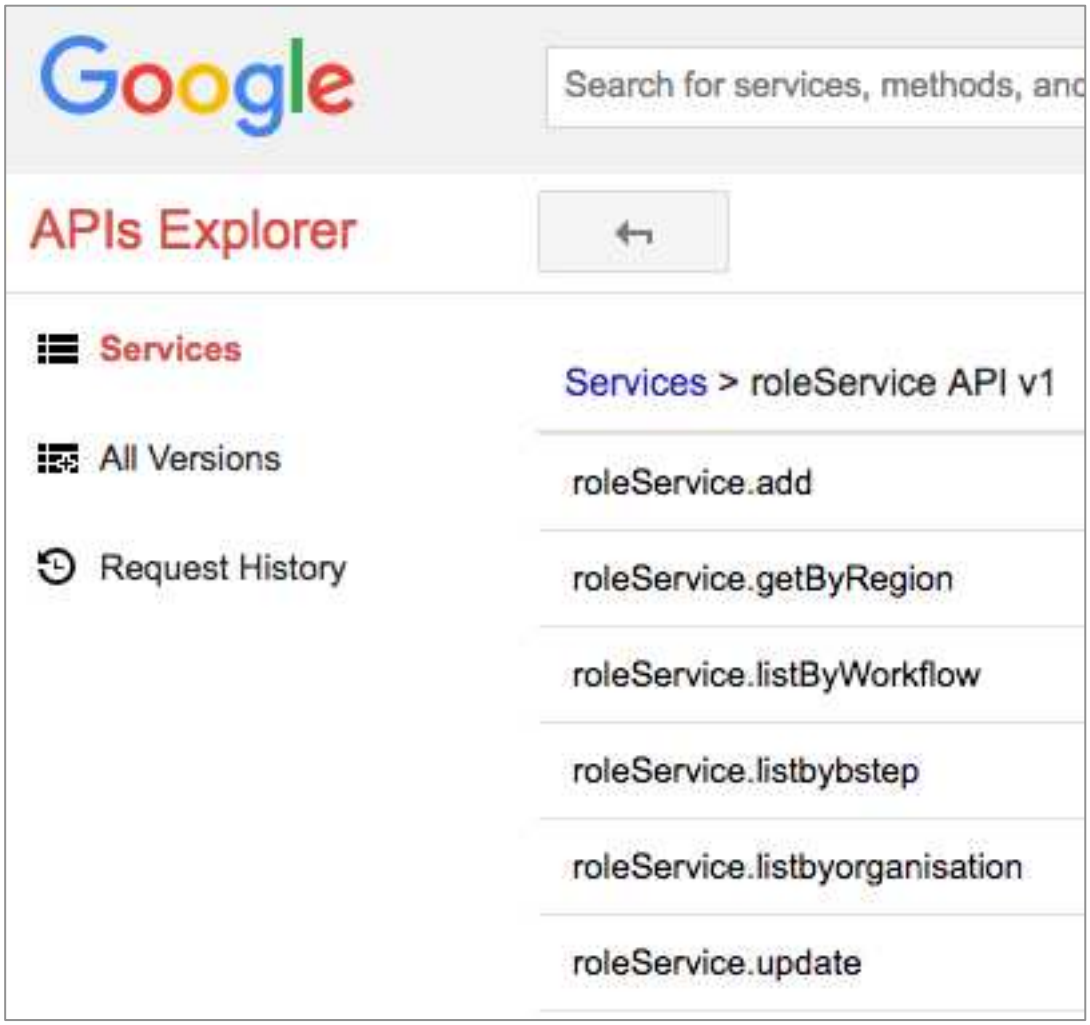


Figure 97. Service de gestion des Rôles et des droits d'accès

### 5.3.7. Module de définition des workflows

Le service de définition des workflows permet de créer le schéma descriptif d'un workflow. Le schéma descriptif comprend les caractéristiques du workflow ainsi que les différentes activités (*stages*) et opérations qui seront executé du début jusqu'à son accomplissement. Il est question ici d'une planification du processus en amont au niveau stratégique avant le démarage et le suivi des activités. Le service permet aussi de gérer les différentes activités du workflow, leurs mises à jour aisni que l'affection des ressources.

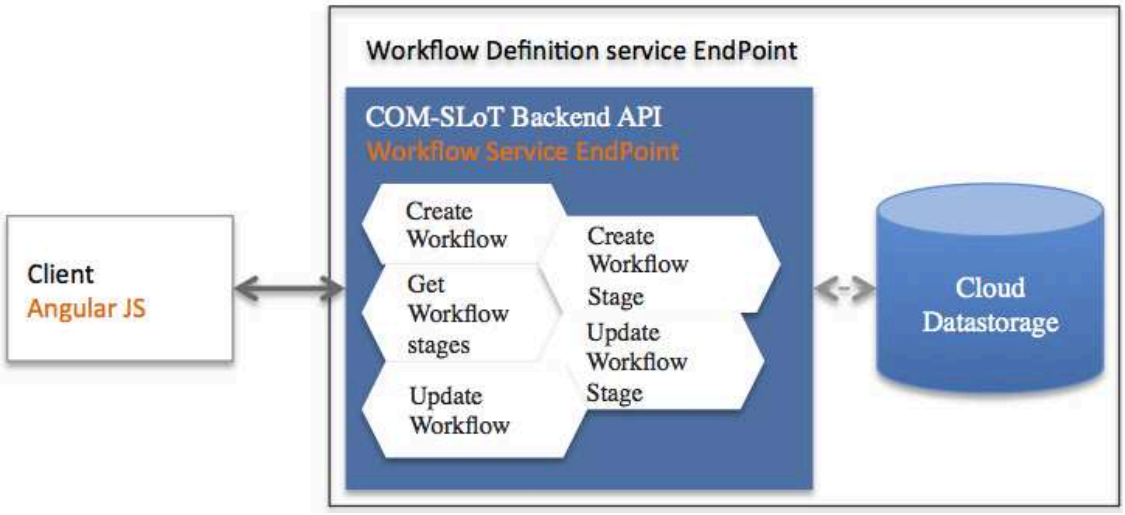


Figure 98. End Point du service de publication d'événements

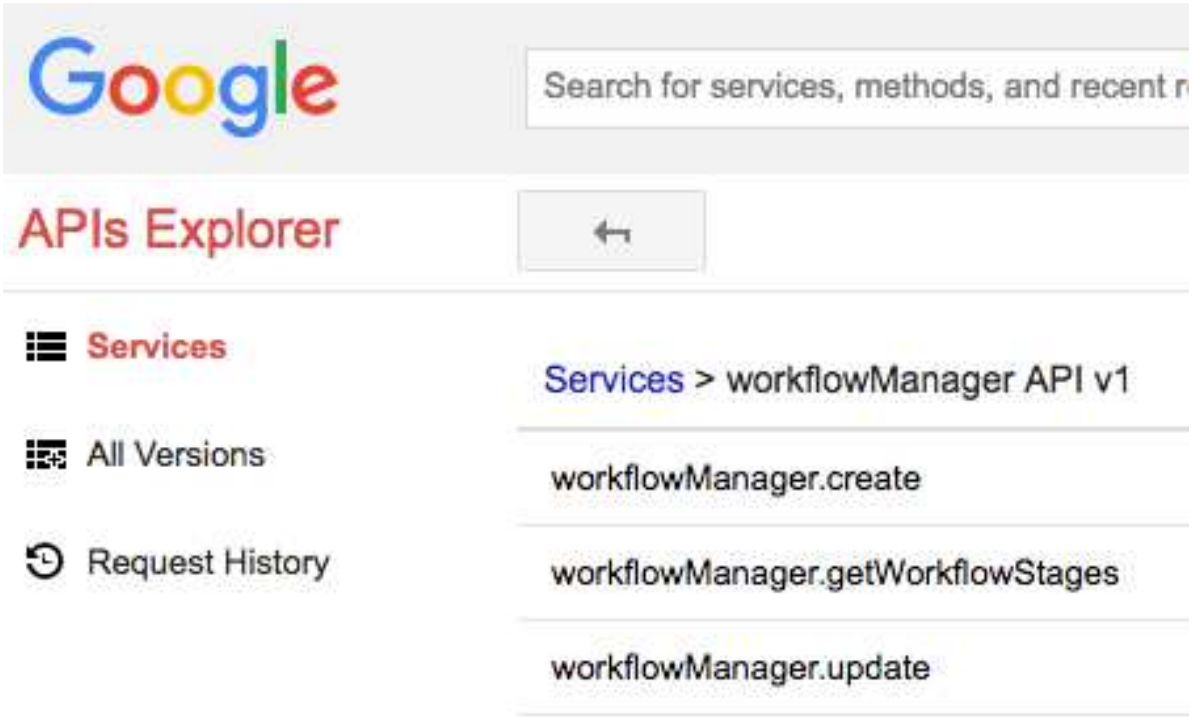


Figure 99. Module de définition des workflows

## 5.4. Mesure des performances de la plateforme

**Cas d'usage : montée en charge au niveau de l'utilisation des services de la plateforme**

Ce cas d'usage consiste à tester la robustesse de la plateforme collaborative à absorber le volume de requêtes et de données en provenances des objets logistiques, des services

propriétaires, ou des véhicules de transport connectés. Il s'agit par exemple des capteurs GPS de suivi de flottes, du matériel mobiles qui remontent des données en temps réel sur leur état. Pour ce cas de figure, nous considérons 16 entreprises (E1, E2, E3, ..., E15, E16) logistiques. Nous supposons que les entreprises subissent une monté en charge de l'activité dûe à une demande très forte pendant un laps de temps très court c'est à dire une forte demande de disponibilité de la plateforme dans cet intervalle de temps (inferieur à 5 minutes). Enfin, nous considérons que toutes les 16 entreprises utilisent la plateforme simultanément.

Pour simuler ce cas de figure, nous considérons 16 processus qui représentent les différentes entreprises. Chaque processus envoie 1000 requêtes à la plateforme. Ces requêtes représentent les différentes sollicitations des utilisateurs de l'entreprise pendant ce laps de temps. De plus nous supposons que les demandes des entreprises sont indépendantes les unes des autres et donc s'exécutent en parallèle. Les 16 processus sont donc exécutés en parallèle sur des machines différentes, la limite des 16 machines est fixée par l'environnement de test. Ce nombre est limité au nombre machines disponibles dans la salle des tests.

La limite des 1000 tests par processus est due à la plateforme Cloud de Google[6]. En effet, en version de développement à notre disposition, une application a droit d'exécuter au maximum 20 000 requêtes d'écriture par jour, 50 000 requêtes de lecture d'entités par jour, 20 000 requêtes de suppression d'entités par jour, 1 GB de stockage de données par jour [7].

- Caractéristiques de l'environnement des tests

Les tests de montée en charge ont été effectués sur un environnement Windows. Les machines sont équipées de processeurs Intel. Le tableau ci-dessous décrit les caractéristiques système des machines ayant servi aux tests. Les machines sont identiques et possèdent les mêmes caractéristiques.

| CARACTERISTIQUE | VALEUR |
|---|---|
| Système d'exploitation | Microsoft Windows 8.1 Professionnel |
| Type du système: | x64-based PC |
| Processeur(s): | 1 processeur(s) Install(s).<br>[01]: Intel64 Family 6 Model 60 Stepping 3 GenuineIntel ~3600 MHz |
| Mémoire physique totale: | 16.326 Mo |

[6] *https://Cloud.google.com/datastore/pricing*

- Résultats des tests et interprétations

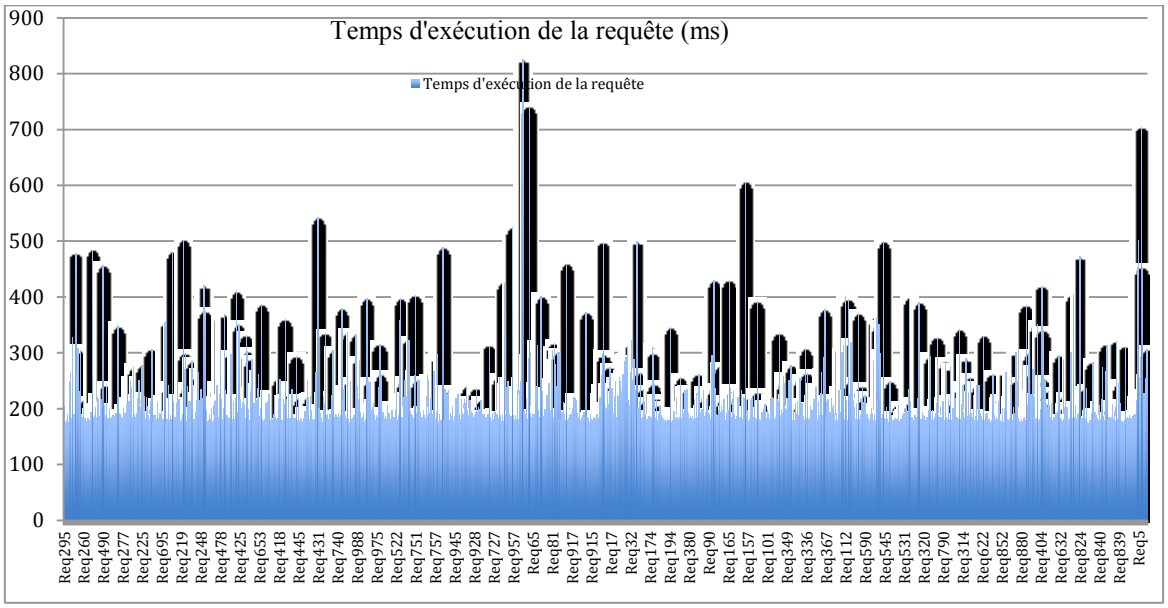


Figure 100. Latence du serveur pour chaque requête cliente (ms)

Nous avons mesuré et représenté sur ce graphique le temps de réponse du serveur pour chacune des 1000 requêtes. La figure 100 illustre les résultats obtenus pour la première entreprise. Sur ce graphique nous pouvons observer par exemple que le temps minimal d'exécution d'une demande cliente pour cette entreprise est de 173 ms. Le temps maximal d'exécution pour ce batch est de 825 ms, soit moins d'une seconde. La moyenne des temps de réponse pour les 1000 requêtes s'établie à 218,194 ms.

Intéressons nous au temps de réponse globale des 16000 requêtes. Nous avons représenté les courbes des minimums et des maximums obtenus dans chaque entreprise. Pour faciliter la comparaison, nous avons ajouté la valeur de la moyenne des temps de réponse pour les 16000 requêtes. Le graphe suivant représente les courbes obtenues.

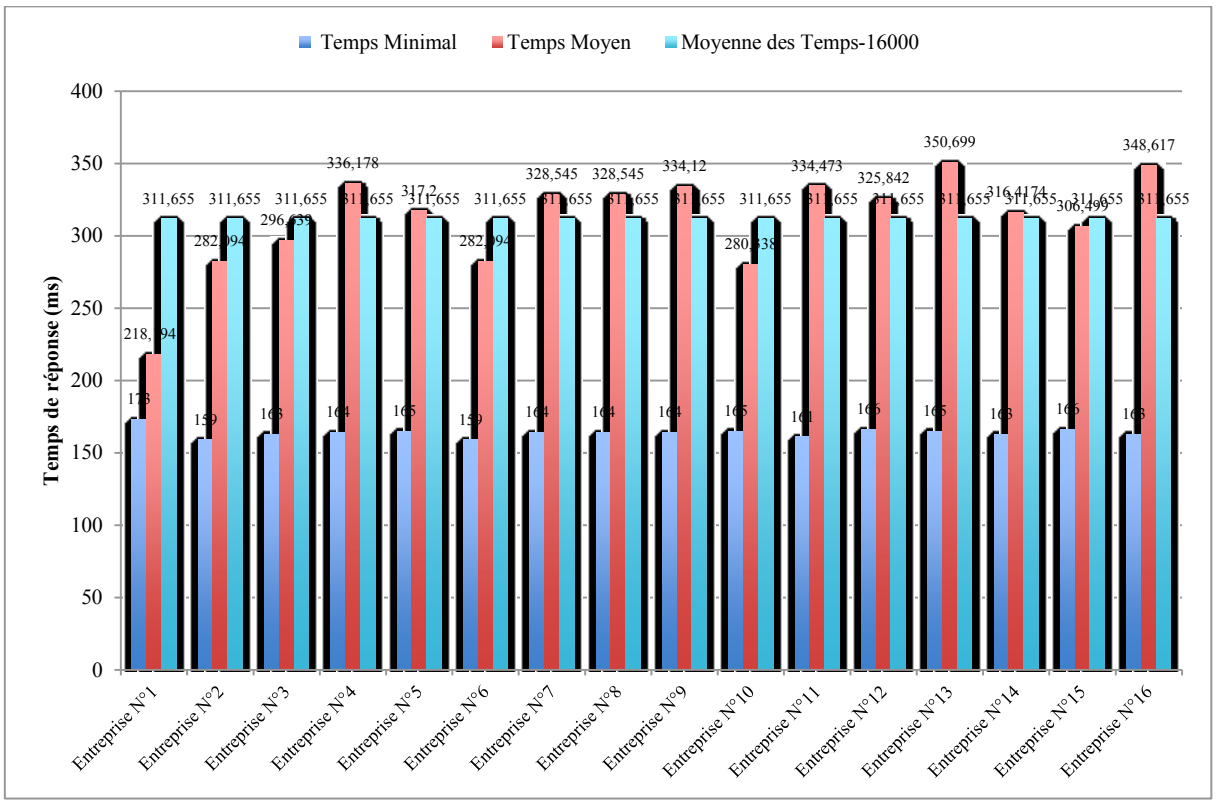


Figure 101. Comparaison des temps de réponse à la moyenne globale des 16000 requêtes

On observe sur ce graphique que la moyenne des temps de réponse par entreprise est inférieure à une démi seconde. Le serveur de la plateforme met en moyenne 311 ms (soit près d'un tiers de seconde) pour répondre à chacune des 16000 demandes. Dans une entreprise sélectionnée au hasard, le temps de réponse moyen sera inférieure à 350 ms, soit presque un tiers de seconde.

**Interprétation des résultats dans le contexte de la collaboration des acteurs logistique**

Le temps moyen d'exécution des 16000 requêtes s'établie autour de 312 ms. Les 16000 requêtes se sont exécutées en un temps total inférieur à 360 s, c'est à dire dans un intervalle de 6 minutes. Cela veut dire que 160000 acteurs logistiques qui envoient simultanément une demande aux services de la plateforme dans un intervalle inférieure à 6 min verront leurs informations mises à jour en moins de 1/3 de seconde en moyenne.
Dans le pire des cas, l'information publiée sera disponible après 20650 ms soit 3 minutes, scenario qui arrive après chaque 1000 demandes dans un batch de 160000.

Prenons le cas concret de la flotte des wagons de la SNCF. Sur le réseau national circulent à peu prés 15000 locomotives par jour (voyageurs et marchandises) [117]. Un train peut transporté au maximum 33 wagons courant type plat (R20) [116].
Si nous interprétons la monté en charge dans cette circonstance, nous pouvons dire que la simulation correspond à la situation où toutes les locomotives du parc national SNCF émettent des événements simultanément. Si on considère le wagon comme une unité logistique connectée, cela voudrait dire que 16000/33 = 484 locomotives chargées au maximum de leur capacité pourront emmètre des évènements simultanément à la

plateforme. Ce cas de figure est peu probable de se produire dans le transport de marchandises.

Ainsi nous pouvons dire que les performances de la plateforme en termes de d'absorption et traitement du volume de données et des messages sont globalement satisfaisantes, comparativement aux autres plateformes dédiées aux objets connectés telles que Sigfox ou LoRA. En effet, Sigfox est limité à 140 messages par jour et la taille des messages ne dépasse pas 12 octets. Les tests effectués montrent que notre plateforme a la capacité d'absorber 1000 requêtes en moins de 6 min, soit une requête toute les 3s. Ce qui est largement au dessus des limites imposées par Sigfox et LoRA aux entreprises qui utilisent leur plateforme de stockage.

Mais il faut noter que nous ne prenons pas en compte les facteurs liés à l'infrastructure réseau et qui peuvent influencer négativement la qualité de service de la plateforme et réduire drastiquement le temps de réponse. Ce cas de figure est courant pour les objets connectés mobiles lorsqu'ils traversent des zones où il n'y a pas de couverture réseau, ou le cas des clients ayant une bande passante très faible, ce qui n'est pas le cas pour l'environnement où les tests ont été effectués.

## 5.5. Conclusion

Le but de ce chapitre est d'utiliser les technologies de l'Internet des Objets et du Cloud computing associés aux *GNSS (GPRS / GPS)* pour proposer une plateforme collaborative dont l'architecture est orientée service et événement, en mode SaaS. De proposer l'intégration du géo-positionnement en temps réel via GPS à de nouvelle architecture Cloud pour le tracking des objets logistiques.

Dans un premier temps, nous avons présenté l'architecture adoptée et expliqué les détails de fonctionnement et le rôle de chaque composant en intégrant les technologies de l'IoT et le Cloud computing. L'accent a été mis sur le partage des données à des fins d'interopérabilité et de collaboration entre les acteurs impliqués dans le flux logistique. En outre, une approche centrée artefact a été utilisée pour représenter le comportement de chaque composant du système et modéliser son interaction avec d'autres composants.

Par la suite nous avons réalisé l'implémentation des modules de l'architecture proposée en nous appuyant sur les APIs de l'infrastructure Cloud de Google, notamment le Cloud Datastore pour le stockage, le Cloud IAM pour la gestion des rôles parmi tant d'autres. Enfin les tests réalisés sur la plateforme ont prouvé le bon fonctionnement des différents services de l'API ainsi que leur robustesse en terme d'absorption du nombre de messages en provenance des objets et acteurs logistiques. D'une manière générale nous pouvons dire que l'architecture proposée et les choix d'implémentation sont satisfaisants au regard des plateformes existantes et du contexte d'utilisation qui est celui du partage d'information en vue de collaboration et d'interopérabilité.

# Conclusion générale

Rappelons tout d'abord que ces travaux de recherche ont été réalisés dans le cadre d'un projet régional, appellé COM-SLoT, et situé dans l'axe des outils numériques pour la performance industrielle. Aussi, le contexte scientifique, technologique et métier dans lequel ces recherches se sont effectuées est celui de la *réorganisation des réseaux logistiques* ces dernières années afin d'améliorer la collaboration entre les acteurs de la *Supply Chain* et la réduction des coûts. Ceci à travers le *recours à de nouvelles architectures logicielles et de nouvelles technologies* pour la collecte, le stockage et l'analyse des flux d'informations comme outil de maîtrise de flux, levier de différentiation et de concurrence. Egalement par le *tracking et la traçabilité temps réel des entités et des objets logistiques* via des tags GPS, EPC et RFID. Il s'agit en outre de la *recherche d'une efficacité au niveau stratégique, tactique et opérationnelle* pour une meilleure optimisation des flux logistiques au niveau de la planification des activités et des ressources, de l'exécution et du suivi en temps réel des opérations. Enfin, il est question d'une meilleure *gestion des risques et des prévisions* afin de mieux faire face aux imprévus de toutes sortes, allant de la catastrophe naturelle à une simple panne de machine ou l'indisponibilité d'un opérateur ou d'une ressource.

C'est dans ce contexte que nous nous sommes intéressé à la problématique de la *Modélisation des flux logistiques avec le développement d'une plateforme d'interopérabilité des objets logistiques*. Rappelons que derrière cette problématique se cache des défis scientifiques et technologiques multiples parmi lesquelles :

la *proposition d'un modèle générique de flux logistique centré sur l'entité logistique* en tant que entité autonome et communicante ;

la transformation des *échanges de données Peer to Peer en partage de données à l'ensemble des acteurs* intervenant sur le flux ;

la *gestion de l'hétérogénéité*, du *grand volume de données* généré et *des droits d'accès* et *faciliter l'interopérabilité des flux* ;

l'*intégration des technologies de l'Internet des objets et du Cloud Computing* pour améliorer l'identification , la collecte des données, le stockage, le tracking et la traçabilité des unités logistiques et des marchandises en temps réel ;

la *mise en œuvre d'une plateforme collaborative accessible en mode SaaS* en se basant sur les architectures SOA et EDA.

La démarche scientifique adoptée est subdivisée en cinq étapes, *l'état de l'art* pour le choix des technologies et des outils de modélisation appropriés. Ensuite la *modélisation de la statique du flux logistique* ainsi que le réseau des flux en appliquant la démarche de l'ingénierie dirigée par les modèles, la *modélisation de la dynamique du flux logistique* par la spécification formelle LOTOS, *l'extraction des connaissance dans le réseaux de flux.* En outre, il est question d'assurer la mise en place d'une *architecture de la plateforme collaborative* et sa réalisation par les technologies de l'Internet des Objets et du Cloud Computing pour enfin terminer par les *tests fonctionnelles et de performance* de la plateforme.

Notre première contribution en ce qui concerne l'*état de l'art* a été de dresser une revue de littérature sur les développements récents des technologies, des architectures et des standards de l'Internet des Objets et sur leur utilisation dans la gestion des flux logistiques. Ceci avec un accent particulier sur les technologies d'identification, de collecte, de transmission sans fil et de traitement de données issues de capteurs et des puces RFID pour une logistique communicante, intégrative, flexible et collaborative [123, 124].

Nous avons subdivisé la modélisation en deux axes d'investigation, ce qui nous a permis d'apporter une modeste contribution sur la *modélisation de la statique du flux* de marchandises. Sur ce point nous avons proposé une *extension de la spécification UML par un profil UML* du niveau M2 des méta-modèles MOF, qui nous a permis de mettre en place un *DSML (Domain Specific Modeling Language).* Ce DSML tient lieu de méta modèle pour la représentation générique des flux logistiques, des entités et des ressources matérielles, humaines et logicielles impliqués dans le flux. Restant dans cette logique de la statique du flux, nous avons *proposé un graphe orienté et labélisé (Directed-Labelled-Graph) pour la représentation des relations* d'interconnexions et d'interdépendances entre les différentes entités logistiques d'une part et entre les entités et les processus d'autre part à *l'instar des réseaux sociaux*. Ce graphe permet aussi de représenter les relations entre un flux et les événements qui perturbent et changent son contexte auquel dépend fortement l'évolution de son cycle de vie.

Après avoir modélisé la statique du flux, le deuxième axe consiste en la *formalisation de la dynamique du workflow en tant que système à événements discrets*. Sur ce point nous avons proposé un modèle dynamique du flux basé sur la norme ISO LOTOS (Language Of Temporal Ordering Specification) afin d'analyser les propriétés quantitatives et qualitatives du workflow en tant que processus communicant (CSP-Communicating Sequential Processes). Il s'agit de la détermination des points de synchronisation, des *deadlocks*, les *lifelocks* et les points de *starvation*.

Avec l'intégration du flux dans l'Internet des Objets et les plateformes du Cloud Computing, l'extraction de connaissances avec les outils du web sémantique prend une place prépondérante. Ainsi, nous avons proposé un *modèle d'extraction de connaissances dans un réseau de flux logistique connecté*, une ontologie web. Cette ontologie est enrichie par des concepts provenant de trois standards (XML, BPML et OWL le plus utilisé pour la spécification des ontologies web). Cette modélisation ontologique a pris en compte l'aspect autonome et connecté du flux pour se différencier des autres ontologies existantes dans le domaine des flux logistiques. Cette ontologie a servi de format d'échange, de représentation et d'extraction de connaissances dans un réseau de flux d'objets logistiques connectés.

Sur le dernier point souligné dans la problématique, notre contribution a été la *proposition d'une architecture orientée SOA et EDA***.** L'architecture proposée a facilité l'intégration du flux avec les différentes couches de l'Internet des Objets à savoir la couche capteurs, la couche réseau de transmission de données, la couche stockage dans le Cloud et la publication des données et des événements recueillies vers les utilisateurs [125, 126]. L'architecture proposée permet de répondre au besoin de partage d'information sur le flux, la traçabilité et la collaboration des différents acteurs impliqués dans le flux par la publication et l'abonnement aux divers canaux de flux d'événements. Elle permet entre autre de gérer les notifications et l'interopérabilité entre les acteurs et les processus qui utilisent des technologies et des formats de données hétérogènes. Un prototype a été

développé pour tester les performances dans le cas de montée en charge de l'activité au niveau de l'utilisation des services de la plateforme, et les résultats sont satisfaisants au regard des plateformes de l'Internet des objets existantes à l'heure actuelle.

# Perspectives

Pour des raisons indépendantes de notre volonté nous n'avons pas pu approfondir les différentes pistes que nous avons explorées.

Au niveau de la démarche MDA nous projetons de rajouter un *model checker* pour vérifier les contraintes d'intégrité du modèle après la phase de transformation du PIM vers le PSM. Il est question aussi de rajouter un générateur de code pour transformer le PSM généré en endPoints conforme à la spécification Google Apple Engine (GAE).

En outre, la recherche des points de blocage du flux (deadlock) par l'outil CADP à l'aide de la spécification proposée serait intéressante surtout dans la phase de planification du workflow. Au niveau ontologique, nous projetons de publier l'ontologie spécifiée sous forme de service, afin que les workflows puissent l'utiliser comme un dictionnaire commun pour faciliter la spécification des workflows et des ressources engagées. Il est aussi question d'automatiser le format de communication interprocessus.

En revanche, le développement d'un prototype de tracking intégrant un module GPS et un module de transmission GSM est en cours d'étude pour faciliter la collecte de données sur les entités logistiques mobiles et leur transmission à la plateforme.

Par ailleurs, l'utilisation des méthodes d'optimisation, des algorithmes du Machine Learning et des techniques du Complex Event Processing (CEP) à des fins de prédiction du comportement du flux et de retro-planification est l'une des pistes que nous continuerons d'approfondir pour améliorer la solution developpée et répondre à des besoins de l'entreprise 4.0.

# Liste des Abréviations

**ATL** : ATLAS Transformation Language
**BPEL** : Business Process Execution Language
**BPMI** : Business Process Model Initiative
**BPML** : Business Process Modeling Language
**BPMN** : Business Process Modeling Notation
**CIM** : Computation Independent Model
**COM-SLOT** : Communauté de Services Logistiques sur l'internet des Objets
**DEVS** : Discrete Event Specification
**DSML** : Domain-Specific Modeling Language
**EDA** : Event Driven Approach
**EDP** : Équations aux Dérivées Partielles
**E-LOTOS** : Enhanced LOTOS
**GSM** : Guard-Stage-Milestone
**IAD** : Intelligence Artificielle Distribuée
**IoT** : Internet of Things
**LOTOS** : Language Of Temporal Ordering Specification
**LTI** : Laboratoire des Technologies Innovantes
**M2M** : Machine to machine (Communication Machine à machine)
**MDA** : Model Driven Architecture
**MIS** : Modélisation Information Système
**MOF** : Meta Object facility
**NAF** : Nato Architecture Framework
**OCL** : Object Constraint Language
**OMG** : Object Management Group
**OOSE** : Object Oriented Software Engineering
**PIM** : Platform Independent Model
**PSM** : Platform Specific Model
**QVT** : Query/View/Transform
**RdP** : Réseaux de Petri
**SADT** : Structured Analysis and Design Technique
**SED** : systèmes à événements discrets
**SLA** : Service Level Agreement
**UML** : Uniform Modeling Language
**UML-S** : UML for Web services composition
**UPJV** : Université de Picardie Jules Verne

# Références bibliographiques


[1] Nesrine SMATA, Cherif TOLBA, Dalila BOUDEBOUS, Senouci BENMANSOUR, Jaouad BOUKACHOUR, *Modélisation de la chaîne logistique en utilisant les réseaux de Petri continus,* CERENE, Université du Havre, 2011, France.

[2] Karim LABADI, *Contribution à la modélisation et à l'analyse de performances des systèmes logistiques à l'aide de nouveaux modèles de réseaux de Petri stochastiques,* Université de Technologie de Troyes, 2005, France.

[3] Caroline PRODHON, *Le Problème de Localisation Routage*, Thèse de Doctorat, Université de Technologie de Troyes, Optimisation et sûreté des systèmes, 2006, France.

[4] Julien FRANÇOIS, *Planification des chaînes logistiques : Modélisation du système décisionnel et performance*, Thèse de doctorat, Université Bordeaux 1, École doctorale des Sciences Physiques et de l'ingénieur, Productique, Décembre 2007, France.

[5] Jihène JLASSI, *Amélioration de la performance par la modélisation de flux logistiques des patients dans un service d'urgence hospitalier*, Thèse de Doctorat, Université de SFAX, Université Paris 8, Méthodes Quantitatives Productique et Génie Industriel, France. 2009.

[6] Mathieu DUPUY, *Contributions à l'analyse des systèmes industriels et aux problèmes d'ordonnancement à machines parallèles flexibles : application aux laboratoires de contrôle qualité en industrie pharmaceutique*, Thèse de Doctorat, Institut national Polytechnique de Toulouse, Ecole doctorale Systèmes, Systèmes Industriels, 2005, France.

[7] Christophe DUMEZ, *Approche dirigée par les modèles pour la spécification, la vérification formelle et la mise en œuvre de services Web composés*, Laboratoire Systèmes et Transports(SeT), Université de Technologie de Belfort-Montbéliard, 90000, Belfort 31 Août 2010, France.

[8] François GALLASSO, *Aide à la planification dans les chaînes logistiques en présence de demande flexible*, Ecole doctorale EDSYS, Systèmes industriels, 23 Avril 2007, France.

[9] Aïcha AMRANI-ZOUGGAR, *Impact des contrats d'approvisionnement sur la performance de la chaîne logistique : Modélisation et simulation*, Université Bordeaux 1, École Doctorale des sciences Physiques et de l'ingénieur, Productique, 20 Novembre 2009, France.

[10] Julien FRANCOIS. Planification des chaˆınes logistiques : mod´elisation du syst`eme d´ecisionnel et performance. Sciences de l'ing´enieur [physics]. Universit´e Sciences et Technologies - Bordeaux I, 2007. Fran ̧cais. <tel-00267825v1>

[11] Alexandre VILLEMINOT, *Modélisation et simulation de la logistique d'approvisionnement dans l'industrie automobile : application pour un grand constructeur,* Université de Nancy 1, 2004, France.

[12] Phuong NGA THANH, *Conception et planification stratégique des réseaux logistiques complexes,* Université de Nantes, 2008, France.

[13] Ayse AKBALIK, *Optimisation de la gestion intégrée des flux physiques dans une chaîne logistique : extensions du problème de dimensionnement de lot,* Grenoble, INPG, 2006, France.

[14] Sami SBOUI, *Modélisation, évaluation des performances et optimisation des flux logistiques d'un réseau d'entreprises partenaires dans la filière Textile-Habillement-Distribution*, Lille 1, France. 2003

[15] Senouci BENMANSOUR, *Modélisation de l'activité routière en vue de son intégration dans la chaîne logistique*, Université du Havre, 2009, France.

[16] Thi Le HOA VO, *Modélisation dynamique des flux logistiques de la filière avicole* française dans un contexte de crise sanitaire, Nantes, 2009, France.

[17] M BAKHOUYA. *Approche auto-adaptative à base d'agents mobiles et inspirée du système immunitaire de l'Homme pour la découverte de services dans les réseaux à grande échelle*. Thèse de doctorat de l'Université de Technologie de Belfort-Montbéliard et l'Université de Franche-Comté. 2005.

[18] Ali KANSOU, *Nouveaux algorithmes d'optimisation combinatoire pour les problèmes de tournées sur arcs*, Laboratoire de Mathématiques Appliquées du Havre, LMAH Université du Havre, France.

[19] Ludwig VON BERTALANFFY, *Théorie générale des systèmes, DUNOD, Traduction de l'ouvrage General System Theory*, publié par Georges Brazziler, 1968, Inc. New York.
[20] Alexandre PAUCHET, *Modélisation des systèmes Complexes, Introduction aux automates à états finis*, INSA Rouen.
[21] Brenda M. MICHELSON, *Even-Driven Architecture Overview,* Patricia Seybold Group, February 2, 2006.
[22] John D. POOLE, *Model-Driven Architecture: Vision, Standars, And Emerging Technologies*, ECOOP 2001, Workshop on Metamodeling and Adaptive Object Models, April 2001.
[23] Christine SOLMON, *Théorie des graphes et optimization dans les graphes*, LIRIS-CNR.
[24] Thomas SCHIEX, Simon DE GIVRY, Graphes, Algorithmes et modélisation, INRA, 24 Octobre 2013.
[25] Lucie MARTINET, *La thèse de Church*, 6 Juin 2008.
[26] Hervé P. HILLION, *Supervision des systèmes discontinus : définition d'un modèle hybride et pilotage en temps-réel. Supervision of batch systems: definition of a hybrid model and real-time control*, 1998, Toulouse, France.
[27] B. HELFFER à partir du texte établi par Thierry RAMOND, *Introduction aux Équations aux Dérivées partielles,* Université Paris-Sud, Janvier-Mai 2007
[28] Thommaso BOLOGNESI, CNUCE-C.N.R via S. Maria, Pisa, *Introduction to the ISO Specification Language LOTOS*, University of Twente, Netherlands,1987
[29] Maamar El Amine HAMRI, Grégory ZACHAREWICZ, *DEVS : Formalismes, Méthodes et outils,* 9th International Conference on Modeling, Optimisation & SIMulation performance, interoperability and safety for sustainable development.
[30] Bruno BOUZY, *UNIFIED MODELING LANGUAGE (UML).*
[31] http://internet-of-things.fr
[32] http://www.lemagit.fr/article/Internet-des-objets-les-usages-de-demain-la-guerre-economique-daujourdhui
[33] http://www.netgouvernance.org/REGARDS-ACTUALITE.PDF
[34] Gilles PACHE *« Logistique et entreprise virtuelle »,* Revue française de gestion 3/ 2005 (no156), p. 131-134 (http://www.cairn.info/revue-francaise-de-gestion-2005-3-page-131.htm.)
[35] Richard CALVI et al. *« Coopérer en conception pour améliorer les supply chains de demain »*, Revue française de gestion 3/ 2005 (no 156), p. 187-202 (http://www.cairn.info/revue-francaise-de-gestion-2005-3-page-187.htm.)
[36] Sofiane AYADI, *« Externalisation et création de valeur au sein de la « Supply Chain » : l'entreprise étendue »*, La Revue des Sciences de Gestion 2/ 2009 (n°236), p. 85-93 (http://www.cairn.info/revue-des-sciences-de-gestion-2009-2-page-85.htm.)
[37] Balamuralidhar P., Prateep MISRA, Arpan PAL., *SOFTWARE PLATFORMS FOR INTERNET OF THINGS AND M2M,* Journal of the Indian Institute of Science, Vol 93, No 3, 2013.
[38] MONONEN, P. ... et al., *TeleFOT, Field Operational Tests of aftermarket nomadic devices in vehicles, early results.* 2010 BEXCO. Proceedings of 17th ITS World Congress, 25th-29th October, Busan, South Korea, pp. 1-10, 2010.
[39] Miao WU, Ting-Jie LU, Fei-Yang LING, Jing SUN, Hui-Ying DU, *"Research on the architecture of Internet of Things",* Advanced Computer Theory and Engineering (ICACTE), 3rd International Conference on (Volume:5), Chengdu. 2010.
[40] Coetzee L., EKSTEEN, J., *"The Internet of Things - promise for the future? An introduction",* IST-Africa Conference Proceedings, May 2011, Gaborone
[41] Lu TAN, Neng WANG, *" Future internet: The Internet of Things",* Advanced Computer Theory and Engineering (ICACTE), 3rd International Conference on (Volume: 5), 2010.
[42] Zhonggui MA, Xinsheng SHANG, Xinxi FU, Feng LUO, "*The* architecture *and key technologies of Internet of Things in logistics"*, International Conference on Cyberspace Technology (CCT 2013), Beijing, China, Nov. 2013.
[43] The Internet of Things – new infographics, http://blog.bosch-si.com/theinternet-of-things-new-infographics/

[44] Vivek KUMAR SEHGAL, Patrick ANUBHAV, Lucky Rajpoot, "*A Comparative Study of Cyber Physical Cloud, Cloud of Sensors and Internet of Things: Their Ideology, Similarities and Differences*", IEEE International Advance Computing Conference (IACC), 2014.

[45] Madria, S. , Kumar V., Dalvi, R., "*Sensor Cloud: A Cloud of Virtual Sensors*", Software, IEEE (Volume:31, Issue: 2 ), 12 November 2013.

[46] Kausalya DEVI, P., RAVICHANDRAN, N. Shakeel Mohammed HANIF, S., "An automated Cloud based vehicular emission control system using Zigbee", IEEE International Conference on Smart Structures and Systems (ICSSS), Chennai, India, 2013.

[47] ZHONGWEN Li , Yi XIE, Chengbin WU, Binbin DING, "*A security query protol of ONS in EPC system*", Anti-Counterfeiting, Security and Identification (ASID), International Conference , 2012.

[48] Perera, C., ZASLAVSKY, A., Liu, C.H. Compton, M., Christen, P., Georgakopoulos, D., "*Sensor Search Techniques for Sensing as a Service Architecture for the Internet of Things*", Sensors Journal, IEEE (Volume:14, Issue: 2 ), 2013.

[49] Marcelo Dias DE AMORIM, Serge FDIDA, Nathalie MITTON, Loïc SCHMIDT, David SIMPLOT-RYL, "Distributed Planetary Object Name Service: Issues and Design Principles", Rapport de recherche n° 704, Centre de recherche INRIA Lille – Nord Europe, Sep 2009.

[50] TAO, F., CHENG, Y. Xu, Li DA, ZHANG, L. , Li, Bo Hu, "*CCIoT-CMfg: Cloud Computing and Internet of Things based Cloud Manufacturing Service System*", IEEE Transactions on Industrial Informatics, (Volume:10 , Issue: 2 ), Fev 2014.

[51] Mao CUIYUN, Han YUANHANG, "*Discussion on the Application of Internet of Things in Logistics Production Management"*, International Conference on E-Business and E-Government (ICEE), 2010.

[52] Kausalya Devi, P., Ravichandran, N., Shakeel Mohammed Hanif, S., "*An automated Cloud based vehicular emission control system using Zigbee"*, IEEE International Conference, Smart Structures and Systems (ICSSS), 2013.

[53] Pierre-Jean BENGHOZI, Sylvain BUREAU, Françoise MASSIT-FOLLEA, "*L'internet des objets: Quelles enjeux pour l'Europe*", Éditions de la Maison des sciences de l'homme, 2009.

[54] Gogliano, O.G., Cugnasca C.E., "*An Overview Of The EPCglobal Network*", Latin America Transactions, IEEE, 17 Sept. 2013

[55] Dominique PARET, Xavier BOUTONNIER, Youssef HOUITI, "*NFC NearField Communication, Principes et applications de la communication en champ propre"*, DUNOD, Paris, 2012.

[56] Surya MICHRANDI NASUTION, Emir MAULUDI HUSNI, Aciek Ida WURYANDARI, "*Prototype of train ticketing application using near field communication (NFC) technology on android device*", International Conference on System Engineering and Technology,Indonesia, 2012.

[57] Zigbee alliance, "*Network device: gateway specification*", version 1.0, March 23rd, 2011

[58] Alexiou, A., Wireless World 2020: Radio Interface Challenges and Technology Enablers, Février 2014

[59] Vignesh, P.J.A., Vignesh, G.K., "*Relocating Vehicles to Avoid Traffic Collision through Wireless Sensor Networks"*, Computational Intelligence, Communication Systems and Networks (CICSyN), Fourth International Conference, 2012.

[60] Olonibua ABIODU, AKINGBADE KAYODE Francis, "*Wireless Transmission of Biomedical Signals Using the Zigbee Technology*", IEEE International Conference on Emerging & Sustainable Technologies for Power & ICT in a Developing Society (NIGERCON), 2013.

[61] Decarli, N., Guidi, F. ; Dardari, D., "*A Novel Joint RFID and Radar Sensor Network for Passive Localization: Design and Performance Bounds*", Selected Topics in Signal Processing, IEEE Journal of (Volume:8 , Issue: 1 ), 24 Octobre 2013.

[62] Al-SHRAIDEH, "*Host Identity Protocol, Networking",* International Conference on Systems and International Conference on Mobile Communications and Learning Technologies. ICN/ICONS/MCL, 2006.

[63] TONGRANG Fan, YANZHAO Chen, "*A scheme of data management in the Internet of Things*, 2nd IEEE International Conference on Network Infrastructure and Digital Content, Beijing. 2010

[64] Perera, C., Jayaraman, P.P., ZASLAVSKY A., GEORGAKOPOULOS, D., Christen, P. "*MOSDEN: An Internet of Things Middleware for Resource Constrained Mobile Devices"*, 47th Hawaii International Conference on System Sciences (HICSS), 2014.

[65] Aazam MOHAMMAD, Khan IMRAN, Alsaffar, Aymen Abdullah, Huh, Eui-Nam, "*Cloud of Things: Integrating Internet of Things and Cloud computing and the issues involved*", 11th International Conference on Applied Sciences and Technology (IBCAST), 2014.

[66] Eric GAUDREAU, Bruno AGARD, Martin TREPANIER, Pierre BAPTISTE, "*Pilotage réactif des systèmes de production à l'aide de capteurs intelligents"*, 6e Congrès international de génie industriel — Besançon (France), 7-10 juin 2005.

[67] NING, H., Liu, H., Yang, L., "*Aggregated-proof Based Hierarchical Authentication Scheme for the Internet of Things*", IEEE Transactions on Parallel and Distributed Systems, (Volume: PP, Issue: 99 ), 14 mars 2014.

[68] Bayat-SARMADI, S., Kermani, M.M., Azarderakhsh, R., Chiou-Yng Lee, "*Dual-Basis Superserial Multipliers for Secure Applications and Lightweight Cryptographic Architectures"*, IEEE Transactions on Circuits and Systems II: Express Briefs, (Vol. 61 , Nu. 2), 2013.

[69] Oh SE WON, Kim, Hyeon Soo, "*Decentralized access permission control using resource-oriented architecture for the Web of Things",* 16th International Conference on Advanced Communication Technology (ICACT), 2014, Pyeongchang, Korea (South).

[70] François BERTUCCI, Béatrice LORIOD, Rebecca TAGETT, Samuel GRANJEAUD, Daniel BIRNBAUM, Catherine NGUYEN, Rémi HOULGATTE, *"Puces à ADN: technologie et applications"*. Bulletin du Cancer. Volume 88, Numéro 3, 243-52, Mars 2001.

[71] Alexander GLUHAK, Srdjan KRCO, Michele NATI, Dennis PFISTERER, Nathalie MITTON and Tahiry RAZAFINDRALAMBO, "*A survey on facilities for experimental Internet of Things research*", IEEE Communications Magazine, November 2011.

[72] A NAIT-SIDI-MOH, M BAKHOUYA, J GABER, M WACK, "*Geopositioning and Mobility*". Wiley-ISTE, Networks and Telecommunications series; 272 pages. isbn:978-1-84821-567-2. Septembre 2013.

[73] Jong HYUN Lim, I-Jeng WANG, Andreas TERZIS, "*Tracking A Non-cooperative Mobile Target Using Low-power Pulsed Doppler Radars*". Research report, Michigan Technological University. 2011.

[74] José ROUILLARD, "*Contextual QR Codes*". The Third International Multi-Conference on Computing in the Global Information Technology (ICCGI08). Athens, Greece, July 27 - August 1, 2008

[75] BOOCH, G. Object Oriented Design with Applications. *Benjamin Cummings, California,* 1991.

[76] Jacobson, I., Christerson, M., Jonsson, P., Overgaard, G. Object-Oriented Software Engineering – A Use Case Driven Approach. *ACM Press, Addison-Wesley, Mass,* 1992.

[77] OMG: MDA Guide Version 1.0.1, http://www.omg.org/docs/omg/03-06-01.pdf, 2003.

[78] OMG: OMG Unified Modeling Language (OMG UML), Superstructure, V2.1.2, http://www.omg.org/spec/UML/2.1.2/Superstructure/PDF , 2007.

[79] Rumbaugh, J., Blaha, M., Premerlani, W., Eddy, F., Lorensen, W. Object-Oriented Modeling and Design. *Prentice Hall, New Jersey,* 1991.

[80] Stuart KENT, *Model Driven Engineering*, University of Kent, Canterbury, UK

[81] http://www.gs1.org/epcglobal

[82] http://www.gs1.org/gs1-innovation-network

[83] Enrique DE ARGAEZ, "Demand Driven Supply Networks DDSN", Supply Chain report (IDC),

[84] Van der Aalst, W., Weske M., & Gru¨nbauer, D. (2005). Case handling: A new paradigm for business process support. Data & Knowledge Engineering, 53(2), 129–162

[85] *Workflow Management Coalition* (WFMC), *Workflow Process definition Interface*, *XML Process Definition Language (XPDL)*, WFMC-TC-1025, Oct. 25 2002

[86] *Object Management Group* (OMG), *Business Process Model and Notation* (BPMN) Version 2.0.2, formal/2013-12-09, http://www.omg.org/spec/BPMN

[87] Thomas R. Gruber, *A Translation Approach to Portable Ontology Specifications*, Appeared in Knowledge Acquisition, 5(2):199-220, 1993.

[88] Sofiane CHEMAA, Faycal BACHTARZI , Allaoua CHAOUI, *A High-level Petri Net Based Approach for Modeling and Composition of Web Services,* Procedia Computer Science Volume 9, 2012, Pages 469-478, Proceedings of the International Conference on Computational Science, ICCS 2012, doi:10.1016/j.procs.2012.04.05

[89] http://www.eurodecision.com/presse/optiflux-optimiser-flux-logistiques

[90] http://www.predit.prd.fr/predit4/document/43571
[91] http://www.eurodecision.fr/recherche-et-developpement-recherche-operationnelle
[92] http://www.gs1.org/epcglobal
[93] http://www.ifm.com
[94] http://www.anrt.asso.fr/com/imgAdmin/1341497721111.pdf
[95] http://www.fp7-aspire.eu
[96] http://www.hydramiddleware.eu
[97] https://www.cetic.be/MidFlex
[98] http://www.traser-project.eu
[99] http://www.grifs-project.eu
[100] OMG Unified Modeling language (UML) Version 2.5
[101] OMG Semantic of Business Vocabulary and Business Rules Version 1.3 (SBVR), http://www.omg.org/spec/SBVR/1.3/PDF
[102] Object Management Group (OMG), Object Constraint Language V2.4 (OCL), http://www.omg.org/spec/OCL/20090501/EssentialOCL.emof
[103] Object Management Group (OMG), Meta Object Facility (MOF) 2.0 Query/View/Transformation Specification V1.2 (QVT), http://www.omg.org/spec/QVT/1.2 .
[104] Darius SILINGAS, Ruslanas VITIUTINAS, Andrius ARMONAS, Lina NEMURAITE, *Domain-specific modeling environment based on UML profiles*.
[105] Van der Aalst, W., Weske, M., & Gru¨nbauer, D. (2005). Case handling: A new paradigm for business process support. Data & Knowledge Engineering, 53(2), 129–162
[106] Object *Management Group* (OMG), *Business Process Model and Notation* (BPMN) Version 2.0.2, formal/2013-12-09, http://www.omg.org/spec/BPMN
[107] *Workflow Management Coalition* (WFMC), *Workflow Process definition Interface*, *XML Process Definition Language (XPDL)*, WFMC-TC-1025, Oct. 25 2002
[108] Tommaso BOLOGNESI, Ed BRINKSMA, *Introduction to the ISO Specification Language* LOTOS
[109] Robin MILNER, *a Calculus of Communicating System*, Springer-Verlag, lecture Notes in Computer Science Vol.92, 1980
[110] Peter WELCH, Neil BROWN, *Communicating Sequential Processes for Java* (*JCSP*), April, 2014
[111] Telecom Design, TD1208 Rev 1.4 (08/14). Evaluation board user's guide; 2014.
[112] Nicolai M. JOSUTTIS. SOA in practice The Art of Distributed System Design, O'REILLY Media, Pages: 344. August 2007.
[113] Arnon ROTEM-GAL-OZ, "SOA Patterns", Manning Publications Co., Sept. 2012.
[114] WEI Li, SONGLIN Hu, Jiantao Li, Hans-Arno JACOBSEN. Community Clustering for Distributed Publish/Subscribe Systems. Cluster Computing (CLUSTER), IEEE International Conference, Beijing. Sept. 2012, Page(s): 24-28. 2012.
[115] K Dar, M BAKHOUYA, J GABER, M WACK, P LORENZ. Wireless Communication Technologies for ITS Applications, IEEE Communications Magazine 48: 5. 156-@jkhjb; 2010.
[116] Centre d'étude et d'expertise sur les risques, l'environnement, la mobilité et l'aménagement (CEREMA), *Transport de marchandises, Caractéristiques de l'offre et capacité des modes de transport*, Paris, 2008
[117] Service d'études sur les transports, les routes et leurs aménagements (Sétra), *rapport d'études, Les matériels ferroviaires de voyageurs sur le réseau ferré national*. Décembre 2013, France
[118] Nicola GUARINO, Daniel OBERLE, and Steffen STAAB, *What Is an Ontology*, Handbook on ontologies, 1-17., Springer Berlin Heidelberg, 2009
[119] Ian NILES, Adam PEASE, *Towards a standard upper ontology,* FOIS O1 Proceedings of the international conference on formal Ontology in Information Systems – volume 2001 Pages 2-9, USA, Oct. 17-19, 2001.
[120] Stefan POSLAD, Stuart E. MIDDLETON, Fernando CHAVES, Ran TAO, Ocal NECMIOGLU, Ulrich BÜGEL, *A semantic IoT Early Warning System for Natural Environment Crisis Management*, IEEE Transactions on Emerging Topics in Computing, Volume :3 Issue :2., Mai 2015.
[121] A-M. Roxin, J GABER, M WACK, A. NAIT-SIDI-MOH, *Survey of wireless geolocation techniques*, IEEE Globecom Workshops, Washington DC, USA, 9 Pages, 2007.

[122] N JABEUR, A. NAIT-SIDI-MOH, M-M. BARKIA, *A Bully Approach for Competitive Redundancy in Heterogeneous Wireless Sensor Network*, Procedia Computer Science 83, 628-635, 2016.

[123] David R. GNIMPIEBA ZANFACK, Ahmed NAIT-SIDI-MOH, David DURAND, Jérôme FORTIN, *Internet des objets et interopérabilité des flux logistiques: état de l'art et perspectives,* UbiMob2014 : 10èmes journées francophones Mobilité et Ubiquité, 5-6 Juin 2014, Sophia Antipolis – France

[124] David R. GNIMPIEBA ZANFACK, Ahmed NAIT-SIDI-MOH, David DURAND, Jérôme Fortin, *A Guard-Stage-Milestone based approach for modeling physical workflow in the context of Internet of Things,* Third World Conference on Complex Systems (WCCS) November 23-25, Marrakech – Morocco 2015

[125] David R. GNIMPIEBA ZANFACK, NAIT-SIDI-MOH A., DURAND D., FORTIN J.*, Publish and Subscribe Pattern for Designing Demand Driven Supply Networks,* Product Lifecycle Management in the Era of Internet of Things. PLM 2015. IFIP Advances in Information and Communication Technology, vol 467. Springer, DOI: 10.1007/978-3-319-33111-9_5

[126] David R. GNIMPIEBA ZANFACK, NAIT-SIDI-MOH A., DURAND D., Fortin J., *Using Internet of Things Technologies for a Collaborative Supply Chain: Application to Tracking of Pallets and Containers*, Procedia Computer Science, Volume 56, 2015, Pages 550-557, DOI: 10.1016/j.procs.2015.07.251