République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Mohamed Seddik Ben Yahia de Jijel
Faculté de Science Exactes et Informatique
Département Informatique

Mémoire de fin d'études

Pour l'obtention du diplôme Master de recherche en Informatique

Option :Système d'information et aide à la décision

Une Approche Outillée pour l'Intégration des Processus Métier (BPMN) dans le Développement des Applications de l'Internet des Objets

---

Réalisé par :
Méghaichi Moufida
Meghlaoui Hadjer

Encadré par :
Dr. Kerkouche Elhillali
Dr. Khalfaoui Khaled

Promotion : 2021/2022

Remerciements

Tout d'abord, nous voudrions remercier « ALLAH » qui nous a donné la force et la patience pour accomplir ce modeste travail.

Nous tenons à remercier notre encadreur Mr **Kerkouche El-hillali** et le co-encadreur Mr **Khalfaoui Khaled** pour leurs précieux conseils et leur aide durant la période du travail.

Nous exprimons également nos sincères remerciements et notre gratitude à **Sofia Abbas** pour son aide et pour tous les conseils et informations qu'elle nous a fournis dans ce travail.

Nous tenons à remercier tout particulièrement les membres du jury pour l'honneur qu'ils me font d'avoir accepté de rapporter ce mémoire.

Nous tenons à remercier tout le corps enseignant de l'informatique qui a contribué à notre formation.

Enfin, nous tenons remercier à tous les personnes qui ont y contribué de près ou de loin.

Merci à tous.

Résumé

L'Internet des Objets (Ido)est une innovation dans le domaine de l'information et de la communication, c'est un concept informatique qui exprime l'idée de connecter à Internet divers appareils physiques ayant la capacité de se lier et d'échanger des données entre eux. La modélisation des applications Ido est devenue une exigence ; De nombreux chercheurs s'intéressent à la modélisation des applications Ido, et parmi les méthodes les plus importantes de modélisation d'Ido : le BPMN (Business Process Modeling Notation) Parce qu'il répond à la plupart des exigences de l'Internet des objets.

Le but du travail présenté dans ce mémoire est de construire un environnement de modélisation graphique capable de modéliser les applications Ido de la santé basée sur le langage BPMN 2.0 Suivant l'approche "Ingénierie Dirigée" Selon les modèles (IDM)" et ses outils. en utilisant la plate-forme Eclipse Modeling Tools et le langage Acceleo pour automatiser la génération de code Maude.

Mots clés : Ido, Processus Métier, BPMN, Ingénierie Dirigée par les Modèles, Metamodélisation, Eclipse Modeling Tools,langue Maude.

Abstract

The Internet of Things (IoT) is an innovation in the field of information and of communication, it is a computer concept that expresses the idea of connecting to Internet various physical devices with the ability to link and exchange data between them. The modeling of IoT applications has become a requirement ; Many researchers are interested in modeling IoT applications, and among the methods the most important IoT modeling tools : BPMN (Business Process Modeling Notation) Because it meets most of the requirements of the Internet of Things.

The aim of the work presented in this thesis is to build an environment of graphical modeling capable of modeling health IoT applications based on the BPMN 2.0 language Following the "Directed Engineering" approach According to the models (IDM)" and his tools. using the Eclipse Modeling Tools platform and Acceleo language to automate code Maude generation.

Keywords : IoT, Business Process, BPMN, Model Driven Engineering, Metamodeling, Eclipse Modeling Tools, Maude language.

Table des matières

Remerciements . . . . . **I**

Résumé . . . . . **II**

Abstract . . . . . **III**

1 Internet des Objets **3**
1.1 Introduction 4
1.2 Définition de l'Internet Des Objets 4
1.3 Historique 4
1.4 Domaines d'application de l'IdO 5
1.4.1 Maisons intelligentes 5
1.4.2 Ville intelligente 6
1.4.3 Bâtiments intelligents 8
1.4.4 Soins de santé 9
1.4.5 Agriculture 10
1.5 Les avantages de l'IdO 12
1.6 Les difficultés associés à l' IdO 12
1.7 L'architecture de l'internet des objets 13
1.8 Comment fonctionne l'Internet des objets ? 14
1.9 Modélisation des applications IdO dans le plus haut niveau d'abstraction : 15
1.9.1 Notation de modélisation 15
1.9.1.1 SYSML (System Modeling Language) 15
1.9.1.2 TheThingML (Internet of Things Modeling Language) . . 15
1.9.1.3 BPMN (Business Process Modeling Notation) 16
1.9.1.4 UML4IoT 16
1.9.2 Modèle conceptuel 16
1.10 Conclusion 18

2 BPMN (Business Process Modeling Notation) **19**
2.1 Introduction 20
2.2 Processus métier 20
2.2.1 Définition de Processus 20
2.2.2 Définition de Processus métier 20
2.3 Gestion des processus métier (BPM) 21
2.3.1 Définition Gestion des processus métier (BPM) 21
2.3.2 Cycle de vie d'un processus métier 21
2.4 BPMN 2.0 22

2.4.1 Le standard BPMN . . . 22
2.4.2 L'historique de BPMN . . . 23
2.4.3 Notation de BPMN 2.0 . . . 24
2.4.4 Types de diagrammes BPMN . . . 34
2.4.4.1 Diagramme de processus (Orchestration) . . . 34
2.4.4.2 Chorégraphie . . . 35
2.4.4.3 Diagramme de collaboration . . . 36
2.4.4.4 diagramme de conversation . . . 36
2.5 Modélisation de l'IdO en utilisant BPMN . . . 37
2.5.1 Model de domaine de l'IdO en utilisant la norme BPMN . . . 37
2.5.1.1 Entité physique . . . 37
2.5.1.2 Utilisateur . . . 37
2.5.1.3 Entité virtuelle . . . 38
2.5.1.4 Entité augmentée . . . 38
2.5.1.5 Appareil IdO . . . 38
2.5.1.6 Ressource et service . . . 38
2.6 Un extensions BPMN pour la modélisation de l'IdO . . . 39
2.7 Une comparaison entre BPMN 2.0, eEPC et UML2.3 pour la modélisation de l'IdO . . . 40
2.8 Conclusion . . . 41

3 Ingénierie dirigée par les modèles . . . 42
3.1 Introduction . . . 43
3.2 Définition . . . 43
3.3 Principes généraux de l'IDM . . . 43
3.4 Notion de base en Ingénierie Dirigée par les Modèles . . . 44
3.4.1 Système . . . 44
3.4.2 Modèle . . . 44
3.4.3 Meta-modèle . . . 44
3.4.4 Méta-méta-modèle (MOF) . . . 45
3.5 L'architecture à quatre niveaux de MOF . . . 45
3.6 Transformation des modèles . . . 46
3.6.1 Définition de base . . . 46
3.6.2 Types de transformation . . . 47
3.6.2.1 Transformation horizontale : . . . 47
3.6.2.2 Transformation verticale : . . . 48
3.6.2.3 Transformation oblique : . . . 48
3.6.3 Les approches de transformation des Modèles : . . . 48
3.6.3.1 Transformation de modèle à modèle M2M : . . . 48
3.6.3.2 Transformation modèle vers texte, M2T : . . . 49
3.7 Avantage et inconvénient de l'IDM . . . 50
3.7.1 Avantages de l'IDM : . . . 50
3.7.2 Inconvénients de l'IDM . . . 50
3.8 Outils de l'IDM . . . 51
3.9 Outils de l'IDM utiliser dans notre travail . . . 51
3.9.1 Eclipse Modeling Tools . . . 51

3.9.2 Plate-forme de métamodélisation sous Eclipse Modeling Tools . . . 52
3.9.2.1 Eclipse Modeling Framework (EMF) 52
3.9.2.2 Sirius 54
3.9.2.3 Acceleo 56
3.10 Langage Maude 56
3.10.1 Modules dans Maude 57
3.10.2 Avantages de Maude : 58
3.11 Conclusion 58

4 Développement d'un Environnement de Modélisation Graphique pour les Applications IdO 59
4.1 Introduction 60
4.2 Description de notre approche 60
4.3 Métamodelisation des application IdO avec BPMN 60
4.3.1 Composants du Méta-modèle 64
4.3.1.1 Classes 64
4.3.1.2 Associations 66
4.3.2 Génération de code pour le méta-modèle crée 68
4.4 Créations de l'environnement Graphique sous Sirius 69
4.4.1 Creation de projet Sirius 69
4.4.2 Création de la palette 71
4.5 Transformation des modèle en Maude En utilisant Acceleo 74
4.5.1 Définition du modèle fonctionnel de BPMN_IdO en Maude 74
4.5.2 Définition des règles de transformation 75
4.5.3 Creation de projet Acceleo pour les génération de code Maude . . . 78
4.6 Cas étudier 81
4.6.1 Description de cas etudier 81
4.6.2 Modélisation de l'application Soins de santé dans notre environnement 81
4.6.3 Transformation de modèle crée en Maude 82
4.7 Écxution dans maude 84
4.8 Conclusion 85

Bibliographie 87

Table des figures

1.1 La classification des domaines de la ville intelligente avec des composants et des domaines d'application connexes. 7
1.2 le concept de l'Internet des objets dans le domaine de la santé. 10
1.3 Une illustration des applications IdO pour l'agriculture intelligente. 11
1.4 Les architectures de l'IdO les plus courantes (trois couches et cinq couches). 13
1.5 Modèle conceptuel pour l'IdO. 17

2.1 Un exemple illustre comment fonction un processus métier. 21
2.2 Le cycle de vie du BPM 22
2.3 Historique de l'évolution du standard BPMN 24
2.4 Les formes de sous-processous. $\text{sous}_p ros$ 26
2.5 Les types de sous-processous. 27
2.6 les formes de Activité d'appel 27
2.7 Événements dans BPMN 2.0. 27
2.8 Les types d'événements de BPMN2.0 28
2.9 Les passerelles de BPMN2.0 30
2.10 Les données de BPMN 2.0 30
2.11 Les flux de séquence de BPMN 2.0. 31
2.12 Les flux de message de BPMN 2.0. 32
2.13 Les associations de BPMN 2.0. 32
2.14 La représentation d'un pool comme boîte noire en BPMN 2.0. 32
2.15 La représentation d'un pool en BPMN 2.0. 33
2.16 La représentation d'une lane en BPMN 2.0. 33
2.17 Les artéfacts 33
2.18 Exemple de processus privé. 34
2.19 Exemple de processus public. 35
2.20 Diagramme de chorégraphie. 35
2.21 Diagramme de collaboration. 36
2.22 Diagramme de collaboration. 37
2.23 Tâche de détection. 39
2.24 Tâche Actionnement. 39

3.1 architecture à quatre niveaux. 46
3.2 Architecture de la transformation des modèles. 47
3.3 Les types de transformations. 48

4.1 La création d'un projet EMF 61
4.2 Extension du méta-modèle BPMN 2.0 pour IdO sous Eclipse Modeling Tools 63

4.3 La génération du fichier model_final_iot.genmodel. 68
4.4 Le résultat de la génération du fichier model_final_iot.genmodel. 68
4.5 Création d'un projet de type « Viewpoint Specification Project » 69
4.6 Le choix de type de représentation 69
4.7 Créer un éditeur de diagramme 70
4.8 Création les éléments de notre modèle par ces images » 70
4.9 Création séquence flow » 71
4.10 Création de la palette » 72
4.11 Création de la palette » 72
4.12 Création de la palette 73
4.13 Unité fonctionnelle de base BPMN_IdO 75
4.14 création de projet de type acccleo 78
4.15 code acceleo partie 1 79
4.16 code acceleo partie 2 79
4.17 code acceleo partie 3 81
4.18 Le processus soins santé créé dans l'environnement graphique proposé. . . 82
4.19 la configuration accleo. 83
4.20 le résultat de configuration accleo. 83
4.21 Le code Maude générer. 84
4.22 Le resultat d'éxécution de code maude. 84

Liste des tableaux

2.2 Les types des tâches dans BPMN 2.0. 26
2.3 Caractéristiques IdO par notations BPM 41

4.2 Les classes du Méta-modèle 66
4.4 Les Composition du Méta-modèle 67
4.6 Les référence du Méta-modèle 67
4.8 L'héritage du Méta-modèle 68
4.10 les règles de transformation en Maude (Sequence Flows) 77
4.12 les règles de transformation en Maude (flux de message) 78

Liste des sigles et acronymes

IdO	*Internet des Objets*
IoT	*Internet of Things*
BPM	*Business Process Management*
BPMN	*Business Process Model and Notation*
OMG	*Object Management Group*
IDM	*Ingénierie Dirigée par les Modelés*
EMF	*Eclipse Modeling Framework*
UML	*Unified Modeling Language*
XML	*Extensible Markup Language*
XMI	*XML Metadata Interchange*
XED	*XML Enabled Directory*
MOF	*Meta Object Facility*

Introduction générale

L'Internet des Objets ou Interent Of Things (IoT) en anglais est devenu l'une des disciplines les plus étudiées de nos jours, car elle est une révolution de la technologie de l'information et de la communication. Il est caractérisé par un ensemble d'objets intelligents (électroniques, logiciels, capteurs, actionneurs) connectés et organisés en réseau pouvant être connectés soit entre eux soit avec l'environnement extérieur. Ces appareils sont capables de percevoir et d'échanger des informations précises entre eux afin d'atteindre les buts recherchés. L'importance de l'internet des objets est qu'il a un énorme potentiel pour changer nos vies. Il augmente l'efficacité, réduit les coûts et améliore tout, de la maintenance, de l'industrie, des transports et de la sécurité aux soins de santé et bien d'autres domaines. C'est ce qui en a fait le centre d'attention des entreprises qui concurrencent pour développer et améliorer cette technologie.

D'autre part, les organisations utilisent de plus en plus des processus métier. Un processus métier est défini comme un ensemble d'évènements, des activités et d'activités et de points de décision inter-connectés qui incluent des acteurs et des ressources qui conduisent collectivement à un résultat de valeur pour une organisation ou un client. Il est devenu possible pour les opérations métier d'utiliser l'internet des objets pour fournir des informations sur le monde réel ainsi que pour mettre en œuvre une partie des opérations métier. Mais les appareils IdO sont de nature hétérogène. Ils sont diffèrent en termes de protocoles de communication, de modèles d'interaction, de puissance de calcul et de stockage, c'est pourquoi les concepteurs métier se tournent vers des processus utilisant des langages de haut niveau (tels que Business Process Modeling Notation désormais simplement appelés BPMN), parce qu'ils doivent connaître le domaine, mais n'ont pas besoin de connaissances particulières en programmation des appareils IdO, et ils ne veulent pas faire face à leur hétérogénéité. Alors Les méthodes actuelles basées sur BPMN permettent aux concepteurs de définir à la fois les processus métier et le comportement des appareils IdO au même niveau d'abstraction.

Le but de notre travail est de développer un éditeur de modèle graphique de haut niveau d'abstraction pour les applications IdO utilisant BPMN 2.0, Pour ce faire, nous allons adopter la démarche Ingénierie Dirigée par les Modèles (IDM) (c'est est une technique de développement pour simplifier et de mieux maîtriser le processus de développement de systèmes). Nous avons donc créé un méta-modèle d'extension BPMN qui serait plus adapté aux application de l'IdO. pour réaliser notre travail, nous avons utilisé Eclipse Modeling Tools en exploitant ses plugins EMF (Eclipse Modeling Framework), la meta-modilisation et Sirius pour la création de la palette, nous avons utilisé aussi Acceleo pour la transformation des modèles crées dans notre palette a un code Maude équivalent dans le but de la vérification et de l'analyse des ces models.

Plan du mémoire : Notre mémoire et divisé en quatre chapitres :

- Dans le premier chapitre intitulé l'interent des objet ; nous avons parlé de l'Internet des objets, l'historique, les domaines d'application, l'architecture et la modélisation au plus haut niveau d'abstraction.
- Dans le deuxième chapitre intitulé les business process modeling and notation ; nous allons introduire une brève description du processus métier et de BPM .Puis nous nous intéresserons en détails a la notation BPMN .Qu'est-ce que bpmn, les types de ses diagrames et les détails de ses éléments.
- Le troisième chapitre intitulé ingénierie dirigée par les modeles présente l'Ingénierie Dirigée par les Modèles (IDM) et les notion de base en Ingénierie Dirigée par les Modèles ainsi que les outils utilisés pour déployer notre approche.
- Le quatrième et dernier chapitre présente les détails de la partie mise en oeuvre de notre travail pour le développement d'un environnement de modélisation graphique pour les applications IdO.

Chapitre 1

Internet des Objets

1.1 Introduction

Le XXIE siècle est témoin d'une révolution numérique rapide et de développements dans divers domaines parmi eux, Internet, qui a fait du monde un petit village. Cette folie technologique s'est considérablement développée dans la mesure où elle s'est étendue aux choses matérielles et les a reliées à Internet pour communiquer et interagir entre elles d'une part et avec humains d'autre part, ou ce qu'on appelle l'internet des objets. Cette technologie est censée améliorer et faciliter la vie des gens dans divers domaines tels que l'industrie, l'énergie, les transports, la santé, l'éducation, le tourisme, l'agriculture, l'environnement, la sécurité, le divertissement, le luxe, et d'autres secteurs. Et c'est devenu l'une des choses que les entreprises les plus célèbres et les plus grandes leur ont donné un espace important ; en raison de son importance à le temps actuelle, et en raison de son brillant avenir. Dans ce chapitre nous allons introduire les concepts généraux de l'IdO, en suite nous allons voir la modélisation de ce dernier dans le plus haut niveau d'abstraction.

1.2 Définition de l'Internet Des Objets

Plusieurs définitions ont été données pour mieux comprendre le concept de l'Internet des Objets, parmi ces définitions on peut citer les suivantes :

- D'après Le CERP-IDO (Cluster des projets européens de recherche sur l'Internet des Objets), il définit l'Internet des objets comme : « une infrastructure de réseau mondial dynamique avec des capacités d'auto configuration basées sur des protocoles de communication standard et interopérables où les « objets » physiques et virtuels ont des identités, des attributs physiques et des personnalités virtuelles. Et utilisent des interfaces intelligentes, et sont parfaitement intégrés dans le réseau d'information ». [1]
- Les membres du groupe RFD ont donné la définition de l'Internet des objets comme « les objets qui sont connectés à travers le monde et possèdent des adresses basées sur certains protocoles standard pour la communication ». [2]
- A la suite d'un travail de recherche effectué par les scientifiques sur Internet des objets, ils ont dit que « Les choses exigent une participation énergique à la domaine d'activité, des informations où ils communiquer entre eux par échanger les données entre eux et réagir à l'environnement du monde réel et les influencer en activant les œuvres et des processus qui créent des actions sans intervention humaine ». [2]

1.3 Historique

L'Internet des objets (IdO) prend rapidement pied dans le secteur de la technologie ; et a pu évoluer comme suit [3][4] :

1974: Distributeurs automatiques de billets Il peut être considéré comme l'une des premières choses intelligentes, qui a été mise.

1982: A l'université Carnegie Mellon, il devient le premier appareil connecté à Internet et contrôlant des équipements électriques tels que cafetières et autres machines.

1991: Mark Weiser affirme que le monde connecté des objets est conçu pour aider les gens dans leurs activités de manière discrète... Dans son article général, « The Computer for the 21st Century », cet ouvrage a depuis été classé parmi les articles académiques les plus cités dans des disciplines académiques connexes qui imaginent un monde connecté d'objets du quotidien.

1999: L'homme d'affaires britannique Kevin Ashton utilise pour la première fois le terme Internet des objets (IdO), dans une présentation qu'il donne à Procter Gamble. À l'époque, il était cofondateur et PDG de l'Auto ID Center du MIT, et la vision de l'Internet des objets était basée sur l'identification par radiofréquence, ou RFID (Radio Frequency Identification La même année, Gershenfeld publie son ouvrage « When Things Start Penser », dans lequel il imagine le développement du World Wide Web.

2005: L'UIT (Internationale des Télécommunications) définit l'IdO comme un concept qui permet aux personnes et aux objets Être connecté n'importe quand, n'importe où, avec n'importe quoi et n'importe qui, et traitant de la relation entre le monde réel et le monde virtuel.

2009: Popularisé par CISCO affirme un concept simple L'IdO est né quand plus de choses sont connectées via Internet en tant qu'être humains.

IEEE 2015, 2017: l'initiative IEEE IdO donne aux membres de sa communauté une Possibilité de contribuer à la définition de l'IdO Peut dire que L'IdO est un réseau qui connecte des « choses » identifiables de manière unique à Internet. Et est la représentation contient des informations telles que l'identité, le statut, l'emplacement de la chose ou tout autre des informations commerciales, sociales ou privées pertinentes. Les choses offrent des services, avec ou sans intervention humaine, grâce à l'exploitation d'une capacité unique d'identification, de saisie et de communication de données et d'actionnement.

Selon les prévisions 2015-2020 de l'International Data Corporation (IDC) sur l'Internet des objets dans le monde, 30 milliards d'objets connectés (autonomes), devraient faire partie de l'IdO d'ici 2020. Une autre estimation prévoit environ 1 000 appareils par personne d'ici 2025.

1.4 Domaines d'application de l'IdO

1.4.1 Maisons intelligentes

Les maisons intelligentes sont conçues pour améliorer et faciliter la vie de leurs résidents en termes d'indépendance, de sécurité, de sûreté et de confort, ainsi que d'utilisation efficace de l'électricité. L'Internet des objets dans les maisons intelligentes est défini comme la connexion des objets ménagers, en particulier des appareils électriques et électroniques, et des capteurs avec des services web mondiaux, et ils sont interconnectés les uns avec les autres afin de comprendre et d'interagir pour répondre aux exigences et aux besoins du résident. Une maison intelligente qui utilise la technologie IdO peut effectuer plusieurs tâches entre elles [5] :

- **Contrôle à distance de divers appareils électriques :** par exemple, dans le cas où vous quittez la maison et oubliez d'éteindre le four électrique et remarquez qu'après vous être éloigné de la maison, grâce à la maison intelligente basée sur l'Internet des objets, vous pouvez contrôler le four via une application, que ce soit par téléphone ou par ordinateur. Ou lorsque vous oubliez de fermer la porte de garage, le contrôleur intelligent de porte de garage vous enverra une alerte indiquant que la porte a été laissée ouverte, puis vous laissera la fermer en appuyant sur une application.
- **Économiser l'énergie et l'électricité consommées :** en spécifiant les heures nécessaires pour l'allumer ou grâce à des capteurs tels que des ampoules intelligentes, qui s'éteignent automatiquement lorsqu'il n'y a personne dans la pièce, et un thermostat intelligent qui fonctionne également pour régler avec précision la température à l'intérieur de la maison et économiser le coût élevé de l'adaptation.
- **Sécurité :** les dispositifs de sécurité, tels que les caméras intelligentes, peuvent vous dire ce qui se passe à la maison lorsque vous n'êtes pas là. Où la caméra est connectée au réseau Wi-Fi domestique ; Cela permet à la vidéo d'être diffusée à travers elle à l'utilisateur où qu'il se trouve. Il permet également de détecter tout mouvement pour vérifier sur la maison et qu'il n'y a pas de changements majeurs autour d'elle.

1.4.2 Ville intelligente

Les villes intelligentes peuvent être définies comme des infrastructures sociales et techniques complexes, constituées d'entités humaines (différentes parties prenantes et utilisateurs tels que les citoyens, institutions, entreprises, etc.) et appareils numériques (par exemple : capteurs et actionneurs urbains, appareils intelligents pour la maison et le bâtiment intelligent et appareils personnels, tels que les smart phones). L'utilisation de l'Internet des objets dans les villes a inclus plusieurs domaines, à savoir le gouvernement, la vie, les infrastructures, les transports, l'économie, l'industrie et la production d'énergie, l'environnement et les soins de santé [6]. comme indiqué sur le figure 1.1.

- **Gouvernance intelligente :** le gouvernement intelligent est une pratique de gestion de la ville visant à améliorer la prise de décision et à accélérer les procédures bureaucratiques et administratives grâce à une collaboration plus intelligente entre les différentes parties prenantes et acteurs sociaux, y compris les administrations publiques, les employés de la ville, les entreprises privées et les citoyens. Par exemple, l'engagement des citoyens peut s'engager dans les activités de gestion de la ville et les processus de prise de décision via des outils basés sur les TIC et les médias sociaux comme preuve d'un modèle de crowdsourcing mobile, selon lequel les citoyens peuvent être des "utilisateurs de capteurs" avec leurs smartphones et appareils portables.
- **Vie intelligente et infrastructure :** le domaine de la vie intelligente comprend toutes les composantes liées au développement d'une ville plus intelligente, aux infrastructures (telles que les maisons intelligentes, les bâtiments intelligents, etc.),

FIG. 1.1 : La classification des domaines de la ville intelligente avec des composants et des domaines d'application connexes.

ainsi qu'à la gestion et à l'amélioration des services publics, tels que les activités culturelles, le tourisme et l'éducation, qui sont impliqués dans l'amélioration de la qualité de vie générale des citoyens.

- **Mobilité et transport intelligents :** le concept de mobilité et de transport intelligents implique un passage des systèmes de transport traditionnels à la mobilité en tant que service (MaaS), où l'infrastructure IdO intelligente connecte différents acteurs (citoyens, administrations publiques, entreprises privées) et entités (véhicules, appareils personnels, capteurs de la ville, actionneurs, etc). L'IdO est utilisé dans les flux de trafic privés et publics, le routage dynamique du trafic, le stationnement intelligent, le partage de véhicules et la mobilité durable, la conduite connectée, etc.

- **Industrie et production intelligente :** l'industrie intelligente et l'industrie 4.0 définissent la transformation dans laquelle l'Internet des objets, le CPS, les systèmes de communication M2M et les technologies de fabrication basées sur le cloud permettent un environnement de production innovant et moins dépendant de l'homme. En termes d'automatisation des chaînes d'approvisionnement des marchandises, elle peut être facilement tracée du processus de fabrication à la distribution finale. Des informations en temps réel peuvent être collectées et analysées pour suivre les expéditions, ainsi que pour évaluer la qualité et la convivialité des produits.

- **Énergie intelligente :** les systèmes énergétiques intelligents incluent l'intégration intelligente des énergies renouvelables décentralisées Les sources d'énergie durables et leur distribution efficace visent à améliorer la consommation d'énergie et à mieux gérer la production et la distribution d'énergie, ainsi que son exploitation, par exemple grâce à l'utilisation de modèles de prévision (développés à partir des données de consommation collectées). De nombreux capteurs IdO sont utilisés dans

le domaine de l'énergie, tels que les résistances dépendant de la lumière (LDRs), consomment du rayonnement solaire et de l'électricité.

- **Environnement intelligent :** ce domaine comprend la collecte et l'analyse de données pour la réduction de la pollution, la qualité de l'eau, la surveillance de l'approvisionnement en eau et la gestion des événements météorologiques et climatiques. La gestion intelligente des déchets est également incluse en raison de son impact environnemental. À Singapoura, les déchets solides sont gérés par le biais de l'installation de gestion intégrée des déchets (IWMF), qui comprend un Internet des objets innovant qui augmente l'efficacité des processus et réduit les émissions de gaz à effet de serre.

- **Soins de santé :** Cela comprend la surveillance à distance, les conseils médicaux à distance et les effets indésirables des médicaments, où les patients sont surveillés au moyen d'appareils portables. L'Internet des objets est également utilisé dans les hôpitaux pour servir les employés et les patients, et la Chine est l'un des premiers hôpitaux intelligents à l'appliquer, où tous les aspects des patients sont gérés via l'Internet des objets et les soins de santé connectés, et il fournit également des services du bâtiment et la gestion durable de l'énergie.

1.4.3 Bâtiments intelligents

Au cours des dernières années, l'Internet des objets a commencé à s'infiltrer dans les bâtiments de l'industrie. Les chercheurs et les praticiens explorent les avantages et les inconvénients de l'Internet des objets par le biais d'applications réelles. Par exemple, de nombreuses entreprises, dont IBM et Intel, lancent déjà leurs produits des bâtiments intelligentes dans le monde. Par exemple concernant l'utilisation de l'IdO dans les bâtiments intelligents, avant d'allumer le climatiseur, qui est une procédure qui se déclenche en raison du changement de température, toutes les fenêtres doivent également être fermées automatiquement. En conséquence, le capteur doit faire fonctionner la climatisation, puis les informations doivent être échangées entre la climatisation et les fenêtres, et pour ce faire, les bâtiments intelligents doivent contenir des capteurs, des actionneurs, des dispositifs et des systèmes de contrôle connectés les uns aux autres pour améliorer le service. Les bâtiments doivent aussi disposer d'un réseau de communication comprenant des éléments de bâtiment afin qu'ils puissent être contrôlés ou surveillés à distance. Voici quelques-unes des choses qui peuvent bénéficier de l'utilisation de l'Internet des objets[7].

- **Localisation des occupants et suivi des ressources :** la localisation des espaces intérieurs est d'une grande valeur pour améliorer les performances du bâtiment car elle peut être fournie avec une navigation vers la destination pour les nouvelles personnes qui ne connaissent pas le bâtiment, elle permet aux gestionnaires de localiser plus facilement tout équipement ou installation qui a besoin maintenance, De même, localiser les personnes permettra d'améliorer les services et d'économiser de l'argent en termes d'éclairage et de chauffage (lorsque le lieu est vide, les lumières peuvent être éteintes pour économiser et arrêter le chauffage).

- **Gestion de l'énergie :** les bâtiments représentent 40% de la consommation d'énergie dans le monde, il est donc très important d'y consommer de l'énergie de manière rationnelle et économique, mais à condition qu'elle ne diminue pas le niveau de service. Il existe en fait des systèmes qui aident à surveiller la consommation d'énergie et à l'améliorer, par exemple en ajustant le chauffage et la ventilation en fonction du nombre de personnes dans la zone et également en fonction de la température dans la zone que vous mesurez à l'aide de capteurs.

- **Améliorer le confort intérieur :** le confort des personnes dans le bâtiment est important pour augmenter leur bien-être et leur productivité, d'autant plus que selon les statistiques elles passent 80% de leur temps à l'intérieur des bâtiments et cela peut se faire en ajustant la température en considérant les personnes comme des capteurs qui utilisent des applications pour envoyer des plaintes pour l'analyser dans le but de l'améliorer à la température optimale qui satisfait le plus grand nombre.

1.4.4 Soins de santé

L'utilisation de l'Internet des objets en termes de soins de santé contribue à la recherche, à la pratique clinique et à la gestion des patients et repose sur quatre principes : le premier est de collecter des informations via des capteurs, des écrans et des caméras, et le second est de transférer des données (Les capteurs et autres dispositifs doivent donner des informations numériques qui peuvent être traitées). Le troisième est le stockage des données et le quatrième est le traitement et l'analyse des données afin d'obtenir des informations pour aider à la prise de décision. La figure 1.2 présente le concept de l'Internet des objets dans le domaine de la santé.

L'Internet des objets peut être utilisé dans les soins contre le cancer. En 2018, ils ont mené un essai clinique dans lequel des appareils portables étaient surveillés pour les patients, avec une application de suivi des symptômes qui envoie des mises à jour régulières et d'urgence aux médecins. Enfin, les résultats de l'étude ont montré que les patients qui utilisaient la technologie IdO présentaient des symptômes plus légers, par rapport au groupe qui était physiquement évalué sur une base hebdomadaire. Il peut également être utilisé pour surveiller les patients atteints de diabète et d'asthme. Il peut également être utilisé par des personnes ordinaires pour surveiller leur consommation alimentaire, leur sommeil et leurs signes vitaux [8].

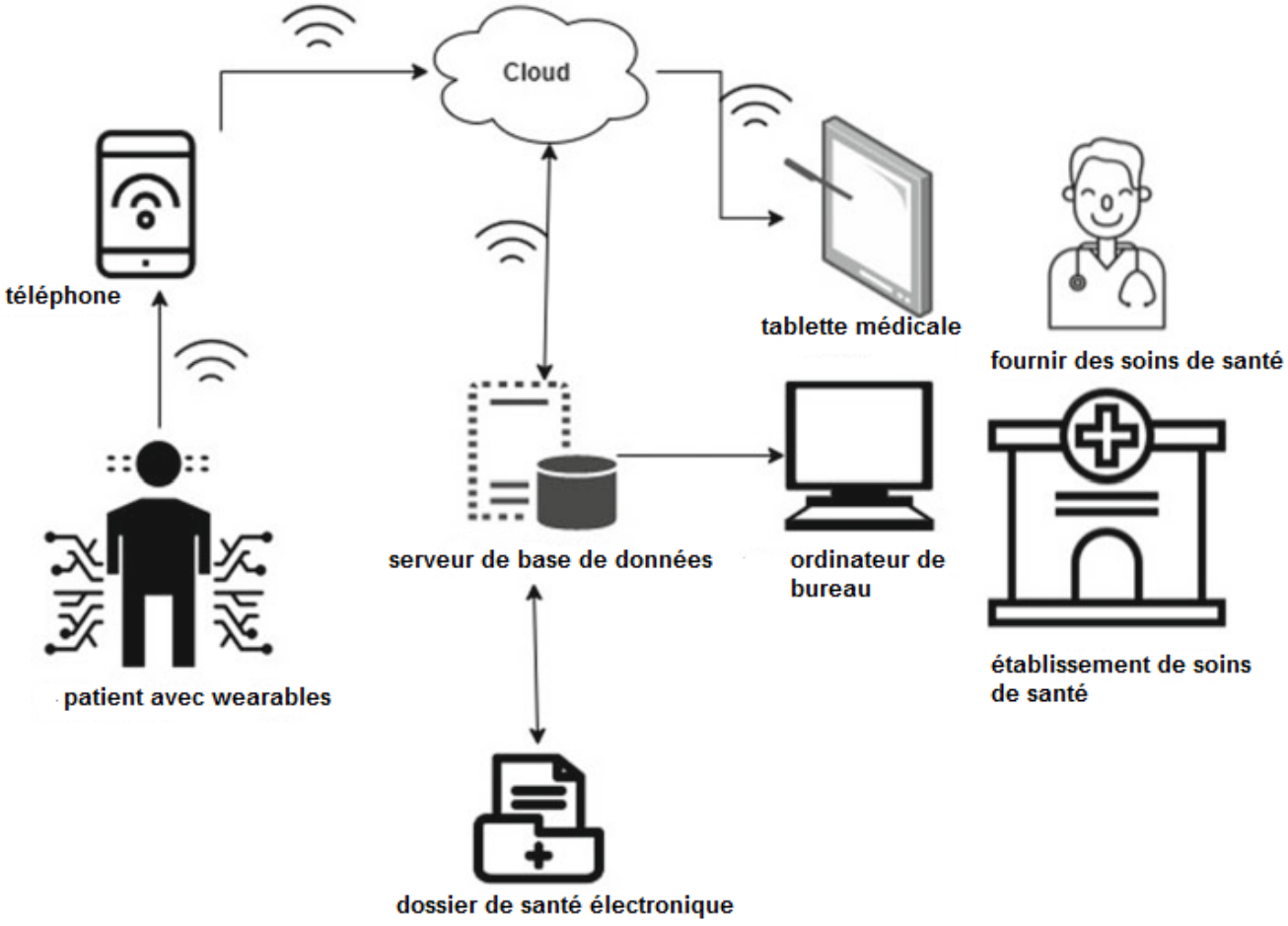


FIG. 1.2 : le concept de l'Internet des objets dans le domaine de la santé.

1.4.5 Agriculture

L'application de l'Internet des objets pour l'agriculture intelligente est très importante car elle améliorera la productivité et fournira Des aliments propres et verts, soutiennent la traçabilité des aliments, réduisent le travail humain et améliorent l'efficacité de la production. Les principales applications de l'Internet des objets pour l'agriculture sont présentées dans la figure suivante 1.3 [9].

FIG. 1.3 : Une illustration des applications IdO pour l'agriculture intelligente.

Les principaux avantages de l'IdO dans l'agriculture sont brièvement décrits ci-dessous :

- Agriculture communautaire dans les zones urbaines et rurales tirant parti du matériel et ressources logicielles et de grandes quantités de données.
- Traçabilité logistique et qualitative de la production alimentaire qui permet de réduire les coûts et le gaspillage d'intrants dû à l'utilisation de données en temps réel pour la prise de décision.
- Génération de modèles économiques dans le contexte agricole permettant d'établir une relation directe avec le consommateur.
- Surveillance des cultures qui permet de réduire les coûts ainsi que le vol de machinerie.
- Des systèmes d'irrigation automatiques qui fonctionnent en fonction de la température, de l'humidité, et les valeurs d'humidité du sol obtenues grâce à des capteurs.
- Collecte automatique des paramètres environnementaux via des réseaux de capteurs pour traitement et analyse ultérieurs.
- Systèmes d'aide à la décision qui analysent de grandes quantités de données pour améliorer efficacité et productivité.

1.5 Les avantages de l'IdO

- **La communication :** L'IdO encourage la communication entre les appareils. De ce fait, les appareils physiques sont capables de rester connectés et donc la transparence totale est disponible avec moins d'inefficacités et une meilleure qualité.

- **Automatisation et contrôle :** en raison des objets physiques connectés et contrôlés numériquement et de manière centralisée avec une infrastructure sans fil, il y a une grande quantité d'automatisation et de contrôle dans le fonctionnement.

- **Information :** il est évident qu'avoir plus d'informations aide à prendre de meilleures décisions.

- **Gagnez du temps :** comme cela réduit l'effort humain, cela fait définitivement gagner du temps. Le temps est le principal facteur qui peut économiser grâce à la plate-forme l'IdO.

- **Minimiser l'effort humain :** comme les appareils de l'IdO interagissent et communiquent entre eux et effectuent beaucoup de tâches pour nous, ils minimisent l'effort humain.

- **l'argent :** le plus grand avantage de l'IdO est d'économiser de l'argent. Si le prix de l'équipement de marquage et de surveillance est inférieur au montant d'argent économisé, alors l'Internet des objets sera très largement adopté. L'IdO s'avère fondamentalement très utile pour les gens dans leurs routines quotidiennes en permettant aux appareils de communiquer entre eux de manière efficace, économisant ainsi de l'énergie et des coûts [10].

1.6 Les difficultés associés à l' IdO

- **Compatibilité :** actuellement, il n'existe aucune norme internationale de compatibilité pour l'équipement de marquage et de surveillance. Nous croyons que cet inconvénient est le plus facile à surmonter. Les fabricants de ces équipements n'ont qu'à se mettre d'accord sur une norme, telle que Bluetooth, USB, etc. Il n'y a rien de nouveau ou d'innovant nécessaire.

- **Complexité :** la conception, le développement, la maintenance et l'activation de la grande technologie du système IdO sont assez compliqués.

- **Sécurité :** comme les systèmes IdO sont interconnectés et communiqués sur des réseaux. Le système offre peu de contrôle malgré toutes les mesures de sécurité, et il peut être à l'origine de divers types d'attaques réseau.

- **Confidentialité :** même sans la participation active de l'utilisateur, le système IdO fournit des données personnelles substantielles avec un maximum de détails. [10]

1.7 L'architecture de l'internet des objets

L'architecture IdO se compose de trois couches **1)** la couche perception/détection **2)** la couche réseau et **3)** la couche application. Dans d'autres œuvres, il se compose de cinq couches : la couche perception/sensation ; couche transport/réseau ; middleware/couche de traitement ; Couche applicative et couche métier. Comme le montre la figure 1.4.

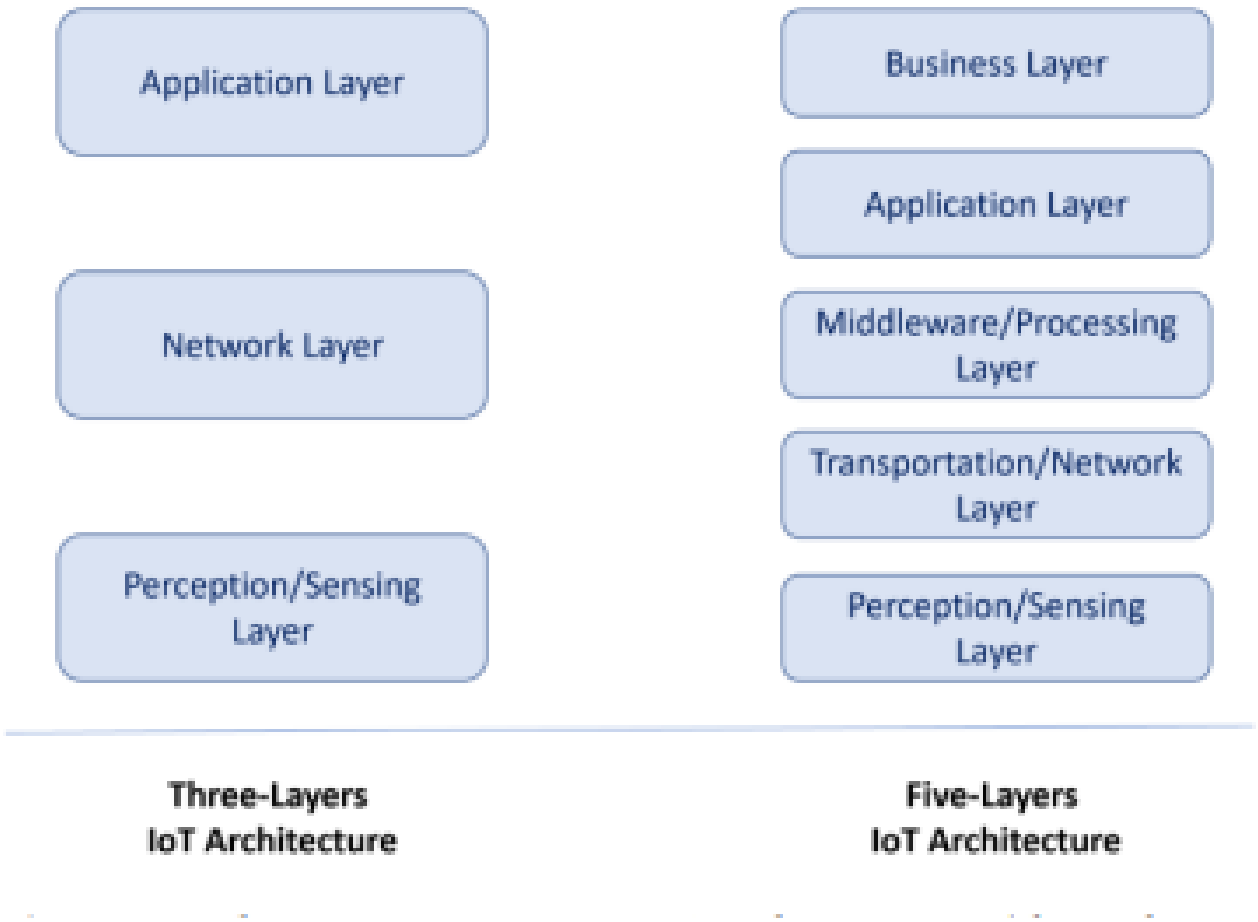


FIG. 1.4 : Les architectures de l'IdO les plus courantes (trois couches et cinq couches).

Dans ce qui suit, nous donne un description de L'architecture à cinq couches car il est considéré comme la version le plus détaillée de l'architecture à trois couches [11].

1. **Couche de perception/détection (Perception Layer) :** La couche de perception ou de détection est liée au niveau physique dont elle est composée les dispositifs, tels que les capteurs et les actionneurs, qui interagissent entre eux et avec le monde physique. Elle collecte des données spécifiques basées sur l'appareil. Il existe de nombreux types d'appareils qui mesurent divers changements tels qu'un capteur pour mesurer la température, l'humidité, la pression, la distance, la vitesse, la tension, le poids, le niveau de luminance, etc. Ces données sont ensuite transmises via le réseau.
2. **couche transport/réseau (Network layer) :** La couche de transport fonctionne pour transférer les données des capteurs et les diriger vers des couches supérieures

et vice versa, pour de transférer ces données, il existe différents protocoles, tels que : Bluetooth, communication en champ proche (NFC), Wi-Fi,3G, 4G.etc.

3. **Couche middleware/ traitement :**

   Cette couche remplit des nombreuses fonctions dont la collecte de données à partir des appareils qui utilisent des protocoles différents, par conséquent la couche middleware doit permettre l'interopérabilité entre les appareils connectés. Cette couche stocke également les différentes données ainsi que leur extraction et règlement.

4. **couche application :**

   La couche d'application fournit les formats de sortie avec lesquels l'utilisateur interagit. C'est ce qui est chargé de fournir des services spécifiques à l'application et à l'utilisateur.

5. **couche métier (Business Layer) :**

   Cette couche classe tous les processus et outils frontaux qui consomment les données de la couche application pour produire des services avancés pour l'analyse et la visualisation de données volumineuses, en créant des diagrammes manuels, des modèles commerciaux, des organigrammes, des rapports exécutifs, etc. pour aidera les responsables fonctionnels ou dirigeants à prendre des décisions plus précises sur les stratégies et les feuilles de route.

1.8 Comment fonctionne l'Internet des objets ?

Il y a plusieurs étapes pour faire fonctionner l'Internet des objets (IdO).

- Chaque « objet » sur l'Internet des objets doit avoir une identité unique. Grâce à l'adresse Internet Protocol Version 6 (IPv6).

- Chaque « chose » doit pouvoir communiquer. Il existe un certain nombre de modernes les technologies sans fil qui rendent communication possibles, telles que Wifi, Bluetooth etc.

- Chaque « chose » doit avoir des capteurs afin que nous puissions obtenir des informations à son sujet. Les capteurs peuvent être des capteurs de température, d'humidité, de lumière, de mouvement, de pression, d'infrarouge, d'ultrasons, etc.

- Chaque "chose" doit avoir un microcontrôleur (ou microprocesseur) pour gérer les capteurs et les communications, et d'effectuer les tâches.

- Enfin, nous aurons besoin de services cloud pour stocker, analyser et afficher les données afin de pouvoir voyez ce qui se passe et agissez via des applications téléphoniques.

1.9 Modélisation des applications IdO dans le plus haut niveau d'abstraction :

La modélisation dans le plus haut niveau d'abstraction permet l'abstraction des plateformes hétérogènes et des dispositifs IdO pour modéliser l'architecture du système Ido souhaité. L'Internet des Objets peut être modélisé en plusieurs modèles, mais nous en aborderons quatre , à savoir : **SYSML** (System Modeling Language), **UML4IOT** (Unified Modeling Langage For the Internet of Things), **TheThingML** (Internet of Things Modeling Language), **BPMN** (Business Process Modeling Notation).

1.9.1 Notation de modélisation

1.9.1.1 SYSML (System Modeling Language)

SysML est un langage couramment utilisé pour la conception des systèmes basés sur des modèles (MBSD). Il s'agit d'une norme OMG (Object Management Group) qui prend en charge la spécification, l'analyse, la conception, la vérification et la validation d'une large gamme de systèmes et de systèmes de systèmes. Il fournit des diagrammes discrets pour décrire la structure et les composants du système, pour explorer les politiques d'allocation cruciales pour la conception du système et pour identifier les exigences de conception. Il est largement appliqué pour l'ingénierie des systèmes de systèmes (SoS).
SysML est un langage à usage général, facilitant la modélisation de tout système ou système de systèmes. Les profils UML peuvent être utilisés pour restreindre ou étendre les fonctionnalités SysML afin de servir un domaine spécifique, comme par exemple les systèmes en temps réel et embarqués ou les systèmes d'information. Ces profils, accompagnés de plugins spécifiques, peuvent être exécutés dans des outils de modélisation UML (tels que Magic Draw ou IBM Rational Modeler) et sont capables de produire des modèles de système valides pour le domaine spécifique, sur la base des spécifications du profil. la modélisation d'applications IdO avec sysml4iot est fait a travers d'un profil SysML nommé SysML4IoT. l'utilisation de profils aide les parties prenantes à gérer la complexité du système, augmente la compréhension et communique le modèle de système de manière non ambiguë (ou aussi non ambiguë que possible). De plus, les profils favorisent la réutilisation et l'interopérabilité entre les composants logiciels et les systèmes.[12]

1.9.1.2 TheThingML (Internet of Things Modeling Language)

L'approche ThingML comprend un langage de modélisation, un ensemble d'outils et une méthodologie. Le langage (ThingML) combine des constructions de modélisation logicielle éprouvées alignées sur UML (états-transitions et composants), un langage d'action impératif indépendant de la plate-forme et des constructions ciblées sur les applications IdO. Les outils comprennent des éditeurs, des transformations (par exemple, l'exportation vers UML) et une infrastructure de génération de code multiplateforme avancée qui prend en charge plusieurs langages de programmation cibles.
Le premier objectif de l'approche ThingML est de permettre de faire abstraction des plates-formes hétérogènes et des appareils IdO pour modéliser l'architecture du système

IdO souhaité. L'approche ThingML intègre également trois fonctionnalités qui facilitent le développement d'applications IdO :

- Le premier est un mécanisme permettant d'encapsuler rapidement les bibliothèques existantes en mélangeant le code ThingML avec le code cible.
- Le second est le traitement d'événements complexes (CEP) pour gérer des modèles réactifs complexes.
- Le troisième est des sessions dynamiques pour gérer les capteurs découverts dynamiquement.[13]

1.9.1.3 BPMN (Business Process Modeling Notation)

Le groupe de gestion d'objets (OMG) a développé un modèle et une notation de processus métier standard (BPMN). L'objectif principal de BPMN est de fournir une notation facilement compréhensible par tous les utilisateurs métier, des analystes métier qui créent les premières ébauches des processus aux développeurs techniques responsables de la mise en œuvre de la technologie qui exécutera ces processus. Ainsi, BPMN crée un pont standardisé pour l'écart entre la conception des processus métier et la mise en œuvre des processus. La nouvelle version de la norme BPMN 2.0 est extensible, ce qui nous permet d'exprimer des modèles de processus plus étendus en intégrant les appareils intelligents d'Internet[14].

1.9.1.4 UML4IoT

l'UML est l'une des ressources de modélisation visuelle qui rend possible l'utilisation d'extensions pour représenter de tels systèmes. Un système IdO peut être représenté par deux types de langages : textuels et visuels. Les auteurs proposent l'utilisation des ressources UML comme langage visuel, pour représenter les différentes parties d'un système IdO.

Thramboulidis et Christoulakis étudient le développement d'UML4IoT, qui intègre CPS et l'IdO. Ils décrivent un cadre pour orienter les défis introduits par l'utilisation de l'IdO dans le processus de développement de produits. UML4IoT présente deux manières de modéliser l'interface d'objets intelligents simples : **(1)** Utilisation de la Diagramme de classes UML et extensions pour spécifier une partie du système et **(2)**utiliser le code source en Java, si les projets de haut niveau ne sont pas suffisamment représentés par des ressources UML. L'UML4IoT est orienté objet (OO) et utilise les diagrammes de classes avec des extensions, formant un profil pour un domaine particulier[15].

1.9.2 Modèle conceptuel

La figure 1.5 fournit un modèle conceptuel incluant les relations entre les concepts de base de l'IdO. Le modèle a été adopté à partir du modèle de domaine AIOTI (Alliance of IoT Inno-vation). Le modèle de domaine représente les concepts de base et les relations dans le domaine au plus haut niveau[16].

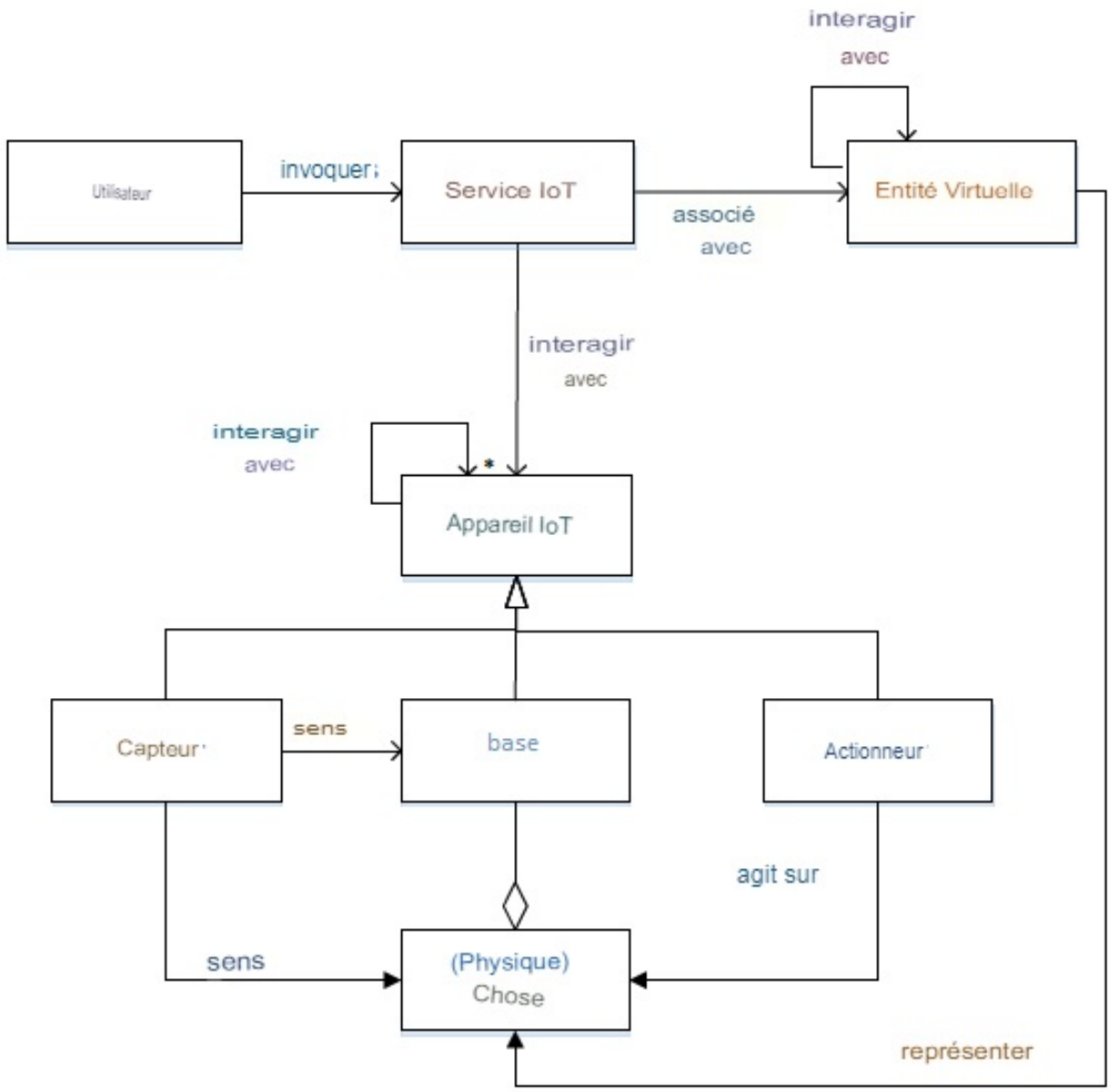


FIG. 1.5 : Modèle conceptuel pour l'IdO.

- **Utilisateur (user) :** Dans le modèle, l'utilisateur interagit avec une entité physique du monde physique. l'utilisateur peut être une personne humaine ou un agent logiciel qui a un objectif, pour l'achèvement duquel l'interaction avec l'environnement physique doit être effectuée par l'intermédiaire de l'IdO.

- **Objet (thing) :** Un objet est une partie discrète et identifiable de l'environnement physique qui peut intéresser l'utilisateur pour la réalisation de son objectif. Les choses peuvent être n'importe quelle entité physique telle que des humains, des voitures, des animaux ou des ordinateurs. L'interaction entre un utilisateur et une chose est médiatisée par un service IdO qui est associé à une entité virtuelle.

- **Entité virtuelle (virtual entity) :** Une entité virtuelle, une représentation numérique de l'entité physique. Une chose peut être représentée dans le monde numérique

par une entité virtuelle. Différents types de représentations numériques des Choses peuvent être utilisées telles que des objets, des modèles 3D, des avatars, des objets ou encore un compte de réseau social. Certaines entités virtuelles peuvent également interagir avec d'autres entités virtuelles pour atteindre leur objectif.

- **IdO appareil(Iot device :)** Un aspect important de l'IdO est que les modifications des propriétés d'une chose et de son entité virtuelle correspondante doivent être synchronisées. Ceci est généralement réalisé par un dispositif IdO qui est intégré, attaché ou simplement placé à proximité de la chose. En principe, nous pouvons identifier trois dispositifs dont les capteurs, les balises et les actionneurs. Les capteurs sont utilisés pour mesurer l'état des choses qu'ils surveillent. Essentiellement, les capteurs prennent un signal mécanique, optique, magnétique ou thermique et le convertissent en tension et en courant. Ces données fournies peuvent ensuite être traitées et utilisées pour définir l'action requise. Les étiquettes sont des dispositifs qui prennent en charge le processus d'identification en utilisant généralement des capteurs spécialisés appelés lecteurs. Le processus d'identification peut être différent, y compris optique comme dans le cas des codes à barres et QRcode, ou basé sur RF. Les actionneurs sont employés pour changer ou affecter les choses.

1.10 Conclusion

Ces dernières années, l'Internet des objets est devenu l'une des technologies les plus importantes du XXIe siècle. L'objectif de ce chapitre est de définir ce nouveau terme " Internet des Objets" et ses domaines d'application. Puis son histoire, ses caractéristiques, ses avantages et ses inconvénients, et enfin la modélisation au plus haut niveau d'abstraction et de modélisation conceptuelle.

Chapitre 2

BPMN (Business Process Modeling Notation)

2.1 Introduction

Un processus d'entreprise est considéré comme une suite d'étapes. De nos jours, beaucoup de sociétés esquissent des représentations graphiques pour leur processus d'entreprise grâce a des techniques de dessin. le BPMN (Business Process Modeling Notation) est une norme pour la représentation graphique des processus métier qui proposant un ensemble des éléments graphiques normalisés et une grammaire permettant leur manipulation.

Dans ce chapitre, nous verrons ce que signifie un processus métier, qu'est-ce que BPMN et quels sont ses différents éléments et diagrammes qui permettent de modélise les processus métier en général et les applications IdO en particulier.

2.2 Processus métier

2.2.1 Définition de Processus

Un processus il se définit comme « une suite d'événements naturels, se déroulant dans le même ordre », il se définit dans le domaine technique comme étant « une suite d'opérations aboutissant à un résultat » notamment dans l'expression processus de fabrication[17].

2.2.2 Définition de Processus métier

Un processus métier ou "Business Process" en anglais est défini par le Workflow Management Coalition (WfMC) comme : "un ensemble de procédures ou d'activités liées les unes aux autres pour atteindre collectivement un objectif métier en définissant les rôles et les interactions fonctionnelles au sein d'une structure organisationnelle" .En termes simples, il est défini comme une série d'activités qui conduisent à une production métier. Il s'agit d'un ensemble de tâches organisées qui, une fois terminées, permettront d'atteindre un objectif organisationnel et un résultat spécifique.

Il peut également être défini comme un ensemble de relations logiques entre un ensemble de tâches, y compris les interactions avec les participants (il peut s'agir acteurs humains ou processus métiers). Les processus métiers sont au coeur des organisations et se déclinent dans tous les domaines, que ce soit dans le management, dans la gestion financière ou encore dans la gestion de l'excellence opérationnelle[18].

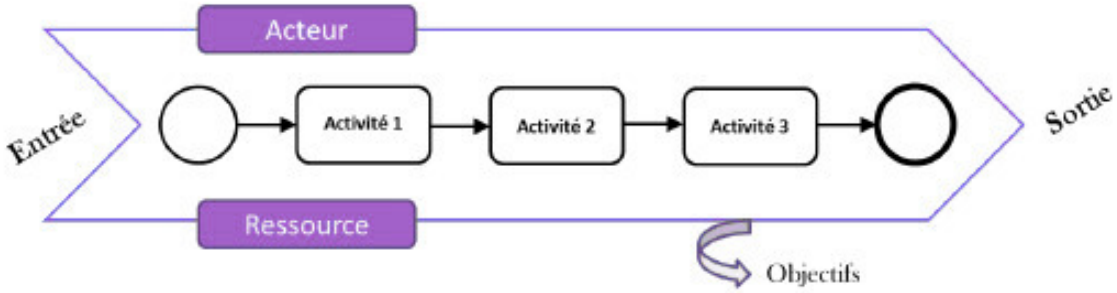


FIG. 2.1 : Un exemple illustre comment fonction un processus métier.

2.3 Gestion des processus métier (BPM)

2.3.1 Définition Gestion des processus métier (BPM)

Il existe des nombreuses définitions du BPM(business process management BPM en anglais) y compris la définition de Van der Aalst : « Soutenir les processus métier à l'aide des méthodes, des techniques et des logiciels pour concevoir, mettre en œuvre, contrôler et analyser des processus opérationnels impliquant des humains, des organisations, des applications, des documents et d'autres sources d'informations »[19].

Son but est d'apporter des gains significatifs en termes d'efficacité et de productivité. Les objectifs de la BPM sont [20] :

- Automatiser les processus métier de l'entreprise ;
- Éliminer où faire ce peut les délais d'attentes à l'intérieur de ces processus ;
- Accélérer grandement les étapes de décisions en apportant en temps réel les informations pertinentes aux bonnes personnes ;

2.3.2 Cycle de vie d'un processus métier

Le cycle de vie du processus métier selon l'approche BPM va de la théorie à la pratique en passant par plusieurs étapes, il permet d'accompagner le processus métier de sa conception à sa gestion et son pilotage avec un développement continu en fonction des objectifs métiers.

Il n'y a pas de présentation standardisée du nombre de phases du cycle de vie du BPM. Et ça varie selon la précision choisie pour les étapes. Il se compose principalement de quatre étapes comme indiqué sur la figure 2.2[21].

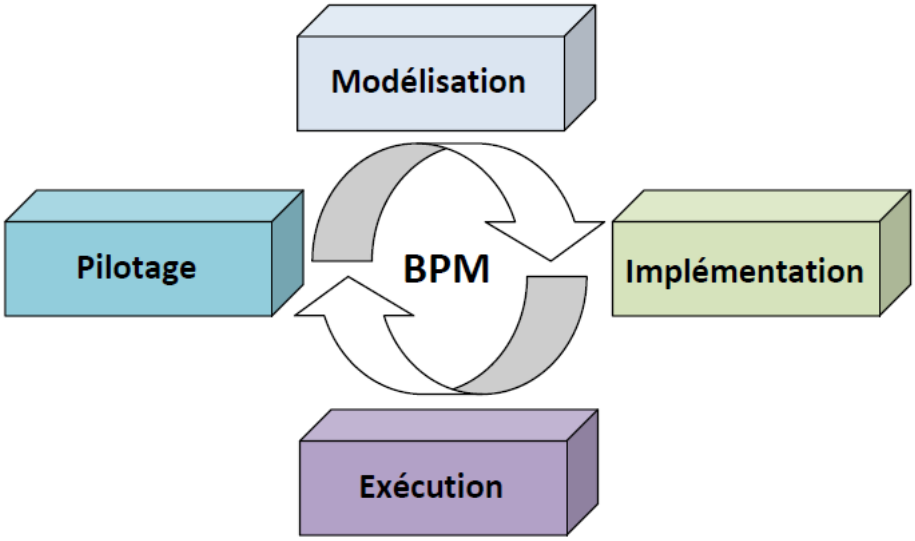


Fig. 2.2 : Le cycle de vie du BPM

- **Phase de modélisation :** a cette étape, les experts métiers définissent, de manière abstraite ou détaillée, les processus métiers ou redéfinissent le processus existant afin de les améliorer à l'aide de l'outil de modélisation.Les modèles de processus créés à cette étape sont généralement à un haut niveau d'abstraction, Ensuite, il est transformé en un modèle de processus exécutable, qui fait l'objet de l'étape suivante.

- **Phase d'implémentation :** le processus créé dans la phase de modélisation est transformé et enrichi par les ingénieurs IT dans le but d'être exécuté par un moteur de processus "Process Engine".

- **Phase d'exécution :** C'est la phase de mise en œuvre du BPM. En effet, durant cette phase, le processus exécutable est interprété par un moteur d'exécution appelé BPE "Business Process Engine".

- **Phase d'analyse et d'optimisation :** cet phase sert à superviser l'exécution opérationnelle des processus métier et mesurer les performances en se basant sur des fichiers logs. Le BPM a besoin des outils de pilotage permettant ainsi une prise de décision concernant l'efficacité et l'amélioration des processus. Les solutions de BPM nomment cette fonctionnalité BAM (Business Activity Monitoring) ou Supervision de l'activité métier, en français.

2.4 BPMN 2.0

2.4.1 Le standard BPMN

Afin de documenter les processus métiers, une manière appropriée est d'utiliser une notation, souvent graphique, pour les modéliser. Une notation graphique adaptée est un moyen de casser cette frontière par un langage compréhensible par tous les maillons de la

chaîne. ce qu'on appelle BPMN acronyme pour « Business Process Modeling Notation ». BPMN est une norme pour les processus de métier modélisation qui fournit une notation graphique pour spécifier les processus métier dans un Diagramme de processus métier (BPD), basé sur les techniques traditionnelles d'organigramme[22].

L'objectif principal de BPMN est de fournir une notation qui est compréhensible par chaque intervenant du projet informatique : de l'analyste métier qui rédige la version initiale du processus, au client qui validera le processus, au développeur technique qui est responsable de l'implémentation technologique du logiciel qui exécutera ces processus puis finalement aux manageurs qui géreront et contrôleront l'efficacité du processus[23].

2.4.2 L'historique de BPMN

Il a été initialement développé par la Business Process Management Initiative (BPMI), en 2001 le marché de la modélisation des processus était fragmenté avec de nombreuses notations et points de vue différents sur la modélisation. De là est venue l'idée de normaliser les techniques orientées métier pour représenter visuellement les composants de processus et d'aligner la notation avec un langage de processus exécutable.

En mai 2004: BPMN 1.0 a été rendue publique l'objectif principal de la spécification BPMN était de fournir une notation facilement compréhensible par tous les utilisateurs professionnels. BPMN 1.0 était également pris en charge avec un modèle interne mappé sur l'exécutable BPEL4WS.

Le 6 février 2006: BPMI a été subsumé par l'OMG, qui a depuis maintenu et développé la norme BPMN.

En janvier 2008: La version BPMN 1.1 a été publiée **et un an plus tard**, la version 1.2 a été publiée.

En janvier 2011: la version 2.0 a été publiée,Cette version incluait de nombreuses modifications, notamment :

- L'ajout d'un diagramme de chorégraphie et d'un diagramme de conversation.
- Evénements sans interruption pour un processus.
- Sous-processus d'événement pour un processus.

et modifications techniques comme :

- Une définition de la sémantique d'exécution des processus.
- Un métamodèle formel.
- Formats d'échange pour l'échange de modèles de syntaxe abstraite dans XML Metadata Interchange (XMI) et XML Schema Definition (XSD).

En décembre 2013: BPMN 2.0.2 a été publié , n'incluait que des modifications mineures dans termes de corrections de fautes de frappe et une modification dans une section [24].

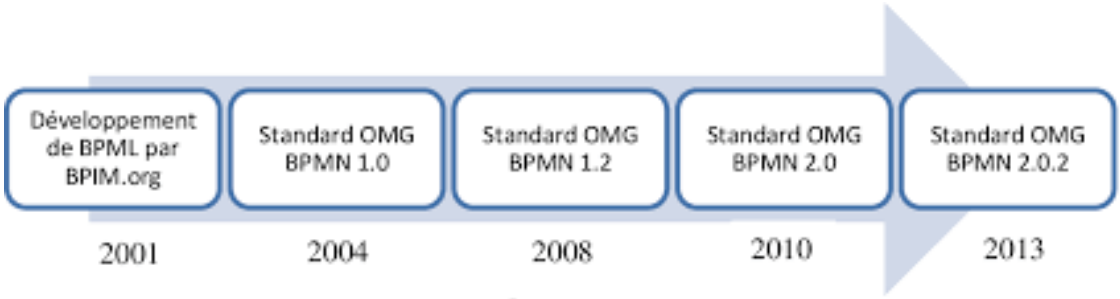


Fig. 2.3 : Historique de l'évolution du standard BPMN
[25]

2.4.3 Notation de BPMN 2.0

BPMN 2.0 contient ces cinq types d'éléments[14] :

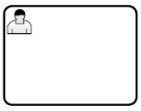
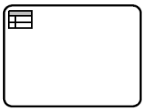
- Les objets de flux (Flow Object) ;
- Les données(data) ;
- Les objets de connexion (Connecting Object) ;
- Les partitions(Swimlanes) ;
- Les artéfacts (Artifacts).

1. **Les objets de flux**

   Les objets de flux sont les principaux éléments graphiques permettant de définir le comportement d'un processus métier. Il existe trois objets de flux « les activités », « les événements » et « les passerelle » :

   (1) **Activities :** une activité est un travail effectuée dans le cadre d'un processus métier. Il est représenté par un rectangle aux coins arrondis. Qui correspondent à une action qui peut être réalisée par un humain ou une machine. Il peut être atomique ou non atomique (composé). Ils peuvent être classés en trois types : tâche, sous-processus et activité d'appel.

   - **Tâche :** une tâche est une activité atomique incluse dans un processus. Une tâche est utilisée lorsque le travail dans le processus ne peut pas être décomposé à un niveau de détail plus fin. Un objet tâche partage la même forme que le sous-processus, qui est un rectangle aux coins arrondis.il existe sept types de tâches comme indiqué dans le tableau 4.12 :

Élément	Description	Notation
Tâche service	Une tâche service est une tâche qui utilise une sorte de service, qui peut être un service Web ou une application automatisée.	
Tâche d'envoi de message	Une tâche d'envoi est une tâche pour envoyer un message à un participant externe au processus.	
Tâche de réception d'un message	Une tâche de réception est une tâche pour attendre l'arrivée d'un message d'un participant externe au processus. Une fois le message reçu, la tâche est terminée.	
Tâche utilisateur	Une tâche utilisateur est une tâche typique de "flux de travail" dans laquelle un interprète humain exécute la tâche avec l'aide d'une application logicielle. Simplement elle est une tâche réalisée par un humain avec une aide logicielle.	
Tâche manuelle	Une tâche manuelle est une tâche qui doit être exécutée sans l'aide d'un moteur d'exécution de processus métier ou d'une application(une tache réalisée par humain Seulement).	
Tâche de règle métier	Une tâche de règle métier fournit un mécanisme permettant au processus de fournir une entrée à un moteur de règles métier et d'obtenir la sortie des calculs que le moteur de règles métier peut fournir.	

Tâche de script	La tâche de script est exécutée par un moteur de processus métier. Le modélisateur définit un script dans un langage que le moteur peut interpréter. Lorsque la tâche est prête à démarrer, le moteur exécute le script. Lorsque le script est terminé, la tâche sera également terminée.	

TAB. 2.2 : Les types des tâches dans BPMN 2.0.

- **Sous-processus :** est une activité composée dont les détails internes ont été modélisés à l'aide d'activités, de passerelles, d'événements et de flux de séquence. Il est un objet graphique dans un processus, mais il peut également être "ouvert" pour afficher un processus de niveau inférieur, elle est représentée par une tâche avec un petit + permettant d'accéder au détail (sous-processus réduit) Ou représenté dans sa version élargie, comme le montre l'imagee, comme le montre l'image 2.4.

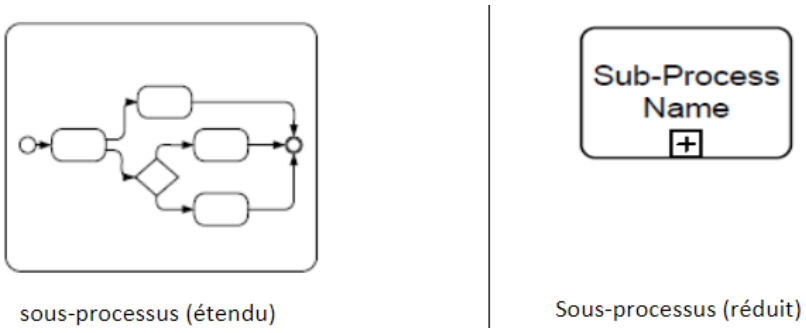


FIG. 2.4 : Les formes de sous-processous. sous$_p ros$

Il existe cinq types de sous-processus, comme le montre l'image 2.5.

- **loop :** déterminer un processus répétitif.
- **Multi-instance :** déterminer un processus répétitif dont les instances s'exécutent en parallèle.
- **Compensation :** représente un sous-processus de compensation en réaction à une activité compensée au sein d'une transaction ayant échoué.
- **Ad hoc :** représente un sous-processus dont les tâches internes sont exécutées dans un ordre indéterminé.
- **Compensation et Ad-Hocc :** représente un sous-processus de rémunération dont les tâches internes sont exécutées dans un ordre indéterminé.

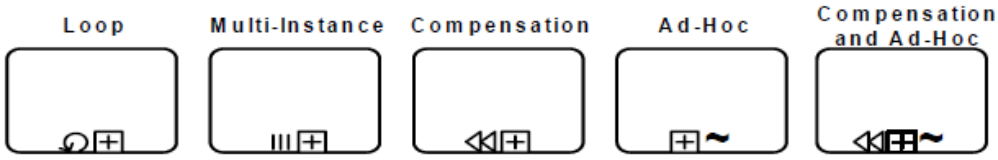


FIG. 2.5 : Les types de sous-processous.

- **Activité d'appel :** l'activité d'appel (ou sous-processus réutilisable) vous permet d'appeler et d'invoquer un autre processus dans le cadre de ce processus. Il est similaire à un sous-processus intégré, mais le processus est externalisé (c'est-à-dire stocké sous forme de BPMN séparé) et peut être invoqué par différents processus. Ses formes sont montrées dans la figur 2.6.

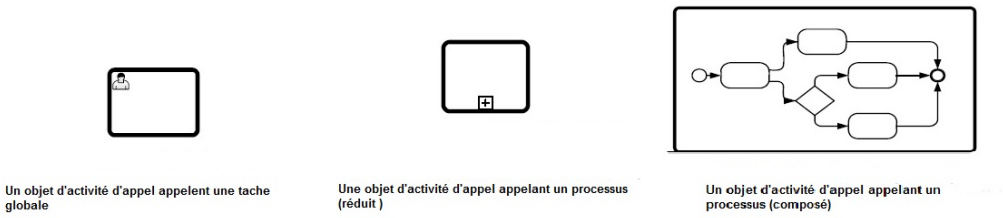


FIG. 2.6 : les formes de Activité d'appel .

(2) **Evénements :** un événement est quelque chose qui "se produit" au cours d'un processus. Ces événements affectent le flux du modèle et ont généralement une cause (Trigger) ou un impact (Result). Il existe trois types d'événements, en fonction du moment où ils affectent le flux : début, intermédiaire et fin.Comme le montre la figure suivant :

FIG. 2.7 : Événements dans BPMN 2.0.

- **Les événements de début** : indique où un processus particulier commencera. En termes de flux de séquence, l'événement de démarrage démarre le flux du processus et, par conséquent, n'aura aucun flux de séquence entrant - aucun flux de séquence ne peut se connecter à un événement de démarrage.

- **Les événements intermédiaires** : indique où quelque chose se passe (un événement) quelque part entre le début et fin d'un processus. Cela affectera le flux du processus, mais ne démarrera pas ou ne terminera pas (directement) le processus.
- **l'événement de fin** : indique où un processus se terminera. En termes de flux de séquence, la fin l'événement met fin au flux du processus et, par conséquent, n'aura aucun flux de séquence sortant - aucun flux de séquence ne peut se connecter à partir d'un événement de fin.

Quelque types d'évènements de BPMN2.0 sont représentée par la figure suivante :

Fig. 2.8 : Les types d'événements de BPMN2.0

- **L'évènement de message** : la réception ou l'envoi d'un message à un Participant.
- **L'évènement de minuterie** : les minuteries peuvent être réglées pour se déclencher à des intervalles spécifiques ou à des dates et heures fixes liées au calendrier.
- **L'évènement d'escalade** : une étape réagit à une escalade et passe à un autre rôle dans l'organisation. Cet événement n'est utilisé que dans un sous-processus d'événement. Une escalade se produit lorsqu'une personne à un niveau de responsabilité supérieur au sein de l'organisation est impliquée dans un processus.
- **L'évènement de condition** : ce type d'événement est déclenché lorsqu'une condition devient vraie. L'expression de condition pour l'événement doit devenir fausse puis vraie avant que l'événement puisse être déclenché à nouveau.
- **L'évènement de lien** : un lien est un mécanisme permettant de connecter deux sections d'un processus. Les événements de lien peuvent être utilisés pour créer des situations en boucle ou pour éviter de longues lignes de flux de séquence.
- **L'évènement d'erreur** : un erreur est un mécanisme permettant de terminer le processus et envoie un signal d'erreur à traiter via un sous-processus d'erreur.
- **L'évènement de compensation** : la compensation concerne l'annulation d'étapes déjà terminées avec succès. La compensation est déclenchée par un événement de compensation lancé, qui sera déclenché par un gestionnaire d'erreurs, dans le cadre annulation, ou récursivement par un autre gestionnaire de compensation.
- **L'évènement d'annulation** : réagit à une transaction qui a été annulée dans un sous-processus. Dans un événement de fin, le symbole d'annulation représente l'annulation déclenchée d'un processus.
- **L'évènement de signal :** les événements de signal sont utilisés pour indiquer l'envoi ou la réception d'un signal.
- **L'évènementmultiple :** le type multiple est utilisé afin de lancer ou recevoir un des déclencheurs associés à l'événement.
- **L'évènement multiple parallèle** : cet initiateur est similaire à un multiple, mais pour déclencher le processus, toutes les règles doivent être respectées.
- **L'évènement de terminaison** : ce type de fin indique que toutes les activités du processus doivent être immédiatement terminées.

(3) **Les passerelles(Gateways)** : les passerelles sont utilisées pour contrôler la façon dont les flux de séquence interagissent lorsqu'ils convergent et divergent au sein d'un processus. Le terme « passerelle » implique qu'il existe un mécanisme de déclenchement qui autorise ou interdit le passage par la passerelle. Les différents passerelles possibles sont :

FIG. 2.9 : Les passerelles de BPMN2.0

- **Exclusive (XOR)** : Une porte divergente exclusive est utilisée pour créer des chemins alternatifs. Il s'agit simplement d'un ensemble de chemins alternatifs dans un flux de processus, mais un seul d'entre eux sera choisi.
- **Inclusive (OR)** : Une passerelle inclusive divergente peut être utilisée pour créer des chemins alternatifs mais aussi parallèles, elle représente un choix inclusif parmi plusieurs flux possibles toutes les combinaisons de chemins peuvent être prises ,peuvent choisir de une à tout.
- **Parallèle** : Une passerelle parallèle est utilisée pour synchroniser (combiner) des flux parallèles et créer des flux parallèles. Une passerelle parallèle crée des chemins parallèles sans vérifier aucune condition. Pour les flux entrants, la passerelle parallèle attendra tous les flux entrants avant de déclencher le flux via ses flux de séquence sortants.
- **Complexes** : Comme leur nom l'indique, les passerelles complexes ne sont utilisées que pour les flux les plus complexes du processus métier. Elle Représente un choix exprimé par une condition inexprimable par les autres portes disponibles.
- **Basée sur les événements** : Est similaire à une passerelle exclusive car les deux impliquent un chemin dans le flux. Dans le cas d'une passerelle basée sur les événements, cependant, vous évaluez quel événement s'est produit, et non quelle condition a été remplie.
- **Parallèle basée sur les événements** : cette passerelle est similaire à une passerelle parallèle. Il permet à plusieurs processus de se produire en même temps, mais contrairement à la passerelle parallèle, les processus dépendent d'événements spécifiques.

2. **Les données** : Les données sont des éléments physiques ou des informations qui sont créés, manipulés ou utilisés lors de l'exécution d'un processus.

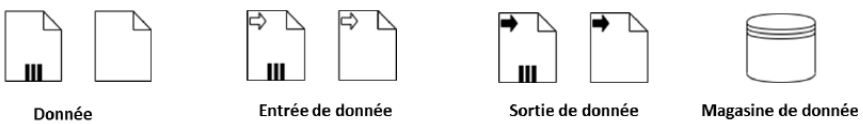


FIG. 2.10 : Les données de BPMN 2.0

- **Donnée (Data)** : Représente une information passant par le processus.
- **Entrée de donnée (Data Input)** : Représente une information externe entrante au processus.

- **Sortie de donnée (Data Output)** : Représente une information externe sortante au processus.
- **Magasine de donné (Data Stores)** : Fournit un mécanisme permettant aux Activités de récupérer ou de mettre à jour les informations stockées qui persisteront au-delà de la portée du Processus. Le même DataStore peut être visualisé, via une Référence de Data Store, à un ou plusieurs endroits du Processus.

3. **Les objets de connexion**

- **Flux de séquence** : un flux de séquence est utilisé pour connecter des objets de flux et pour afficher l'ordre des éléments de flux dans un processus ou une chorégraphie. La source et la cible de flux de séquence doivent appartenir à l'ensemble des éléments de flux suivants : événements, activités, activités de chorégraphie et passerelles.BPMN 2.0 utilise cinq (5) type de flux de séquence.

Fig. 2.11 : Les flux de séquence de BPMN 2.0.

  - **Les flux normal** : fait référence aux chemins du flux de séquence qui ne commencent pas à partir d'un événement intermédiaire attaché à la limite d'une activité.
  - **Un flux incontrôlé** : fait référence à un flux qui n'est affecté par aucune condition ou qui ne passe pas par une passerelle. L'exemple le plus simple est un flux de séquence unique reliant deux activités.
  - **Flux conditionnel** : Un flux de séquence peut avoir une expression qui est évaluée au moment de l'exécution pour déterminer si le flux de séquence sera utilisé ou non. Si le flux conditionnel sort d'une activité, le flux de séquence aura un mini diamant au début du connecteur. Si le flux conditionnel sort d'une passerelle, la ligne n'aura pas de mini-diamant (Flux par défaut).
  - **Flux par défaut** : Pour les passerelles exclusives basées sur les données ou les passerelles inclusives, un type de flux est le flux de condition par défaut. Ce flux ne sera utilisé que si tous les autres flux conditionnels sortants ne sont pas vrais au moment de l'exécution.
  - **Le flux d'exception** : se produit en dehors du flux normal du processus et est basé sur un événement intermédiaire attaché à la limite d'une activité qui se produit pendant l'exécution du processus.

- **Les flux de message(Message Flows)** : Les flux de messages assurent l envoi de messages d une entité à une autre. Ils doivent connecter un pool ou

un objet se situant dans ce pool à un autre pool ou un objet se situant dans ce dernier.(voir la figure 2.12).

FIG. 2.12 : Les flux de message de BPMN 2.0.

- **Associations** : Une association est utilisée pour lier des informations et des artefacts avec des éléments graphiques BPMN. Des annotations de texte et d'autres artefacts peuvent être associés aux éléments graphiques. Une pointe de flèche sur l'association indique une direction de flux. (voir la figure 2.13)

FIG. 2.13 : Les associations de BPMN 2.0.

4. **Les partitions (Swimlans)** :

- **Les participant (Pool)** : est la représentation graphique d'un participant dans une collaboration. Il agit également comme un « swimlane » et un conteneur graphique pour partitionner un ensemble d'activités d'autres pools, généralement dans le cadre de situations B2B. Un pool PEUT avoir des détails internes, sous la forme du processus qui sera exécuté. Ou un pool peut n'avoir aucun détail interne, c'est-à-dire qu'il peut s'agir d'une "boîte noire".

FIG. 2.14 : La représentation d'un pool comme boîte noire en BPMN 2.0.

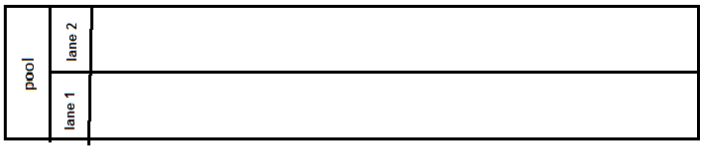


FIG. 2.15 : La représentation d'un pool en BPMN 2.0.

- **Les couloirs (Lane)** : une voie est une sous-partition au sein d'un processus, parfois au sein d'un pool, et s'étendra sur toute la longueur du processus, soit verticalement, soit horizontalement. Les couloirs sont utilisés pour organiser et catégoriser les activités.

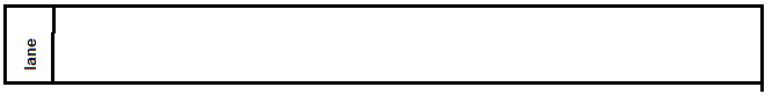


FIG. 2.16 : La représentation d'une lane en BPMN 2.0.

5. **Les artéfacts** : BPMN a été conçu pour permettre aux modélisateurs et aux outils de modélisation une certaine flexibilité dans l'extension de la notation de base et dans la possibilité d'ajouter un contexte approprié à une situation de modélisation spécifique, comme pour un marché vertical (par exemple, l'assurance ou la banque).

FIG. 2.17 : Les artéfacts

- **Group** : un groupe est un regroupement d'éléments graphiques appartenant à la même catégorie. Le nom de la catégorie apparaît sur le diagramme en tant qu'étiquette de groupe. Les groupes sont un moyen d'afficher visuellement les catégories d'objets sur le diagramme.

- **Texte annotation** : les annotations de texte sont un mécanisme permettant à un modélisateur de fournir des informations textuelles supplémentaires au lecteur d'un diagramme BPMN.

2.4.4 Types de diagrammes BPMN

BPMN 2.0 offre quatre types de diagrammes (modèles) pour prendre en compte les différents types de processus : les processus (privés ou publics), les collaborations, les chorégraphies et les conversation.

2.4.4.1 Diagramme de processus (Orchestration)

Diagramme de processus Pour structurer des modèles de processus dans BPMN, un modélisateur peut utiliser des pools et des couloirs. Un pool représente les participants au processus. Un pool agit comme un conteneur pour un processus métier complet. Un système de processus peut être composé de plusieurs pools. Les pools ne peuvent interagir qu'en échangeant des messages. Les couloirs sont utilisés pour organiser et catégoriser les activités au sein d'un pool. Une pools peut être divisée en plusieurs couloirs.
Un diagramme de processus décrit un processus au sein d'une seule entité commerciale qui est contenus dans un pool.[26] Il existe deux types de base de diagramme de processus BPMN .[14]

- **Processus métier privé :** Les processus métier privés sont ceux internes à une organisation spécifique. Ces processus sont généralement appelés workflow ou processus BPM. Il existe deux types de processus privés : exécutables et non exécutables.
  - **Processus métier privé non exécutable :** est un processus qui a été modélisé pour être exécuté selon la sémantique d'exécution BPMN définie,lors du développement cycle du Processus.
  - **Processus métier privé non exécutable :** est un processus privé qui a été modélisé dans le but de documenter le comportement du processus à un niveau de détail défini par le modélisateur.

FIG. 2.18 : Exemple de processus privé.

- **Processus métier public :** représente les interactions entre un processus métier privé et un autre processus ou participant. Seules les activités utilisées pour communiquer avec les autres participants sont incluses dans le processus public. Toutes les autres activités « internes » du processus métier privé ne sont pas affichées dans

le processus public. Ainsi, le processus public montre au monde extérieur les flux de messages et l'ordre de ces flux de messages qui sont nécessaires pour interagir avec ce processus.

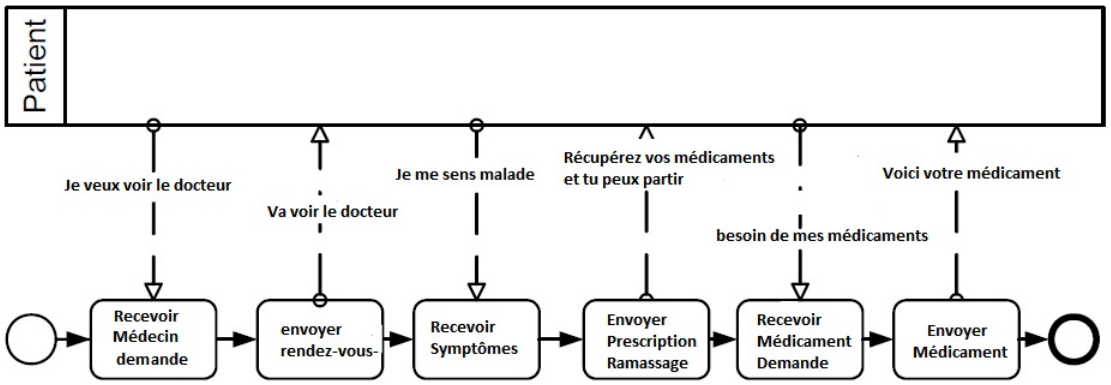


FIG. 2.19 : Exemple de processus public.

2.4.4.2 Chorégraphie

Un modèle de processus de chorégraphie est une définition du comportement attendu (un type de contrat procédural ou de protocole) entre les participants en interaction. Ces participants peuvent être des rôles commerciaux généraux ou être une entité commerciale spécifique. La chorégraphie ressemble à un processus métier privé puisqu'elle se compose d'un réseau d'activités, d'événements et de passerelles. Cependant, une Chorégraphie est différente en ce que les Activités sont des interactions qui représentent un ensemble (1 ou plus) d'échanges de Messages, qui impliquent deux ou plusieurs Participants. De plus, contrairement à un Processus normal, il n'y a pas de contrôleur central, d'entité responsable ou observateur du Processus.

La figure 2.20 montre un exemple simple de diagramme de chorégraphie. Chaque rectangle représente une interaction. Ils montrent le récepteur d'un message dans la bande ombrée, le nom de l'expéditeur dans la bande non grisée et le nom du message dans la milieu. La figure 2.20 est montré que le client envoie le message de commande au le fournisseur. Après ce message, le message de confirmation est envoyé par le fournisseur au client. Enfin, le message produit est envoyé du fournisseur au client.[27][14]

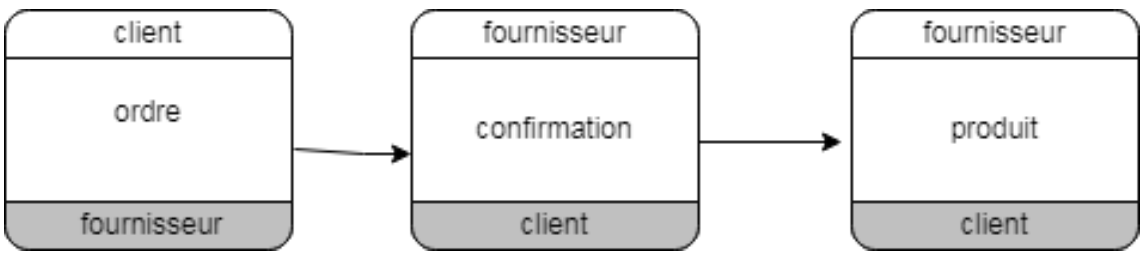


FIG. 2.20 : Diagramme de chorégraphie.

2.4.4.3 Diagramme de collaboration

Une collaboration décrit les interactions entre deux ou plusieurs entités commerciales. Une collaboration contient généralement deux pools ou plus, représentant les participants à la collaboration. Pour être plus précis, une collaboration est un diagramme BPMN qui contient deux participants ou plus, comme indiqué par les pools. L'échange de messages entre les pools est illustré par un flux de messages qui relie deux pools. Une collaboration peut contenir une chorégraphie et une ou plusieurs orchestrations.[27]. La figure 2.21 montre un exemple simple de diagramme de collaboration.

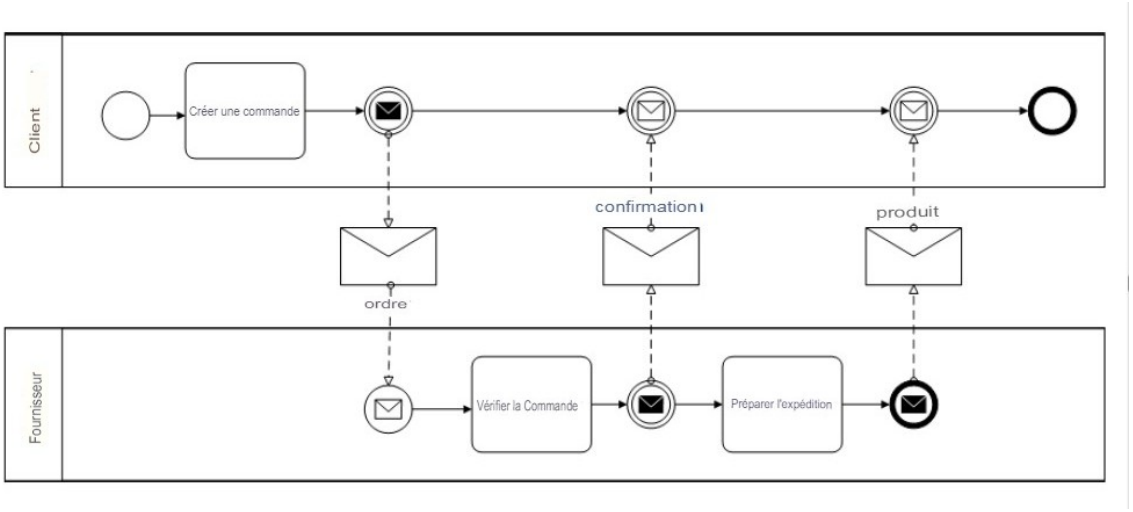


Fig. 2.21 : Diagramme de collaboration.

2.4.4.4 diagramme de conversation

Le diagramme de conversation est une utilisation particulière et une description informelle d'un diagramme de collaboration. Cependant, les pools d'une conversation ne contiennent généralement pas de processus et une chorégraphie n'est généralement pas placée entre les pools d'un diagramme de conversation. Les diagrammes de conversation ne montrent que les parties impliquées et leurs relations de communication. Ils ne décrivent pas les séquences autorisées dans lesquelles les messages sont échangés.[14]

La figure 4.22 présente un exemple simple de diagramme de conversation. La conversation affichée se compose de deux parties ; Client et Fournisseur. Ils échangent des messages qui appartiennent à la communication d'ordre. La communication de commande peut être considérée comme un ensemble de messages uniques qui sont échangés pour coordonner la commande et la livraison de produits. Dans BPMN, les parties impliquées affichées dans un diagramme de conversation correspondent à des pools.[26]

FIG. 2.22 : Diagramme de collaboration.

2.5 Modélisation de l'IdO en utilisant BPMN

2.5.1 Model de domaine de l'IdO en utilisant la norme BPMN

Pour chaque élément du domaine IdO, les propositions de recherche sont décrites et analysées sur leur adéquation à la représentation des concepts comme indiqué ci-dessous.

2.5.1.1 Entité physique

L'Internet des objets est un réseau où plusieurs choses ou objets communiquent entre eux. Ces choses (”thing” in english) intéressent le processus métier, elle peut définis comme une entité physique. Une Entité Physique doit être représentée dans le processus métier. Voici quelques idées pour la représenter [28] :

- Le BPMN 2.0 actuel fournit l'annotation de texte, l'objet de données et le participant en tant que concepts de modélisation pouvant représenter l'entité physique. L'élément Participant représente le mieux pour l'entité physique.
- Meyer et son groupe présentent à la fois un élément graphique et un élément lisible par machine pour le nouveau concept d'entité physique. L'entité physique est représentée comme une couloir(lane) vide avec des lettres horizontales, car elle ne peut contenir aucun élément de flux.
- Une autre façon de voir les choses est l'approche de Meyer. Ils traitent la chose physique comme une boîte noire et un élément indépendant du processus. On peut se demander quelle est la meilleure approche pour modéliser une entité physique. ils doivent être utilisés dans des cas d'utilisation et doivent être comparés pour déterminer la meilleure solution.

2.5.1.2 Utilisateur

L'utilisateur doit être modélisé d'une manière qui peut accéder aux services ou aux tâches des appareils. BPMN contient des concepts suffisants pour l'invocation des tâches. Il peut être modélisé en tant que participant envoyant un message, un événement ou une ressource. Nous pouvons conclure que BPMN est suffisant pour représenter l'utilisateur. Mais nous ne devons modéliser l'utilisateur que lorsqu'il est pertinent pour le processus et lorsqu'il participe activement.[29]

2.5.1.3 Entité virtuelle

Une approche de modélisation de l'entité virtuelle par Sperner. consiste à la représenter comme un objet de données. Ils soutiennent que l'entité virtuelle est intrinsèquement définie comme un artefact numérique et qu'un objet de données est le concept actuel pour modéliser les données dans BPMN. Lors de la modélisation de l'entité virtuelle, cela rendra le modèle trop compliqué et peut confondre le lecteur.[29]

2.5.1.4 Entité augmentée

Sperner déclarent que l'entité augmentée est un lien abstrait entre une entité physique et une entité virtuelle, elle n'ajoute aucun avantage lorsqu'elle est modélisée. De plus, l'entité virtuelle n'étant pas modélisée, modéliser l'entité augmentée n'aurait aucun sens. Nous concluons que l'entité augmentée ne doit pas être modélisée.[29]

2.5.1.5 Appareil IdO

L'appareil IdO est une partie importante du processus métier compatible avec l'IdO. Il assure la connexion entre le monde physique et le monde numérique. L'appareil IdO peut exécuter des tâches telles que la détection, d'actionnement et de surveillance.

La recherche ne distingue qu'un seul élément BPMN étroitement lié pour le dispositif IdO, le participant au processus. Dans BPMN, le participant au processus est représenté par un pool ou un couloir. Le concept incarne une entité qui exécute un processus métier. Le processus doit être modélisé à l'intérieur de son propre participants(pool) ou couloir. Bien que le participant au processus puisse représenter un appareil IdO, la conclusion est que La représentation a encore quelques lacunes et devrait être étendue.[29]

2.5.1.6 Ressource et service

Dans le modèle de domaine IdO, les ressources sont les composants logiciels sur les appareils IdO. Dans BPMN, les ressources signifient autre chose, cela signifie que les participants exécutent des activités. En raison de ces définitions divergentes et de la confusion qui en résulte, des auteur comme Sperner et Meyer conclurent que les ressources ne doivent pas être modélisées.

Spener expliquent que le service est similaire à la tâche de service de l'élément BPMN.Le problème avec cette approche est qu'il est impossible de dire si le service est un une tâche de détection ou d'actionnement.

La ressource et le service d'un appareil IdO sont similaires car ils sont tous deux des logiciels exécutés sur l'appareil IdO pour activer les fonctionnalités de détection et d'actionnement. Étant donné que le modèle de processus n'a besoin que de la tâche de détection et d'actionnement pour être représenté, la ressource d'un service peut être abstraite en tant qu'activité IdO.[29]

2.6 Un extensions BPMN pour la modélisation de l'IdO

Le BPMN n'est pas suffisant pour présenter tous les éléments de l'IdO en nécessite deux éléments très importants pour modéliser sont [28] :

- **Tâche de détection(Sensing Task)** La détection est une tâche spécifique aux appareils IdO qui est généralement exécutée par un capteur. La tâche de détection extrait les données de l'entité physique et doit donc avoir une connexion directe avec celle-ci.
  Les activités de détection n'ont aucune intervention humaine, ni ne sont connectées à un participant externe, en d'autres termes, il n'y a pas de flux de messages et il n'y a pas d'ensemble de données d'entrée. L'activité de détection est ajoutée au métamodèle et liée à un ensemble d'exigences afin de exprimer les propriétés nécessaires.

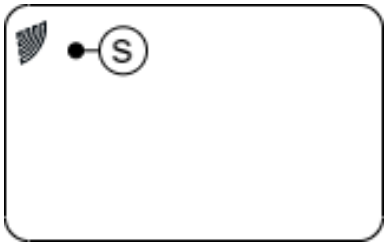


Fig. 2.23 : Tâche de détection.

- **Tâche Actionnement** L'actionneur fonctionne exactement dans le sens inverse d'un capteur. Au lieu d'observer l'environnement, il exécute une action physique. La plupart du temps, les données du capteur sont utilisées pour envoyer une commande à l'actionneur pour déclencher l'action. L'exécution d'une tâche d'actionnement peut modifier l'état de l'entité physique. Un exemple simple serait le déverrouillage d'une porte. L'état de la porte, l'entité physique, passe de verrouillé à déverrouillé.

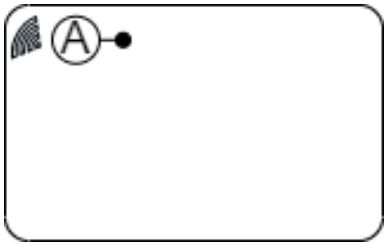


Fig. 2.24 : Tâche Actionnement.

2.7 Une comparaison entre BPMN 2.0, eEPC et UML2.3 pour la modélisation de l'IdO

Le projet IoT-A a examiné trois notations standard pour identifier celle qui convient le mieux à la modélisation de processus pour l'IdO. BPMN 2.0, eEPC et UML2.3 ont été évalués sur les onze exigences Suivant [29] :

1. **Concept basé sur l'entité** : Le modèle de processus doit inclure les différents éléments spécifiés par le modèle de domaine IdO. En raison de la nature et de la diversité de l'IdO, il est important d'étendre le BPMN pour décrire les fonctionnalités et capacités supplémentaires des appareils IdO.

2. **Exécution distribuée** : Une propriété que possèdent la plupart des réseaux IdO est une exécution de processus distribuée sur de nombreux appareils IdO, contrairement à une exécution de processus centralisée inhérente à la plupart des processus métier.

3. **Interactions** : Un réseau IdO introduit certaines formes spécifiques d'interactions avec les processus métier (comme les services interagissant avec les ressources sur les appareils). Le modèle de processus doit pouvoir représenter ces différents types d'interaction. BPMN n'a pas la capacité de modéliser ces interactions.

4. **Données distribuées** : De nombreux appareils IdO contiendront leurs propres magasins de données, ce qui est contraire à un magasin de données central utilisé dans la plupart des processus métier. Le modèle de processus doit être capable de distribuer les données sur la gamme d'appareils.

5. **Évolutivité** : Une exigence pour représenter la dynamique d'augmentation et de diminution des appareils au fil du temps.Le modèle de processus doit être capable de gérer l'évolutivité du réseau IdO. En d'autres termes, la complexité et les performances d'exécution doivent rester constantes avec un nombre croissant d'appareils IdO et entités physiques.

6. **Abstraction** : Le modèle de processus doit être capable d'abstraire les activités en activités de plus grande valeur. Cela devrait permettre l'abstraction des composants de processus individuels à l'entité physique.

7. **Disponibilité/Mobilité** : Un problème spécifique à l'IdO Représente la possibilité de défaillance des appareils, de suppression, de qualité de service en raison du manque de fiabilité des ressources. Important dans la définition de nouveaux éléments IdO dans BPMN et pendant l'exécution car le moteur doit gérer les appareils intermittents.

8. **Tolérance aux pannes** : La précédente propriété spécifique à l'IdO des appareils IdO non fiables/indisponibles peut impliquer que les données qui en découlent peuvent être sujettes aux erreurs, ce qui est une préoccupation majeure car les processus métier doivent être hautement fiables.

9. **Flexibilité/basé sur les événements** : Dans des environnements complexes, il est souvent nécessaire d'agir rapidement sur des événements se produisant dans

le monde réel. Des capteurs peuvent être intégrés dans un processus métier pour détecter ces événements. Ces processus IdO reposent sur une communication basée sur les événements. Ces événements spécifiques à l'IdO doivent être modélisés pour créer une logique de flux de contrôle plus flexible.

10. **Incertitude des informations** : Le modèle de processus doit pouvoir exprimer l'incertitude des informations fournies par les dispositifs IdO.

11. **Temps réel** : La dernière exigence postule ceci : « Le modèle doit fournir des moyens pour exprimer des contraintes en temps réel.

Les résultats de l'évaluation sont présentés dans le tableau 2.3. Le score total est calculé comme suit : oui donne un point, en partie un demi-point et non zéro point. Les résultats montrent que BPMN 2.0 est l'approche la plus appropriée pour modéliser des processus sensibles à l'IdO.[28]

IdO Caractéristiques	BPMN 2.0	eEPC	UML 2.3
Concept basé sur les entités	Partiellemente	Non	Partiellement
Exécution distribuée	Partiellement	Non	Partiellement
Interactions	Non	Non	Non
Données distribuées	Partiellement	Non	Partiellement
Évolutivité	Partiellement	Non	Partiellement
Abstraction	Oui	Non	Partiellement
Disponibilité/Mobilité	Non	Non	Non
Tolérance aux pannes	Partiellement	Non	Partiellement
Flexibilité/ basée sur les événements	Oui	Partiellement	Partiellement
Incertitude des informations	Non	Non	Non
En temps réel	Oui	Non	Non
Note totale	5,5	0,5	3,5

Tab. 2.3 : Caractéristiques IdO par notations BPM

Les résultats montrent que BPMN 2.0 est l'approche la plus appropriée pour modéliser des processus sensibles à l'IdO.

2.8 Conclusion

L'objectif de ce chapitre est de comprendre et connaître le concept de BPMN et les informations les plus importantes le concernant pour connaître le rôle de ce dernier dans la modélisation du comportement des applications IdO.

Chapitre 3

Ingénierie dirigée par les modèles

3.1 Introduction

Pour appréhender la complexité du logiciel, tout en utilisant une pratique de spécification de très haut niveau. Se base sur l'IDM, ce dernier est baser sur deux mécanismes (La création de langages de modélisation spécifiques au domaine, L'utilisation de moteurs de transformation et de générateurs).
Ce chapitre se structure en deux parties. La première vise à présenter les concepts de base de de l'Ingénierie Dirigée par les Modèles (IDM). La seconde partie est présenter transformation model.

3.2 Définition

L'IDM est une approche moderne du génie logiciel, elle est née du constat que le modèle objet a atteint ses limites à la fin du siècle dernier. Le site production de logiciels a reposé, et repose toujours, en grande partie sur la fabrication de codes. La production de masse et la pénurie de main-d'œuvre qualifiée ont fait naître l'idée de la fabrication de la production de logiciels. L'industrie du logiciel est en train de prendre l'initiative avec la proposition de la spécification MDA (Model Driven Architecture) par l'Object Management Group (OMG 2001).[30]
Donc, l'IDM est une méthode de développement de logiciel, s'appuyant sur la création et l'utilisation de modèles spécifiques à un domaine. L'utilisation de ces modèles permet de s'affranchir du choix d'une plateforme spécifique de développement. L'IDM a engendré un changement important dans les pratiques du génie logiciel.

3.3 Principes généraux de l'IDM

Suite à l'approche objet des années 80 Le principe de base de 'ingénierie dirigée par les modèles (IDM) est "tout est un modèle", comparé au principe de base de l'orientation objet "tout est un objet". Cependant, l'IDM a une portée large, combinant le processus et l'analyse avec l'architecture. L'IDM est une approche ouverte et intégrée qui englobe de nombreux autres domaines technologiques de manière unifiée. Un espace technologique est un contexte de travail avec son ensemble associé de concepts, de connaissances, d'outils, de compétences et de capacités.
L'IDM vise donc, de manière plus radicale que pouvaient l'être les approches des patterns et des aspects, à fournir un grand nombre de modèles pour exprimer séparément chacune des préoccupations des utilisateurs, des concepteurs, des architectes, etc. C'est par ce principe de base fondamentalement différent que l'IDM peut être considérée en rupture par rapport aux travaux de l'approche objet. [31]
L'objectif de l'IDM est de définir une approche pouvant intégrer différents espaces technologiques Par exemple, l'espace technologique des grammaires, des documents structurés XML, ou des bases de données relationnelles, ne sont pas fondés sur le concept d'objet. Les deux relations de base InstanceOf et InheritsFrom ne sont pas adaptées aux autres espaces technologiques comme elles le sont pour le monde des objets. L'IDM vise à fournir un grand nombre de modèles pour exprimer séparément chacune des préoccupations des

utilisateurs, des concepteurs, des architectes, etc. C'est par ce principe de base, fondamentalement différent, que l'IDM peut être considérée en rupture par rapport aux travaux de l'approche objet. Ainsi, il est nécessaire de définir quels sont les concepts et les relations essentielles à l'IDM.

3.4 Notion de base en Ingénierie Dirigée par les Modèles

Les éléments de base de l'IDM sont :

3.4.1 Système

Un système est une entité complexe formée d'un ensemble ordonné d'éléments liés les uns aux autres et qui interagissent entre eux.[32]

3.4.2 Modèle

Il existe diffèrent définition de terme modelé, dans la suit nous citons quelque définitions :

- Un modèle est une abstraction d'un système, modélisé sous la forme d'un ensemble de faits construits dans une intention particulière. Un modèle doit pouvoir être utilisé pour répondre à des questions sur le système modélisé[31].
- Un modèle est une représentation d'un système pour véhiculer l'information d'une réalité.
- Un ensemble de déclarations sur le système à étudier[33].
- Jean Bézivin et Olivier Gerbé défini un modèle comme la suit : " Un modèle est une simplification d'un système construit dans un but précis. Le modèle doit être capable de répondre à des questions à la place du système réel "[30].

3.4.3 Meta-modèle

- Un méta-modèle est le langage de modélisation pour décrire un modèle.
- Un méta-modèle est un modèle qui définit le langage d'expression d'un modèle et le langage de modélisation[31].
- Un méta-modèle est un modèle de spécification pour lequel les systèmes à spécifier sont des modèles dans un certain langage de modélisation[33].
- Favre 2004 : " Un métamodèle est un modèle de modèles "[30].

3.4.4 Méta-méta-modèle (MOF)

- MOF (Meta-Object Facility) : est un ensemble d'interfaces standards permettant de définir et de modifier des méta-modèles et leurs modèles correspondants. Le MOF est un standard de métamodélisation pour définir la syntaxe et la sémantique d'un langage de modélisation, il a été créé par l'OMG afin de définir la notation UML[34].
- MOF est un méta-méta-modèle est un modèle qui décrit un langage de méta-modélisation, les éléments de modélisation nécessaires à la définition des langages de modélisation. Il a de plus la capacité de se décrire lui-même[31].
- Seidewitz définie MOF comme un méta-métamodèle fait des déclarations sur ce qui peut être exprimé dans les métamodèles valides d'un certain langage de méta-modélisation"[30].

3.5 L'architecture à quatre niveaux de MOF

Cette architecture est hiérarchisée en quatre niveaux. En partant du bas [35] :

- **Le niveau M0 (ou instance) :** correspond au monde réel. Ce sont les informations réelles de l'utilisateur, instance du modèle de M1.
- **Le niveau M1 (ou modèle) :** est composé de modèles d'information. Il décrit les informations de M0. Les modèles UML, les PIM et les PSM appartiennent à ce niveau. Les modèles M1 sont des instances de méta-modèle de M2.
- **Le niveau M2 (ou méta-modèle :)** il définit le langage de modélisation et la grammaire de représentation des modèles M1. Le méta-modèle UML qui est décrit dans le standard UML, et qui définit la structure interne des modèles UML, fait partie de ce niveau. Les profils UML, qui étendent le méta-modèle UML, appartiennent aussi à ce niveau. Les méta-modèles sont des instances du MOF.
- **Le niveau M3 (ou méta-méta-modèle) :** est composé d'une unique entité qui s'appelle le MOF. Le MOF permet de décrire la structure des méta-modèles, d'étendre ou de modifier les métamodèles existants. Le MOF est réflexif, il se décrit lui-même.

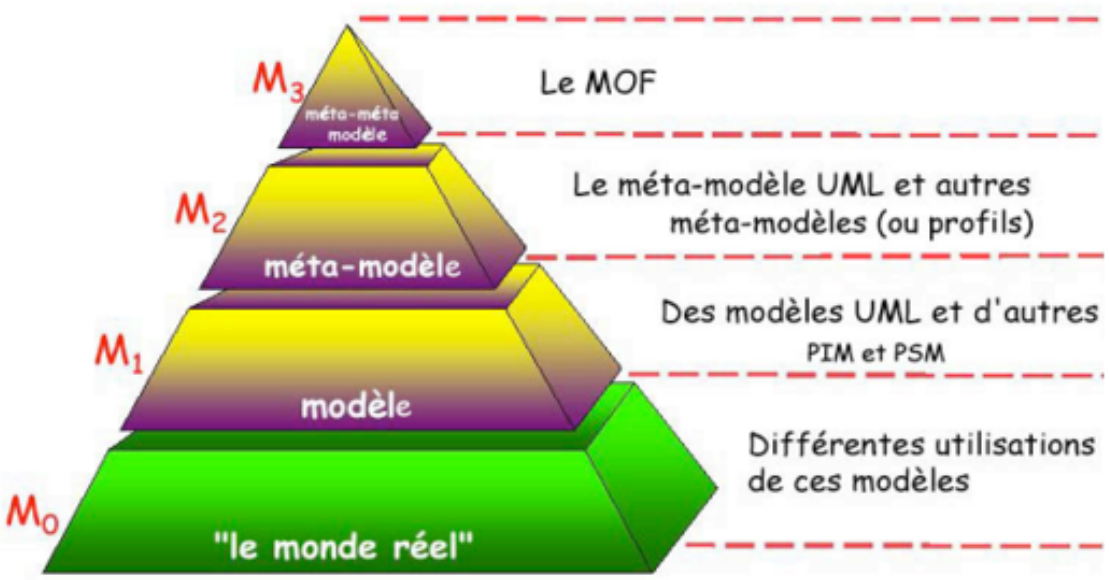


FIG. 3.1 : architecture à quatre niveaux.

3.6 Transformation des modèles

3.6.1 Définition de base

La transformation de modèle est le cœur et l'âme de l'IDM. C'est une manière de modifier et de créer des modèles. En effet, la notion de transformation de modèles porte sur l'automatisation de l'opération de transformation pendant le cycle de développement, qui peut avoir des sémantiques différentes en fonction des utilisations : raffinement, optimisation, génération.
La transformation de modèles est l'opération qui consiste à générer un ou plusieurs modèles cibles conformément à leurs méta-modèles à partir d'un ou plusieurs modèles sources conformément à leurs méta-modèles en se basant sur des règles de transformation bien définies. La transformation est défini de la manière suivante : [36]
Une transformation est une génération automatique d'un ou plusieurs modèles cibles à partir d'un ou plusieurs modèles sources, en respectant la définition de transformation. Une définition de transformation est un ensemble de règles de transformation qui d´écrivent la manière avec laquelle un modèle dans le langage source peut être transforme en un modèle dans le langage cible. Une règle de transformation est une description de la façon avec laquelle une ou plusieurs constructions dans le langage source peuvent être transformées en une ou plusieurs constructions dans le langage cible.

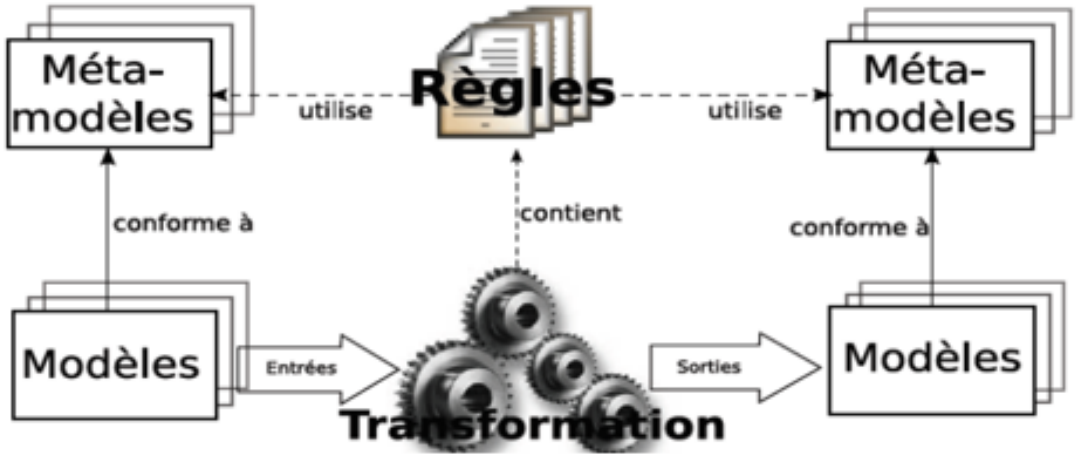


Fig. 3.2 : Architecture de la transformation des modèles.

On qualifie une transformation de modèles d'exogène lorsque les méta-modèles source et cible sont différents, en d'autres termes lorsque l'on veut produire un modèle cible exprimé dans un formalisme différent que celui dans lequel est exprimé le modèle source. Inversement, une transformation dans laquelle le modèle source et le modèle cible possèdent le même méta-modèle est qualifiée d'endogène. Les règles de transformation sont définies sur un unique méta-modèle.

On a recours à des transformations endogènes dans le cadre, par exemple, d'une optimisation de modèles, ou bien d'une simplification, ou encore de refactoring. On utilise des transformations exogènes pour toute transformation de modèle impliquant des méta-modèles différents, l'exemple le plus significatif étant sans aucun doute la génération de code. On fait également appel aux transformations exogènes pour effectuer des migrations de modèles (depuis une plateforme vers une autre) tout en restant au même niveau d'abstraction, ou encore dans opérations de reverse engineering (rétro-ingénierie ou rétro-conception)[37].

3.6.2 Types de transformation

Le niveau d'abstraction est un facteur important à prendre en considération. On basant sur ce dernier on peut distinguer les types de transformation suivants [37] :

3.6.2.1 Transformation horizontale :

Une transformation horizontale modifie la représentation source tout en conservant le même niveau d'abstraction. La modification peut être l'ajout, la modification ou la suppression d'informations.

3.6.2.2 Transformation verticale :

La source et la cible d'une transformation verticale sont définies aux différents niveaux d'abstraction. La transformation est un raffinement, si elle baisse le niveau d'abstraction "PIM (Platform Independent Model) vers PSM(Platform Specific Model) ", sinon c'est une abstraction (elle élève le niveau "PSM vers PIM").

3.6.2.3 Transformation oblique :

Une transformation oblique combine une transformation horizontale et une verticale. Ce type de transformation est notamment utilisé par les compilateurs, qui effectuent des optimisations du code source avant de générer le code exécutable.

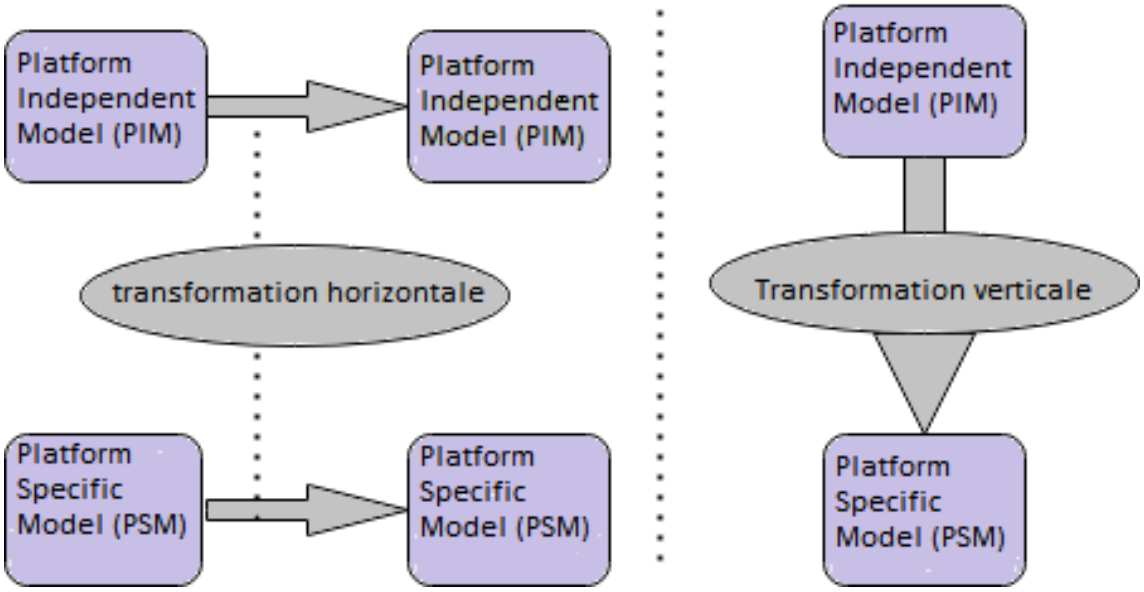


FIG. 3.3 : Les types de transformations.

3.6.3 Les approches de transformation des Modèles :

Nous classons ces approches en deux grandes familles : les approches de modèle à modèle ou M2M, et les approches de modèle à texte ou M2T [37] [38].

3.6.3.1 Transformation de modèle à modèle M2M :

Les transformations de modèle à modèle traduisent entre les modèles source et cible, qui peuvent être des instances de métamodèles identiques ou différents. Toutes ces approches supportent le typage syntaxique des variables et des modèles. Nous distinguons cinq techniques de transformation « Modèle vers Modèle » :

Approches de manipulation directe : Ces approches offrent une représentation interne du modèle ainsi qu'une API pour le manipuler. Elles sont généralement mises en

œuvre dans un cadre orienté objet, qui peut également fournir une infrastructure minimale pour organiser les transformations (par exemple, une classe abstraite pour les transformations). Toutefois, les utilisateurs doivent mettre en œuvre les règles de transformation et la programmation en partant de zéro, en utilisant un langage de programmation tel que Java.

Approches relationnelles : Cette catégorie regroupe les approches déclaratives dont le concept principal est constitué de relations mathématiques. L'idée de base est d'indiquer le type d'élément source et cible d'une relation et de la spécifier à l'aide de contraintes. Dans sa forme pure, une telle spécification est non exécutable. Cependant, les contraintes déclaratives peuvent être dotées d'une sémantique exécutable, comme dans la programmation logique.

Approches basées sur la transformation de graphes : Ces approches sont similaires aux approches relationnelles, dans le sens où elles permettent l'expression des transformations sous une forme déclarative. Cependant, les règles sont exprimées sous forme graphique et elles ne sont plus décrites pour des éléments simples mais pour des fragments de modèles Cette transformation de modèles s'inspire des travaux théoriques sur les transformations de graphes. En particulier, ces approches opèrent sur des graphes typés, attribués, étiquetés, qui sont un type de graphes spécifiquement conçus pour représenter des modèles de type UML. Les exemples d'approches de transformation de graphes pour la transformation de modèles incluent VIATRA, ATOM, GreAT, UMLX, et BOTL.

Approches à base de structure : Les approches de cette catégorie comportent deux phases distinctes : la première phase concerne la création de la structure hiérarchique du modèle cible, tandis que la deuxième phase définit les attributs et les références dans la cible. Le cadre général détermine la stratégie d'ordonnancement et d'application ; les utilisateurs ne s'occupent que de fournir les règles de transformation.

3.6.3.2 Transformation modèle vers texte, M2T :

Cette catégorie permet la génération de code textuel dans un langage de programmation (Java, XML, Maude, etc.) à partir d'un modèle source. La transformation modèle vers code est considérée comme un cas particulier de la transformation modèle vers modèle où le méta-modèle cible se restreint à la grammaire textuelle du langage de programmation de destination. On distingue deux approches de transformation modèle vers texte :

Approches basées sur les visiteurs : Une approche très basique de génération de code consiste à fournir un mécanisme visiteur pour traverser la représentation interne d'un modèle et écrire du code dans un flux de texte. Un exemple de cette approche est Jamda, qui est un cadre orienté objet fournissant un ensemble de classes pour représenter les modèles UML, une API pour manipuler les modèles, et un mécanisme visiteur (appelé CodeWriters) pour générer du code. Jamda ne supporte pas la norme MOF pour définir de nouveaux métamodèles ; cependant, de nouveaux types d'éléments de modèle peuvent être

introduits en sous-classant les classes Java existantes qui représentent les types d'éléments de modèle prédéfinis.

Approches basées sur les templates : Un template de modèle est un modèle comportant du méta-code intégré dont le but est d'évaluer, de calculer, les parties variables des instances correspondant à ce modèle. Souvent, ces templates sont exprimés dans la syntaxe concrète du langage cible, et le métacode utilisé est en OCL. Cette catégorie est actuellement la plus utilisée dans les outils MDA disponibles.

3.7 Avantage et inconvénient de l'IDM

L'approche IDM a des points forts et des points faibles selon [32] [39] sont :

3.7.1 Avantages de l'IDM :

- Les technologies IDM avec une l'architecture et l'automatisation permettent d'atteindre des niveaux d'abstraction plus élevés dans le développement de systèmes modèles.
- Il est conçu comme un moyen de doter un langage ou un formalisme d'une notation d'une notation abstraite, en séparant la syntaxe et la sémantique abstraites de la sémantique abstraite du langage de ses différentes notations concrètes.
- Bien que les constituants de base de l'IDM soient encore en évolution, certaines mises en œuvre des principes de l'IDM peuvent dans les cadres de Métamodélisation/programmation tels que l'OMG MDA (Model Driven Architecture) Modelintegrated Computing (MIC), Software Factories et Microsoft DSL Tools, Eclipse/EMF.
- Le processus de développement dans cette approche revient à raffiner, maintenir, et éventuellement de transformer les modèles en d'autres modèles ou de générer le code exécutable. Il est important à noter que ces différentes activités se font d'une manière automatique.

3.7.2 Inconvénients de l'IDM

- La définition de la sémantique de cette classe de langages est un problème ouvert et question ouverte et cruciale. Actuellement, les environnements de méta-modélisation sont capables de faire face à la plupart des problèmes de définition syntaxique et de transformation, mais il leur manque un support standard et rigoureux pour fournir la sémantique (éventuellement exécutable) des méta-modèles
- Le manque de notations conviviales, d'intégration des techniques et d'interopérabilité des outils reste un défi important pour les méthodes formelles. Défi pour les méthodes formelles.

- Le problème de fournir un moyen d'exprimer la sémantique des métamodèles est très difficile à résoudre. D'exprimer la sémantique des langages basés sur des méta-modèles et de d'effectuer la validation et la vérification des modèles peut être résolu par l'utilisation de FMs.

3.8 Outils de l'IDM

Outils pour :

- La définition de méta-modèles
- La définition de transformations et leur exécution
- La génération de code
- La génération d'environnement graphique de modélisation
- La vérification de conformité de modèles
- La spécification de contraintes

Parmi celles-ci on peut citer [40] :

- Eclipse Modeling Project
- AToM3(A Tool for Multi-Formalism and Meta-Modelling)
- VIATRA [VIATRA, 2010],VIsual Automated model TRAnsformations
- BOTL (Bidirectional Object-oriented Transformation Language)
- AGG (Attributed Graph Grammar) [Taentzer 2003
- ...etc

3.9 Outils de l'IDM utiliser dans notre travail

3.9.1 Eclipse Modeling Tools

Le package Modeling fournit des outils et des temps d'exécution pour construire des applications basées sur des modèles. On peut l'utiliser pour concevoir graphiquement des modèles de domaine, pour tirer parti de ces modèles au moment de la conception en créant et en éditant des instances dynamiques, pour collaborer via le support d'équipe d'Eclipse avec des facilités pour comparer et fusionner des modèles et des instances de modèle structurellement, et enfin générer du code Java à partir de ces modèles pour produire des applications complètes [41].

3.9.2 Plate-forme de métamodélisation sous Eclipse Modeling Tools

Eclipse est une plateforme open-source, son utilisation en tant qu'environnement de développement est particulièrement répandue dans la communauté Java. Il s'agit d'autre part de l'environnement retenu par les développeurs d'ITK pour la réalisation des applications orientées aide à la décision. Son architecture, basée sur le concept de plug-in, est orchestrée par la fondation d'Eclipse.
Il est décomposé en un ensemble de sous-projets focalisés sur des aspects spécifiques de l'IDM : métamodélisation, édition graphique de modèles, édition textuelle, transformations, persistance de modèles.
Dans le cadre du travail de thèse, nous nous sommes orientés sur l'utilisation : d'EMF (Eclipse Modeling Framework), qui est l'élément de base de la métamodélisation sous Eclipse ; Sirius, qui permet de créer facilement des outils de modélisation graphique et Acceleo pour la transformation (transformation M2T).

3.9.2.1 Eclipse Modeling Framework (EMF)

Le cadre de modélisation Eclipse (EMF) est un outil permettant de construire des modèles de structures de données. L'EMF, étant donné qu'il est lié à un outil particulier (Eclipse), présente quelques avantages supplémentaires dans la mesure où il peut générer des outils et des applications raffinées qui sont adaptés à Eclipse. À la base, il est encore assez étroitement lié à la création de modèles logiciels. EMF est un sous-ensemble restreint des concepts de diagramme de classe UML, à savoir les définitions de classes, de classes et d'objets. à savoir les définitions des classes, des attributs appartenant à ces classes et des relations entre les classes.[42]
EMF se compose de trois éléments fondamentaux [43] :

- Ecore : pour décrire les modèles et le support d'exécution des modèles, y compris la notification des changements, le support de la persistance avec la sérialisation XMI par défaut, et une API réflective très efficace pour manipuler les objets EMF de manière générique.
  Le fichier ecore permet de définir les éléments suivants.
  - EClass : représente une classe, avec zéro ou plusieurs attributs et zéro ou plusieurs références.
  - EAttribute : représente un attribut qui a un nom et un type.
  - EReference : représente une extrémité d'une association entre deux classes. Il possède des drapeaux pour indiquer s'il représente un confinement et une classe de référence vers laquelle il pointe.
  - EDataType : représente le type d'un attribut, par exemple, int, float ou java. util. Date. Le modèle Ecore présente un objet racine représentant l'ensemble du modèle. Ce modèle a des enfants qui représentent les packages, dont les enfants représentent les classes, tandis que les enfants des classes représentent les attributs de ces classes.

- EMF.Edit : Le cadre EMF.Edit comprend des classes génériques réutilisables pour construire des éditeurs de modèles EMF. Il fournit :
  - Des classes de fournisseurs de contenu et d'étiquettes, un support de source de propriété et d'autres classes de commodité qui permettent aux modèles EMF d'être affichés en utilisant des visualiseurs de bureau standard (JFace) et des feuilles de propriété.
  - Un cadre de commande, comprenant un ensemble de classes d'implémentation de commande générique pour construire des éditeurs qui supportent le défaire et refaire entièrement automatique.

- EMF.Codegen : La facilité de génération de code EMF est capable de générer tout ce qui est nécessaire pour construire un éditeur complet pour un modèle EMF. Il comprend une interface graphique à partir de laquelle les options de génération peuvent être spécifiées et les générateurs peuvent être invoqués. La fonction de génération s'appuie sur le composant JDT (Java Development Tooling) d'Eclipse. Trois niveaux de génération de code sont pris en charge :

- Modèle : fournit des interfaces Java et des classes d'implémentation pour toutes les classes du modèle, ainsi qu'une usine et une classe d'implémentation de package (méta-données).

- Adapters : génère des classes d'implémentation (appelées ItemProviders) qui adaptent les classes du modèle pour l'édition et l'affichage.

- Editor : produit un éditeur correctement structuré qui se conforme au style recommandé pour les éditeurs de modèles Eclipse EMF et sert de point de départ pour la personnalisation.
  Donc ; EMF est de permettre la définition d'un métamodèle en se basant sur le méta-métamodèle Ecore. De plus, il permet la conception de modèles conformes à ce métamodèle par la génération du code Java d'un éditeur simple sous forme arborescente. EMF fournit également une API de réflexion la création et la manipulation de modèles de manière programmatique. EMF est également doté d'une API de validation, celle-ci vérifie la conformité syntaxique des modèles créés. Enfin, EMF est doté d'une API de persistance qui, par défaut, génère un fichier XMI (XML Meta data Interchange). Ce fichier contient l'ensemble des descripteurs correspondant au modèle conçu.

L'objectif général de EMF est de proposer un outillage qui permet de passer du modèle au code Java automatiquement. Pour cela le framework s'articule autour d'un modèle (le Core Model).EMF va proposer plusieurs services[44] :

1. La transformation des modèles d'entrées présentés sous diverse formes, en Core Model.

2. La gestion de la persistance du Core Model.

3. La transformation du Core Model en code Java.

EMF peut de base gérer en entrée des modèles présentés sous trois formats : UML, XMI et en code Java Annoté : **UML :** Pour cette option, il existe trois possibilités :

- L'édition directe conformément au méta-modèle Ecore.
- L'importation de modèles UML.
- L'exportation de modèles UML.

XMI : Ce format de fichier standard de l'OMG (Object Management Group)est utilisé conjointement à UML :

- UML se charge de décrire les contenus des modèles.
- XMI se charge de formater ces contenus pour permettre de leur assurer une persistance standardisée.

Java annoté : Une des solutions tentantes pour modéliser les classes qui vont être concrétisées par une application Java est d'utiliser les interfaces Java :

- Elles n'implémentent pas les méthodes : on s'abstrait donc de cette implémentation.
- Les méthodes get/set peuvent être utilisées pour modéliser les attributs.
- Une classe pourra implémenter plusieurs interfaces, ce qui est une manière détournée d'autoriser l'héritage multiple (impossible en Java de classe à classe et possible en UML).

3.9.2.2 Sirius

Sirius est un nouveau projet d'Eclipse qui permet de créer facilement des outils de modélisation graphique. Il s'appuie sur les technologies de modélisation Eclipse, notamment EMF pour la gestion des modèles et GMF pour la représentation graphique. Basé sur une approche par points de vue, Sirius permet d'équiper les équipes qui doivent traiter des architectures complexes sur des domaines spécifiques.
Il est particulièrement adapté aux utilisateurs qui ont défini un DSL (Domain Specific Language) et qui ont besoin de représentations graphiques pour mieux élaborer et analyser un système et améliorer la communication avec les autres membres de l'équipe, les partenaires ou les clients. Toutes les caractéristiques et les comportements de la forme peuvent être facilement configurés avec un minimum de connaissances techniques. Cette description est interprétée dynamiquement pour matérialiser le banc de travail au sein de l'IDE Eclipse. Un atelier de modélisation créé avec Sirius fournit un ensemble d'éditeurs Eclipse qui permettent aux utilisateurs de créer, modifier et visualiser des modèles EMF. [45]
Il existe trois principaux types d'éditeurs : diagrammes, tableaux et arbres. Ces éditeurs sont regroupés par préoccupation afin de fournir des points de vue sur les modèles. [46]

- **Diagrammes :** Avec Sirius, un diagramme affiche les éléments du modèle selon des règles qui ont été définies par le créateur de l'atelier de modélisation. Ces règles définissent quels objets seront affichés et comment ils seront affichés.
- **Tableaux et matrices :** Les modèles peuvent également être édités à l'aide de tableaux et de matrices. Un tableau contient des objets dans une colonne et affiche les caractéristiques dans des autres colonnes. Le style graphique des cellules peut être configuré selon des règles définies par le créateur de
- **L'Arbres :** Le troisième type de représentation fourni par Sirius est l'arbre. Il affiche les objets de manière hiérarchique atelier de modélisation.

Les principes concepts de Sirius

Les cinq principaux concepts sur lesquels repose Sirius sont [47] :

- Le point de vue "viewpoint" : est un élément de base qui est un ensemble logique de spécifications de représentation et de spécifications d'extension de la représentation ;
- Représentation" representation" : est un groupe de constructions graphiques qui représentent les données du domaine. Il décrit également la structure, l'apparence et le comportement des modèles.
- La cartographie "mapping " : dépend fortement du dialecte et identifie un sous-ensemble d'éléments du modèle sémantique qui doivent apparaître dans une représentation et indique comment ils doivent être représentés.
- Style" style" : est utilisé pour configurer l'aspect visuel des éléments.
- Outil "tool " : décrit le mappage des comportements

Les points forts de Sirius :

Sirius simplifie le produit, réduit le temps de conception et augmente rapidement la productivité globale. Temps de conception et augmente rapidement la productivité globale de la construction d'un éditeur graphique spécifique à un domaine. Les forces fondamentales de la plate-forme Sirius sont [47] :

- La fondation sur une norme industrielle ouverte et largement utilisée.
- Standard industriel ouvert et largement utilisé EMF.
- L'adaptabilité à tout DSM compatible EMF.
- Une forte séparation entre les modèles sémantiques et de représentation.
- Le support de différentes représentations de modèles de domaine.
- La facilité d'utilisation et le développement rapide.
- Le haut niveau d'extensibilité.

3.9.2.3 Acceleo

Acceleo est un projet open-source réalisé au sein de la société française Obeo. Qui'il est vise à fournir un générateur de code (transfert de modèles en code) pour les personnes qui souhaitent bénéficier d'une approche dirigée par les modèles afin d'augmenter leur productivité en matière de développement logiciel. Il est sous licence Eclipse Public License (EPL). Acceleo (combiné avec l'approche Model Driven Architecture) est actuellement considéré comme l'un des plus puissants outils de génération de code de sa génération pour le Model Driven Development (MDD) et le Model Driven Engineering (MDE).[48]

L'obtention du code Java correspondant aux modèles grande culture conçus à partir de CMF a été réalisée à l'aide Acceleo. Cette solution, est intégrée au projet M2T (Model To Text) dédié à la génération de code à partir de modèles. Elle se base sur un système de 'templates' appelés modules. Un module principal est défini comme point de départ de la génération de code. Acceleo exploite sous la forme d'un fichier XMI le modèle à transformer ainsi que le métamodèle auquel il se conforme. L'utilisation de modules permet d'obtenir un code particulièrement lisible. Le langage de script utilisé est assez simple d'abord, il est possible de le combiner avec des fonctionnalités plus avancées à l'aide du langage OCL (Object Constraint Language). Ce dernier offre la possibilité d'effectuer des requêtes sur l'ensemble du modèle fourni afin d'obtenir des informations synthétiques. Acceleo gère un cache de requête OCL, évitant ainsi une interrogation supplémentaire du modèle lorsqu'une requête OCL est utilisée plusieurs fois au cours de la génération.[30]

3.10 Langage Maude

Maude est un langage déclaratif de haut niveau et un système à haute performance supportant le calcul logique bithéorique et réécriture pour une large gamme d'applications. Maude a été influencé de manière importante par OBJ3; en particulier, le sous-langage de logique équationnelle de Maude contient essentiellement OBJ3 comme sous-langage. Il est caractérisé par sa simplicité, expressivité, efficacité et par sa performance. Autrement dit, Maude est un langage qui offre peu de constructions syntaxiques et une sémantique bien définie.[49]
Maude n'est pas utilisé seulement pour la simulation des systèmes comme des modèles mathématiques, nous pouvons aussi exploiter la spécification du Maude comme un moyen d'analyse et de vérification, par exemple, pour vérifier que nos modèles vont satisfaire certaines propriétés importantes, ou donner un contre-exemple pour dire qu'une propriété est violée . Parmi les outils utilisés pour la vérification dans Maude, nous trouvons :

- Vérification du modèle par la recherche de l'invariants (Model Checking Invariants Through Search) : est une méthode de vérification formelle largement utilisée spécialement dans le cas des systèmes complexes et des systèmes concurrentiels

- Vérification du modèle en LTL (linear temporal logic Model Checking) : est une logique spécifique qui peut être utilisée comme un formalisme afin de décrire le changement au fil du temps. la vérification du modèle en LTL, prend en charge la vérification instantanée dite à la volée et à état explicite (on-the-fly explicit-state

model checking) des systèmes concurrents exprimés sous forme de théories de réécritures, ce qui offre plus de performances et permet d'élargir la gamme d'applications pouvant être analysées.[50]

Le système Maude possède deux versions à savoir : Full Maude et Core Maude[51] :

- **Core Maude :** Core Maude qui accepte des hiérarchies de modules fonctionnels et de modules système (non paramétrés) avec des syntaxe mixfix définissable par l'utilisateur. Il est implémenté en C++ et se compose de deux parties : le moteur de réécriture, et le moteur d'analyse.et le frontal mixfix.
- **Full Maude :** Une algèbre de modules extensible d'opérations de modules qui peut rendre les modules Maude hautement réutilisables. L'idée de base est que le module META-LEVEL peut être Étendu avec de nouveaux types de données étendant la sorte de modules plats du module à des à des modules structurés et paramétrés et avec de nouvelles fonctions Correspondant à de nouvelles opérations sur les modules tels que l'instanciation de modules paramétrés par vues, l'aplatissement des hiérarchies de modules en modules simples, la désagrégation des modules orientés objet en modules système, etc.

3.10.1 Modules dans Maude

Maude regroupe trois types de modules[52] [53] :

1. **Le module fonctionnel :** Les modules fonctionnels Maude introduits avec la syntaxe **fmod ... endfm** sont des spécifications équationnelles d'appartenance exécutables et leur sémantique est donnée par l'algèbre d'appartenance initiale correspondante dans la classe des algèbres satisfaisant cette spécifications.
   Dans un module fonctionnel, nous pouvons déclarer des tris (au moyen des mots-clés sort(s)) ; des sous-tri entre les tris (subsort) ; des opérateurs (op) pour construire des valeurs de ces tris, en donnant les tris de leurs arguments et de leur résultat, et qui peuvent avoir des attributs tels qu'être associatifs (assoc) ou commutatifs (comm).Un module fonctionnel est déclaré avec la syntaxe suivante : fmod Nom du Module) is Declaration liste d'instructions endfm.
2. **Les Modules systèmes :** Un module système indique une théorie de réécriture. Une théorie dont laquelle on trouve des sortes, des opérateurs (peut-être avec des arguments gelés), et peut avoir plusieurs types de déclarations : des équations, et des règles, dont toutes peuvent être conditionnelles. Un module système est déclaré dans Maude en employant les mots-clés :
   mod Nom du Module) is Declaration liste d'instructions endm
3. **Modules orientés-objet :** Les systèmes orientés objet concurrents peuvent être définis au moyen de modules orientés objet - introduits par le mot-clé omod...endom - utilisant une syntaxe plus pratique que celle des modules de système, car elle suppose la connaissance des entités de base, telles que les objets, les messages et les configurations, et supporte les distinctions linguistiques appropriées au cas orienté

objet. Un module orientés objet est déclaré avec la syntaxe suivante :
omod < Nom du Module > is < Déclarations et instructions > endom

3.10.2 Avantages de Maude :

Maude est basé principalement sur les différents avantages offerts, parmi lesquels nous mentionnons [50] :

- La simplicité de la syntaxe de Maude et la précision de la sémantique.
- Maude inclut plusieurs outils pour faciliter la spécification et la vérification des Systèmes.
- Maude est un métalangage et aussi un méta-tool, ce qui lui donne une haute extensibilité et une grande souplesse.
- Maude est un langage de spécification formelle ce qui lui permet de bénéficier des avantages de cette approche.

3.11 Conclusion

Dans ce chapitre nous avons présenté un état de l'art sur le développement logiciel dirigé par les modèles. Nous avons plus particulièrement mis l'accent dans cette étude sur les concepts, langages de modélisation, la transformation de modèles.

Ce chapitre a présenté les outils de modélisation offerts par Eclipse qui sont très puissants et facilitent énormément la tache de modélisation durant le développement orienté modèle des applications qui nous permettent de mettre en œuvre et d'examiner notre approche. Jusqu'à présent, nous avons défini un cadre réalisation de cette proposition, logique de transformation de modèle, ainsi que les règles Développé pour mettre en œuvre ces transformations et la solution technique qui permet de générer le code source BPMN final et leur vérification.

Chapitre 4

Développement d'un Environnement de Modélisation Graphique pour les Applications IdO

4.1 Introduction

Dans ce chapitre, nous allons proposer une approche et un outil d'environnement pour modéliser les applications IdO en utilisant le standard BPMN2.0.
Notre approche est basée sur la metamodelisation, nous avons enrichie un sous ensemble des éléments de BPMN en ajoutant des éléments liés à l'IdO pour adapter le BPMN dans le cadre de modelisation des application IdO. Pour l'implémentation de notre meta-model nous avons utilisé Eclipse Modeling Framework (voir le chapitre 3, section 3.9.2.1). En suite nous avons créer un éditeur graphique sous Sirius pour la visualisation des modèles crées dans notre projet. En fin pour l'analyse et la vérification, on traduit modèles à des spécifications Maude équivalentes. Pour réaliser la transformation nous avons utilise Acceleo (voir le chapitre3, section 3.9.2.3).

4.2 Description de notre approche

Notre approche est composée en 4 phase :

1. Métamodèlisation des applications IdO en utilisant de BPMN.
2. La visualisation de l'environnement graphique sous Sirius.
3. La création d'un code Acceleo pour la transformation des modèles crées dans l'environnement Sirius a Maude.
4. L'exécution du code généré dans Maude.

4.3 Métamodelisation des application IdO avec BPMN

Nous avons utilisé EMF pour l'implantation de notre méta-modèle, qui peut créer des classes, des associations et des relations d'héritage sur ce dernier. Nous commençons d'abord par la création du projet EMF (Eclipse Modeling Framwork –> Ecore Modeling Project), comme le montre la figure suivante :

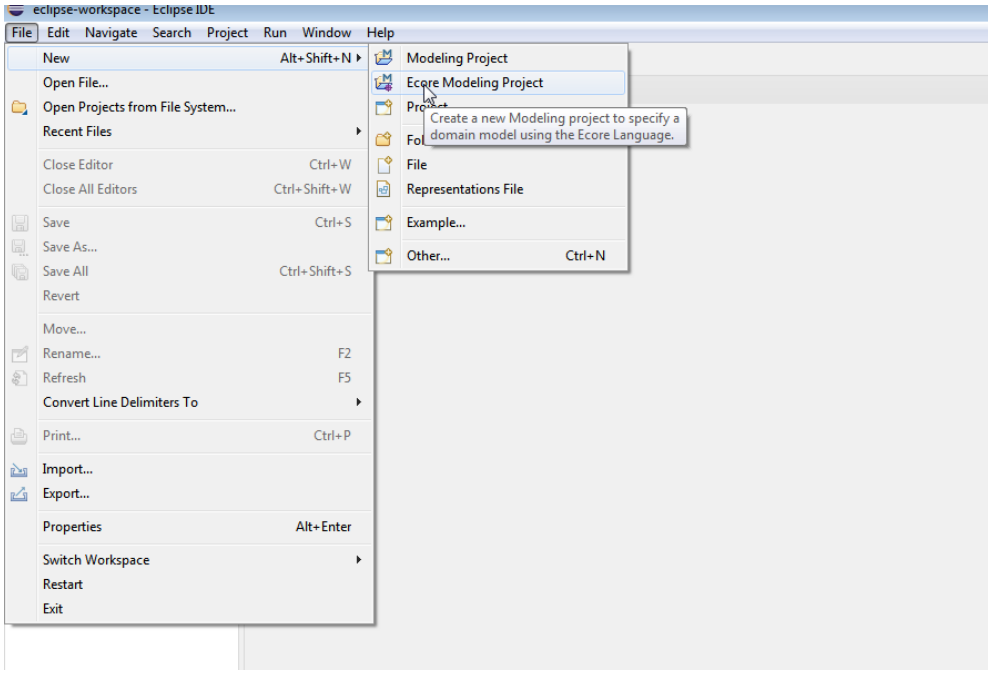


Fig. 4.1 : La création d'un projet EMF

Notre méta modèle est une extension du méta modèle BPMN 2.0, dans lequel nous avons ajouté des éléments lieé a l'IdO telque la tache Sensor et Action pour l'adaptation de BPMN a la modélisationn de l'IdO.
Notre métamodàle est composé de 37 classes parmi elles 7 classes abstraites et une énumération de classe ; reliées entre elles par 11 associations.

- **Les relations des héritiers de méta-modéle sont les suivantes :**
  - Les classes : "task script , action activity , task service , task send message , sensor activity , task rule management , task reception message , task manual , task user " héritent de classe d'activité ;
  - Les classes :"iot apperiel et processus" héritent de la classe participan ;
  - Les classes "activité , événement et passerelle " héritent de classe node ;
  - Les classes :" évènement debut abstract, évènemen intermediaiare abstract ,événements fin abstract" héritent de classe évènement ;
  - Les classes "évènement debut message,évènement debut minuterie" héritent de classe évènement debut ;
  - Les classes :"évènement intermediaiare,évènement intermediaiare minuterie,évènement intermediaiare envoi message,évènement intermediaiare reception message" héritent de classe évènemen intermediaiare ;
  - Les classes :"évènement fin, évènement fin message" héritent de classe événements fin
  - Les classes :"inclusif, exclusif, parallèle, complexe" héritent de classe node ;
  - La classe "intéraction" hérite de classe node.

- **Les relation de notre métamodele :**
  l'arc flux sequence permet d'aller :
  - D'une activité vers une autre
  - D'un événement vers un autre événement.
  - D'une passerelle vers une autre passerelle.
  - D'une activité vers une passerelle et vise vers ça.
  - D'une activité vers un événement et vice vers ça.
  - D'une passerelle vers un événement et vise vers ça.

  L'arc flow de message Permet d'aller :
  - D'un événement vers un autre.
  - D'une activité vers une autre.
  - D'une activité vers un événement et vise vers ça. .
- Un flux de séquence contient une seule source (activité, événement ou paserelle) et une seule cible (activité, événement ou paserelle).
- Un flux de messages n'a qu'une seule source (qui peut être une activité ou un événement) et une seule destination (qui peut être une activité ou un événement).

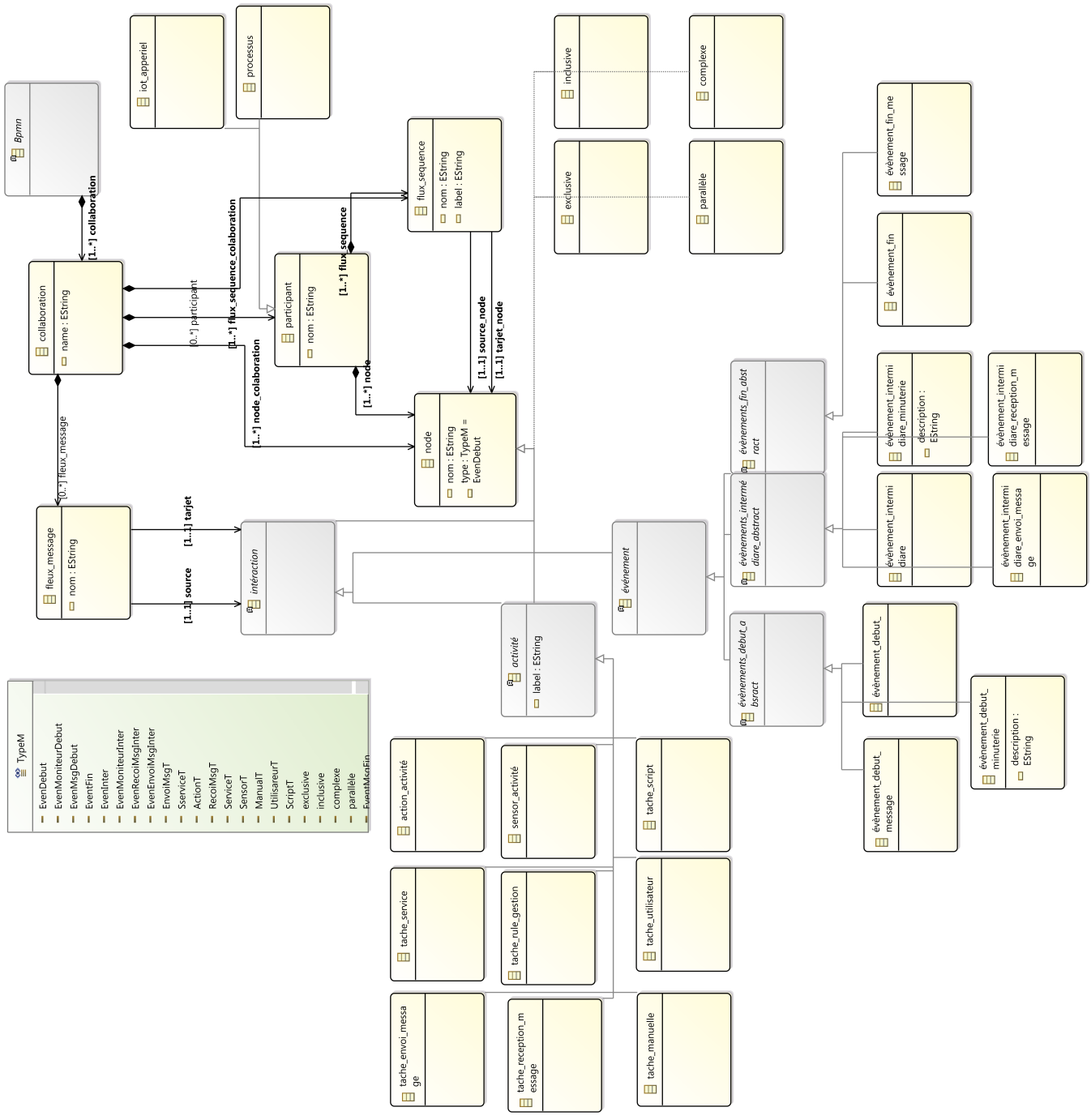


Fig. 4.2 : Extension du méta-modèle BPMN 2.0 pour IdO sous Eclipse Modeling Tools

4.3.1 Composants du Méta-modèle

4.3.1.1 Classes

Le méta-modèle se compose de 37 classes. Le tableau ci-dessous représente une description de ces classes :

Classes	Description
bpmn	C'est une classe abstraite qui représente la classe mère du modèle BPMN
collaboration	Une classe qui peut être instanciée pour créer un modèle bpmn qui peut ne pas avoir des participant Contient la attribut : Name pour Spécifie le nom du collaboration.
participant	Cette classe représente le participant qui peut être un processus ou un appareil IdO. Elle Contient au moins un noeud. Contient la attribut : Nom pour spécifier le nom du l'instance de cette classe.
processus	Une sous classe de la classe participant, elle représente le processus central.
iot apperiel	Une sous classe de la classe participant elle représente les dispositifs IdO.
node	Elle représente les noeuds du diagramme BPMN.(activité,événement,passerelle)Elle a comme attribut : nom : pour spécifier le nom de l'instance et type pour spécifier le type de l'instance. cet attribut "type" est hérité par la classe "évènement de debut" à l'instantiation, nous choisissons son type qui est "événement de debut" .le type de l'attribut "type" est spécifié dans la classe enumeration TypeM
activité	Une sous classe de la classe node, elle représente les activités du processus dans le diagramme BPMN contient l'attribut : label pour spécifier le nom à afficher dans le diagramme.
tache script	Une sous classe de la classe activité, elle représente la tâche script du processus dans le diagramme BPMN.
tache utilisateur	Une sous classe de la classe activité, elle représente la tache utilisateur du processus dans le diagramme BPMN.
tache manuelle	Une sous classe de la classe activité, elle représente la tâche manuelle du processus dans le diagramme BPMN.

tache rule gestion	Une sous classe de la classe activité, elle représente la tâche règle de gestion du processus dans le diagramme BPMN.
tache réception message	Une sous classe de la classe activité, elle représente la tâche de réception de message du processus dans le diagramme BPMN.
tache envoie message	Une sous classe de la classe activité, elle représente la tâches d'envoie de message du processus dans le diagramme BPMN.
tache service	Une sous classe de la classe activité, elle représente la tâches service du processus dans le diagramme BPMN
action activité	Une sous classe de la classe activité, elle représente la tâches Actionnement dans l'extension de BPMN
sensor activité	Une sous classe de la classe activité, elle représente la tâches Actionnement dans l'extension de BPMN
événement	Une sous classe de la classe Node, elle représente les évènements dans le diagramme BPMN.
événement debut abstract	Une sous classe abstract de la classe évènement,
événement intermédiare abstract	Une sous classe abstract de la classe évènement,
événement fin abstract	Une sous classe abstract de la classe évènement,
événement debut	Une sous classe de la classe événement debut abstract,elle représente les évènements début dans le diagramme BPMN
événement debut message	Une sous classe de la classe événement debut abstract,elle représente les évènements début de réception message dans le diagramme BPMN
événement debut minuterie	Une sous classe de la classe événement debut abstract,elle représente les évènements début minuterie dans le diagramme BPMN.Elle a comme attribut : description : qui indique temps spécifique pour se déclencher .
événement intermédiare	Une sous classe de la classe évènement intermédiare abstract.elle représente les évènements intermédiaires dans le diagramme BPMN.
événement intermédiare minuterie	Une sous classe de la classe évènement intermédiare abstract.elle représente les évènements intermédiaires minuterie dans le diagramme BPMN.

événement intermédiare envoi message	Une sous classe de la classe évènement intermédiaire abstract.elle représente les évènements intermédiaires d'emission de message dans le diagramme BPMN.
événement intermédiare reception message	Une sous classe de la classe évènement intermédiaire abstract.elle représente les évènements intermédiaires réception de message dans le diagramme BPMN.
événement fin	Une sous classe de la classe évènement fin abstract.elle représente les évènements fin dans le diagramme BPMN.
événement fin message	Une sous classe de la classe évènement fin abstract.elle représente les évènements fin d'emission de message dans le diagramme BPMN.
fleux sequence	elle représente le flux de séquence de bpmn.Contient la attribut : label pour Spécifié le nom à afficher dans le diagramme et nom pour Spécifie le nom d'instance .
exclusive	Une sous classe de la classe node, elle représente la passerelle exclusiver
inclusive	Une sous classe de la classe node, elle représente la passerelle inclusive
complex	Une sous classe de la classe node, elle représente la passerelle complex
parallèle	une sous classe de la classe node, elle représente la passerelle parallèle
fleux message	Elle représente le flux de message .Elle a comme attribut : nom : qui indique le nom de l'instance.
interaction	Une classe abstraite pour effectuer des interactions entre les flux de messages et «les activité les événement».
TypeM	Un classe enumeration , On choisit le type de nœud directement dans la liste de cette classe.

Tab. 4.2 : Les classes du Méta-modèle

4.3.1.2 Associations

1. Composition :Le méta-modèle se compose de 7 association de composition. Le tableau ci-dessous représente une description de ces Composition :

Nom de la composition	Description	Cardinalités
colaboration	composition De la classe bpmn vers la classe colaboration	1..*

flux sequence colaboration	composition De la classe colaboration vers la classe flux_sequence	1..*
node colabiration	composition De la classe colaboration vers la classe node	1..*
fleux_message	composition De la classe colaboration vers la classe flux_message	0..*
node	composition De la classe participant vers la classe node	1..*
participant	composition De la classe colaboration vers la classe participant	0..*
fleux_sequence	composition De la classe participant vers la classe fleux_sequence	1..*

TAB. 4.4 : Les Composition du Méta-modèle

2. Référence : Le méta-modèle se compose de 4 association de référence. Le tableau ci-dessous représente une description de ces référence :

Nom de la referene	Description	Cardinalités
source node	De la classe flux sequence vers la classe node	1..1
tarjet node	De la classe flux sequence vers la classe node	1..1
source	De la classe flux message vers la classe interaction	1..1
tarjet	De la classe flux message vers la classe interaction	1..1

TAB. 4.6 : Les référence du Méta-modèle

3. Héritages : Le méta-modèle se héritage. Le tableau ci-dessous représente une description de ces héritages :

la classe mère	les class fils
activité	tache script , action activity , tache service , tache send message ,sensor activity , tache rule management , tache reception message , tache manual , tache user
participant	iot apperiel et processus
node	activité , événement et intéraction et inclusif, exclusif, parallèle, complex
évènemen	évènement debut abstract, évènemen intermediare abstract,événements fin abstract
évènement debut abstract	évènement debut message,évènement debut minuterie

évènemen Intermediare abstract	évènement intermidiare,évènement intermidiare minuterie,évènement intermidiare envoi message,évènement intermidiare reception message
événements fin abstract	évènement fin, évènement fin messag

TAB. 4.8 : L'héritage du Méta-modèle

4.3.2 Génération de code pour le méta-modèle crée

La génération de code et de l'diteur arborescent généré automatiquement à partir de la description du méta-modèle proposé est réalisée comme nous le montre la figure.4.3

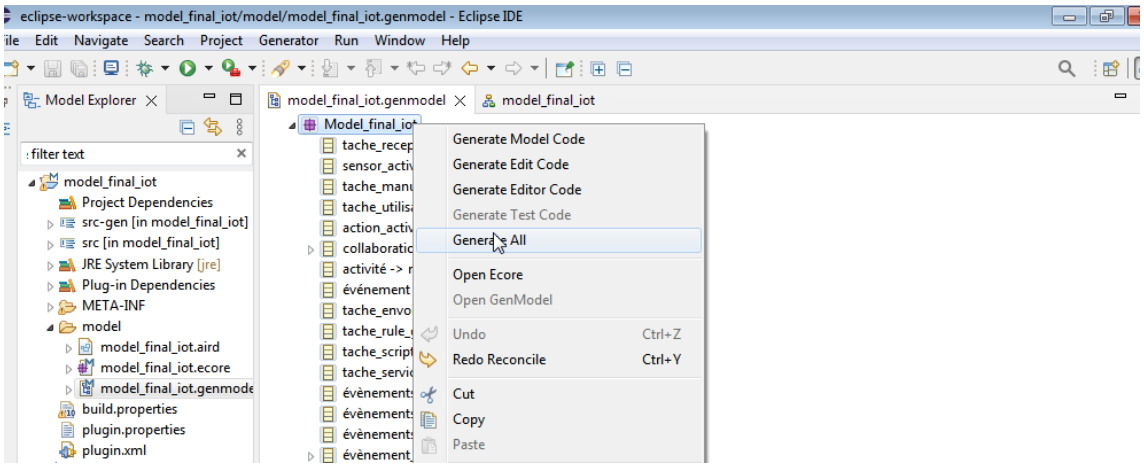


FIG. 4.3 : La génération du fichier model_final_iot.genmodel.

Deux nouveaux projets sont générés comme illustré dans la figure 4.4. Le projet « model_final_iot.edit » et « model_final_iot.editor »

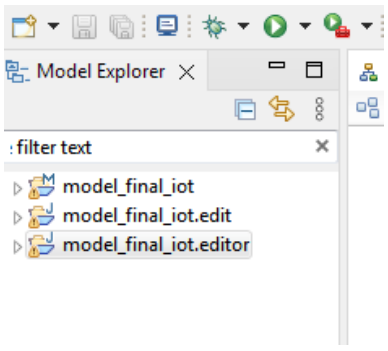


FIG. 4.4 : Le résultat de la génération du fichier model_final_iot.genmodel.

4.4 Créations de l'environnement Graphique sous Sirius

4.4.1 Creation de projet Sirius

La création des éditeurs de Sirius se fait au travers des projets que l'on nomme « Viewpoint Specification Project ». Via le menu « New > Project > Viewpoint Specification Project »,comme le montre la figure 4.7. nous créont un projet « test.projects.design ».

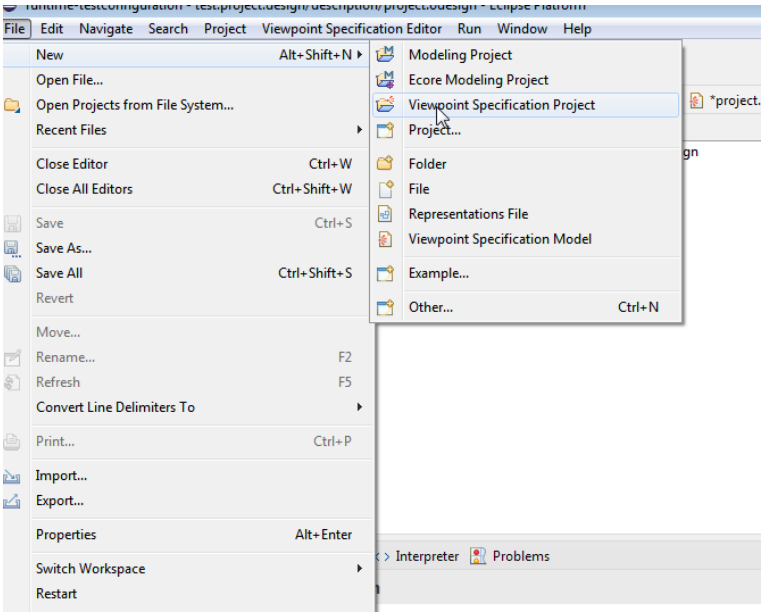


FIG. 4.5 : Création d'un projet de type « Viewpoint Specification Project »

Nous devons associer ensuite une représentation à notre viewpoint afin d'indiquer à Sirius quel type d'éditeur nous voulons créer. l'éditeur est possible de type arbre, tableau, de diagramme ou encore diagramme de séquence , dans notre project nous avons utilisé l'éditeur de diagramme.

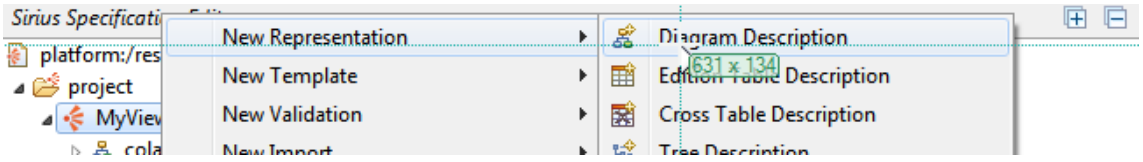


FIG. 4.6 : Le choix de type de représentation

Ensuite nous avons nommé le graphe "collaboration" ; c'est une instance de la classe «colaboration» de notre modèle.

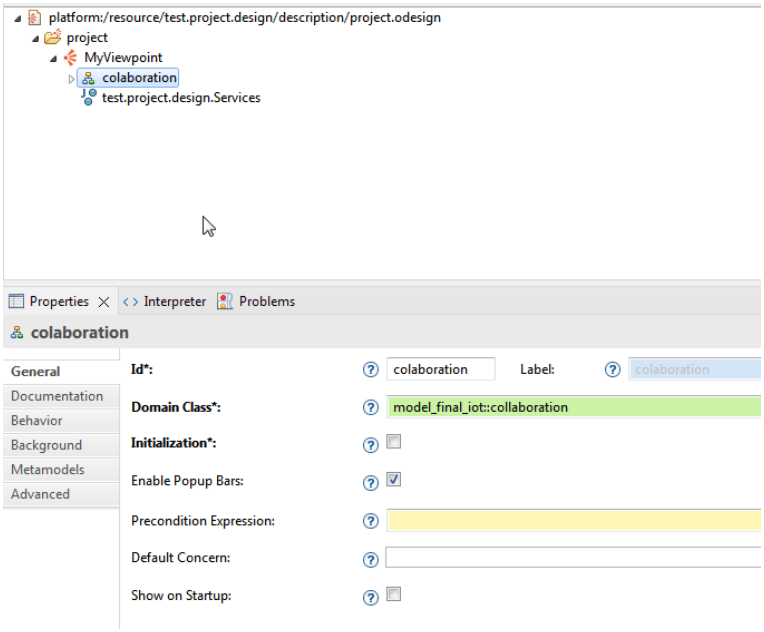


FIG. 4.7 : Créer un éditeur de diagramme

Par un clic gauche sur «default», nous choisissons «new Diagram element» puis node, c'est ainsi que nous créons les éléments du modèle puis nous choisissons la classe de domaine et «id» pour chaque élément et ajoutons l'image de l'élément.

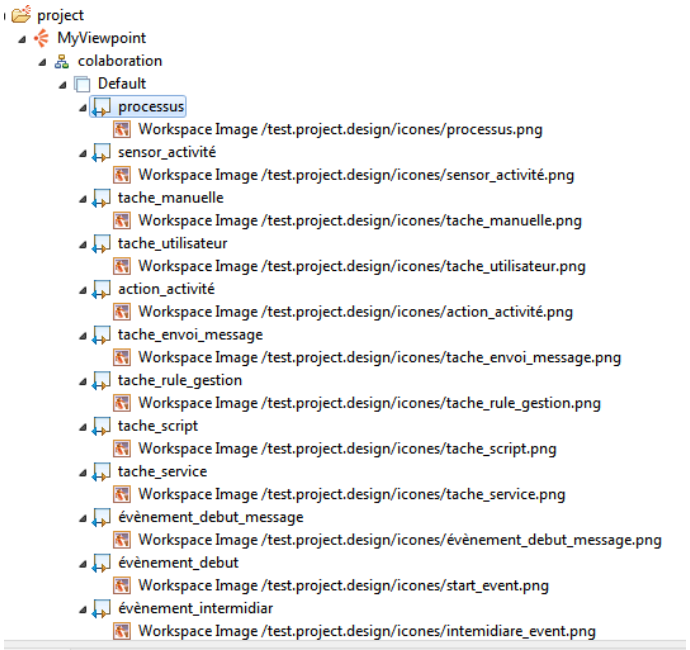


FIG. 4.8 : Création les éléments de notre modèle par ces images »

Pour la séquence flow et le flow de messages, un clic gauche sur «default», on choisit «new Diagram element» puis relation based Edge.

Nous écrivons l'identifiant «id» et choisissons la classe de domaine. Pour définir la source mapping , choisissons les éléments qui peuvent être la source du flux de séquence : tous les éléments qui héritent de l'activité, de passerelle et de la classe d'événement, à

l'exception des événements de fin.

Définit comme «tarjet mapping »tous les éléments peuvent être une cible : tous les éléments qui héritent de l'activité, de la porte et de l'événement de la classe, à l'exception des événement de début.

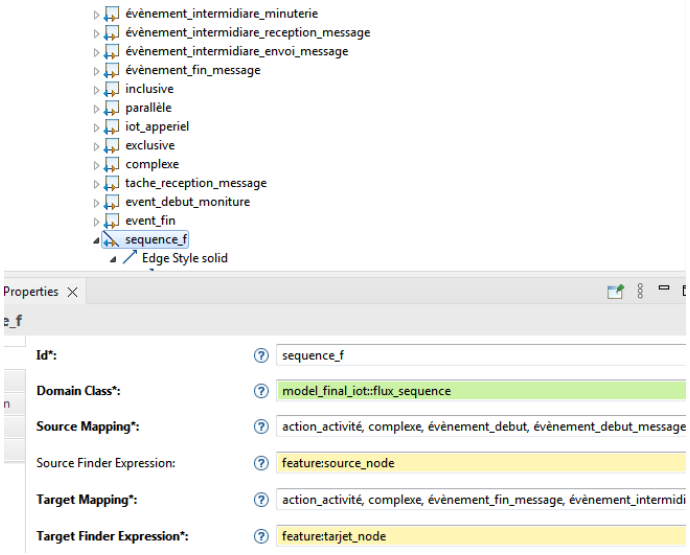


Fig. 4.9 : Création séquence flow »

Pour spécifier la source mapping et target mapping du message de flux : tous les éléments qui héritent des classes Activité et événement uniquement.

4.4.2 Création de la palette

En commençant par un clic gauche sur «default», choisissez «new tool», puis «section» nous avons créé 4 sections Comme le montre l'image 4.10:

- la section événement : contient les événements ;
- section activité : contient les activités et tâches ; section
- pasaralle : contient les passerelles ;
- la section générale : contient les flux de séquence, les flux de messages et les processus.

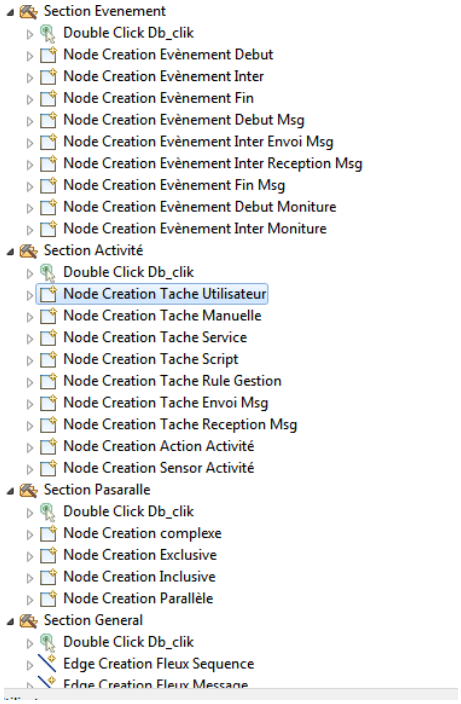


Fig. 4.10 : Création de la palette »

Ensuite, nous remplissons les informations pour chaque élément comme indiqué dans l'image.

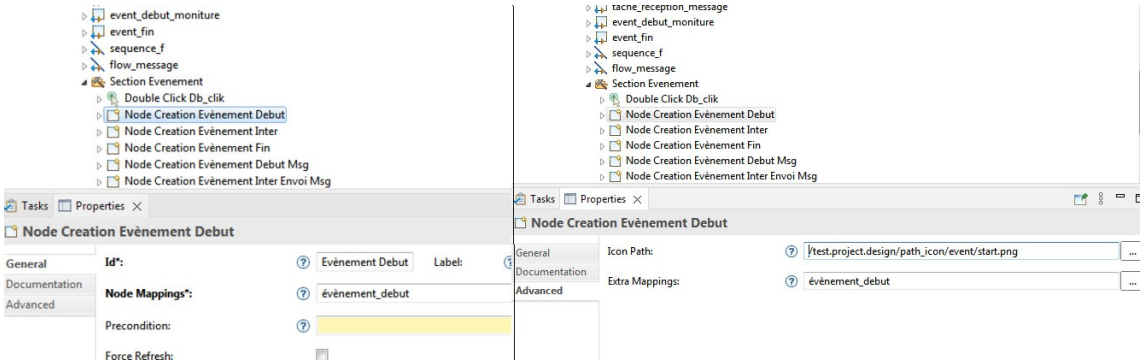


Fig. 4.11 : Création de la palette »

voici la palette à la fin

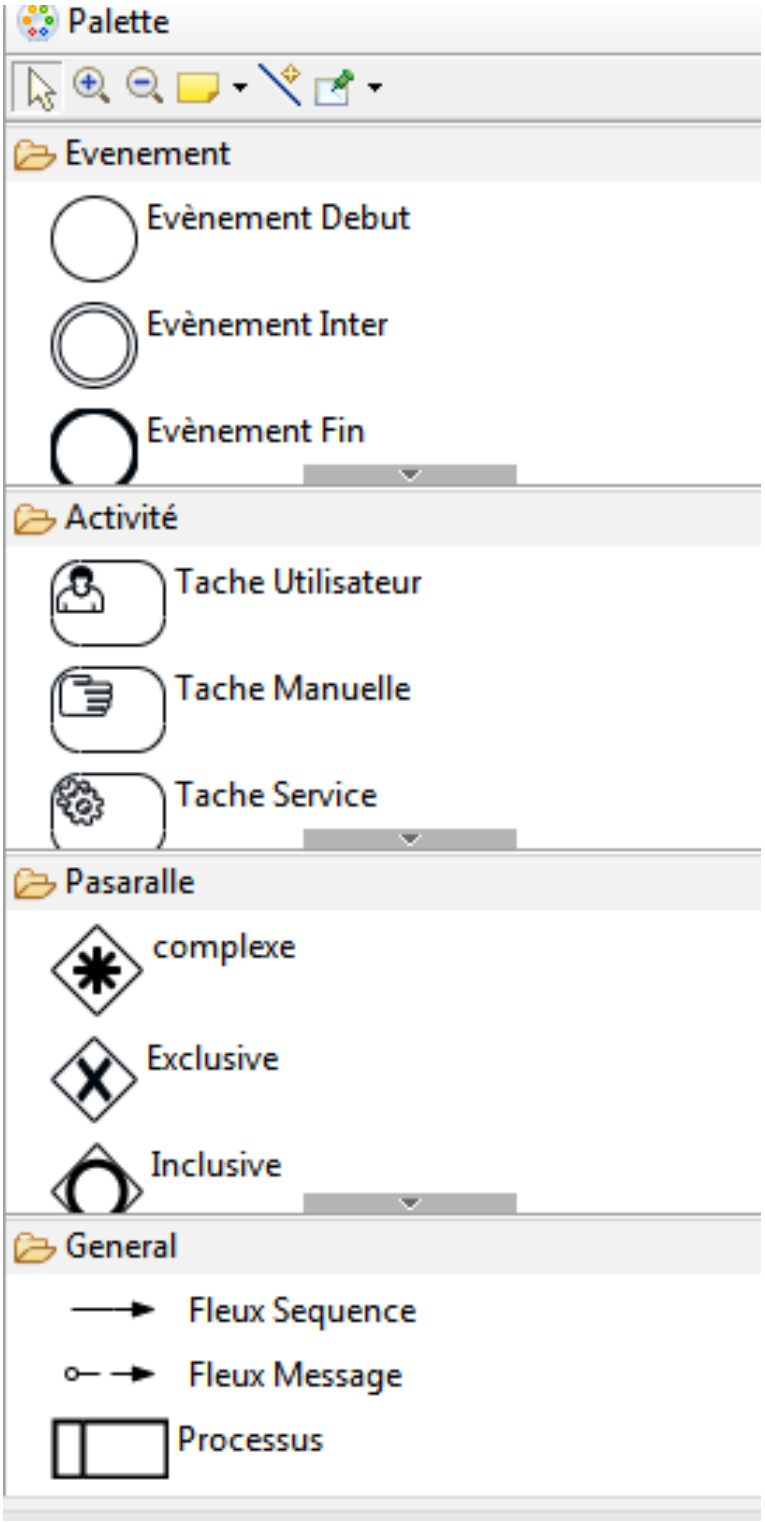


FIG. 4.12 : Création de la palette

4.5 Transformation des modèle en Maude En utilisant Acceleo

4.5.1 Définition du modèle fonctionnel de BPMN_IdO en Maude

On commence par nommer le code par basic -Bpmn-IdO ; puis nous définissons les éléments de nos modèle : les activités, les passerelles, évènement de debut, intermidiare et fin Comme le montre l'image 4.13 ;

Commençons par évènement de debut :

"sorts DebutEvent DebuteventType DebutEventNom" :Pour définir les éléments de évènement de début dont nous avons besoin ;nous avons besoin d'un type que nous déclarons avec le mot " DebuteventType" et leur nom par "DebutEventNom".

Ensuite, nous devons déclarer la liste des types d'événements de début que nous déclarons avec cette ligne : **"ops EvenDebut EvenMsgDebut EvenMoniteurDebut : -> DebutEventType "**

enfin :**"op [_ :_] : DebutEventNom DebutEventType -> DebutEvent"** : pour que en va conné l'évenement de debut par le nom et le type.

Tous les autres éléments sont faits de la même manière et à la fin nous définissons ces éléments avec sort CurrentNodes . subsorts DebutEvent FinEvent InterEvent Activité passerelle < CurrentNodes .

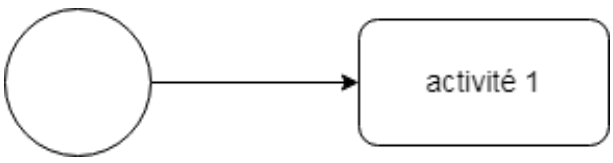
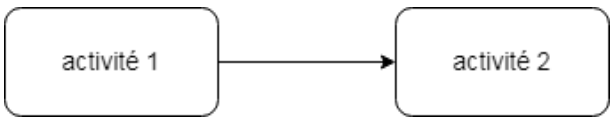
Remarque : les lignes qui commencent par des *** sont des commentaires pour mieux comprendre le code.

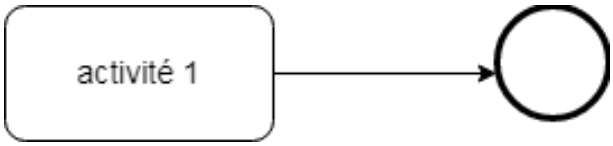
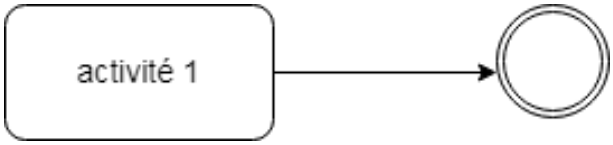
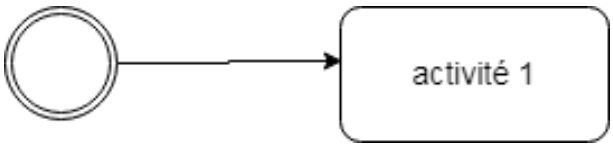
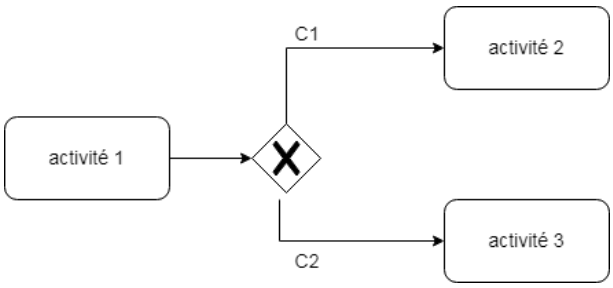
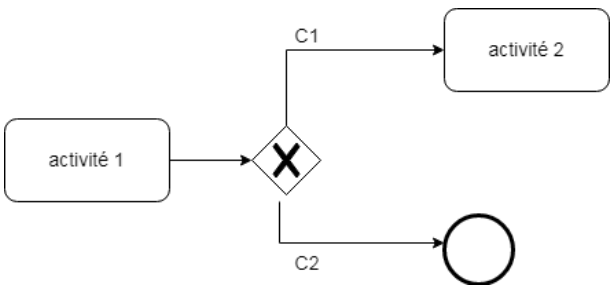
```
fmod Basic-BPMN-IdO is
protecting BOOL .

*** definition des evenement de debut
sorts DebutEvent DebutEventType DebutEventNom .
*** list des type des evenement de debut
ops EvenDebut EvenMsgDebut EvenMoniteurDebut : -> DebutEventType .
op [_:_]  : DebutEventNom DebutEventType -> DebutEvent . ***represent un evenement de debut dans le diagrame .

*** definition des evenement de fin
sorts FinEvent FinEventType FinEventNom .
*** list des type des evenement de fin
ops EventFin EventMsgFin : -> FinEventType .
op [_:_]  : FinEventNom FinEventType -> FinEvent . ***represent un evenement de fin dans le diagrame .

*** definition des evenement intermediare
sorts InterEvent InterEventType InterEventNom .
*** list des evenement intemedaite type
ops EvenInter EvenMoniteurInter EvenRecoiMsgInter EvenEnvoiMsgInter : -> InterEventType .
op [_:_]  : InterEventNom InterEventType -> InterEvent . ***represente un evenement intermediare dan le diagrame .

**********definition des activité
sorts Activité ActivitéType ActivitéNom .
*** list of activity type
ops EnvoiMsgT ServiceT ActionT RecoiMsgT GestionT SensorT ManualT UtilisareurT ScriptT : -> ActivitéType .
op [_:_]  : ActivitéNom ActivitéType -> Activité . ***represent an activite in the diagram .

**********definition des passerelle
sorts passerelle passerelleType passerelleNom .
*** list of passerelle type
ops exclusive inclusive complexe parallèle : -> passerelleType .
op [_:_]  : passerelleNom passerelleType -> passerelle . ***represent an passerelle in the diagram .

sort CurrentNodes .
subsorts DebutEvent FinEvent InterEvent Activité passerelle < CurrentNodes .
op __ : CurrentNodes CurrentNodes -> CurrentNodes [ctor config assoc comm id: null] .
***represent current Nodes of an instance.
op null : -> CurrentNodes [ctor] .

endfm
```

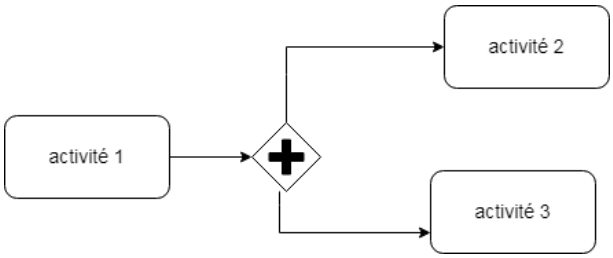
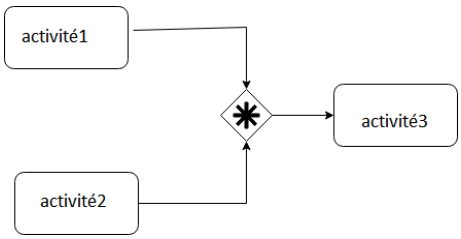
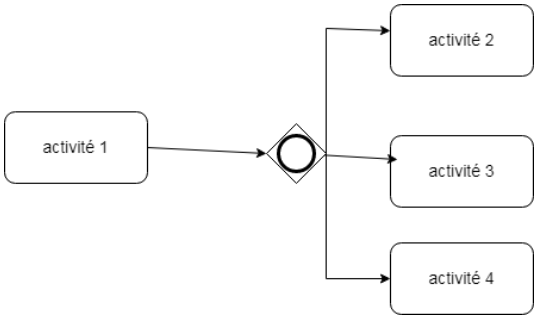
Fig. 4.13 : Unité fonctionnelle de base BPMN_IdO

4.5.2 Définition des règles de transformation

les règles de transformation de Flux de séquence :

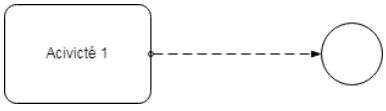
Structures BPMN	Règles de réécriture Maude correspondantes
	rl [debut2activité] : [debut : debutEvent] => [activité1:activité].
	rl [Activité] : [activité1: Activité] => [activité2:Activité].

	rl [activité2fin] : [activité1: activité] => [fin : finEvent].
	rl [activité2Event] : [activité1: activité] => [event :InterEvent].
	rl [inter2activité] : [event : InterEvent] => [activité1:activité].
	rl [SF] : [activité1: activité]=>; [exclisiveG : exclusive]. rl [C1] : [exclisiveG : exclusive]=> [activité2: activité]. rl [C2] : [exclisiveG : exclusive] => [activité3: activité].
	rl [SF] : [activité1: activité]=> [exclisiveG : exclusive]. rl [C1] : [exclisiveG : exclusive]=> [activité2: activité]. rl [C2] : [exclisiveG : exclusive] => [event : finEvent].

	rl [SF] : [activité1: activité]=> [parallelG : parallel]. rl [parallel] : [parallelG : parallel] => [activité2: activité] [activité3: activité].
	rl [complex] : [activité1: activité] [activité2: activité] => [complexG : complex]. rl [SF] : [complexG : complex] => [activité3: activité].
	rl [SF] : [activité1: activité]=> [inxclusiveG : exclusive]. rl [inxclusiveG] : [inxclusiveG : inxclusive] => [activité2: activité] [activité3: activité]. rl [inxclusiveG] : [inxclusiveG : inxclusive] => [activité2: activité]. rl [inxclusiveG] : [inxclusiveG : inxclusive] => [activité3: activité]. rl [inxclusiveG] : [inxclusiveG : inxclusive] => [activité4: activité].

Tab. 4.10 : les règles de transformation en Maude (Sequence Flows)

les règles de transformation de flux de message :

Structures BPMN	Règles de réécriture Maude correspondantes
	rl [activité2event] : [activité1: activité] => [event : debutEvent] .

TAB. 4.12 : les règles de transformation en Maude (flux de message)

4.5.3 Creation de projet Acceleo pour les génération de code Maude

Alors en commence par la création d'un projet de type acccleo par le menu « New > Acceleo Project », en choisissant un nom le projet puis clique next puis nous remplissons c'est information affichées dans l'image 4.17.

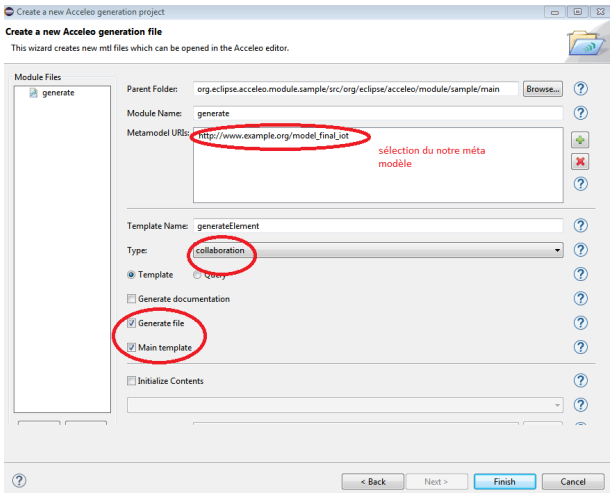


FIG. 4.14 : création de projet de type acccleo

Ensuite, nous écrivons le code en generate.mtl , nous divisons le code en trois parties pour être plus clair.

- partie 1

```
[comment encoding = UTF-8 /]
[module generate('http://www.example.org/model_final_iot')]


[template public generateElement(acollaboration : collaboration)]
[comment @main/]
[file (acollaboration.name.concat('.maude'), false, 'UTF-8')]
in BPMN_MAUDE.maude
mod [acollaboration.name.toUpper()/] is
including Basic-BPMN-IdO .
ops [for (adEvent : évènements_debut_absract | acollaboration.eContents(évènements_debut_absract))]
[adEvent.nom/] [/for]: -> DebutEventNom .

ops [for (adEvent : évènements_fin_abstract | acollaboration.eContents(évènements_fin_abstract))]
[adEvent.nom/] [/for]: -> FinEventNom .

ops [for (adEvent : activité | acollaboration.eContents(activité))][adEvent.nom/] [/for]
: -> ActivitéNom .

ops [for (adEvent : passerelle | acollaboration.eContents(passerelle))][adEvent.nom/] [/for]
: -> passerelleNom .
```

FIG. 4.15 : code acceleo partie 1

- partie 2 :

```
[for (aSequenceFlow : flux_sequence | acollaboration.eContents(flux_sequence))]
[if (aSequenceFlow.source_node.eClass().name = 'exclusive')]
rl ['['/][aSequenceFlow.source_node.eClass().name/]] : ['['/][aSequenceFlow.source_node.nom/] : [aSequenceFlow.source_node.type/]]
 => ['['/][aSequenceFlow.tarjet_node.nom/] : [aSequenceFlow.tarjet_node.type/]] .
[/if]
[if (aSequenceFlow.source_node.eClass().name = 'inclusive')]
rl ['['/][aSequenceFlow.source_node.eClass().name/]] : ['['/][aSequenceFlow.source_node.nom/] : [aSequenceFlow.source_node.type/]]
 => ['['/][aSequenceFlow.tarjet_node.nom/] : [aSequenceFlow.tarjet_node.type/]] .
[/if]
[if (aSequenceFlow.source_node.eClass().name = 'parallèle')]
rl ['['/][aSequenceFlow.source_node.eClass().name/]] : ['['/][aSequenceFlow.source_node.nom/] : [aSequenceFlow.source_node.type/]]
 => [for (aSequenceFlow : flux_sequence | acollaboration.eContents(flux_sequence))]
[if (aSequenceFlow.source_node.eClass().name = 'parallèle')]['['/][aSequenceFlow.tarjet_node.nom/] : [aSequenceFlow.tarjet_node.ty
[/if][/for] .
[/if]
[if (aSequenceFlow.tarjet_node.eClass().name = 'complexe')]
rl ['['/][aSequenceFlow.tarjet_node.eClass().name/]] :
[for (aSequenceFlow : flux_sequence |acollaboration.eContents(flux_sequence))]
[if (aSequenceFlow.tarjet_node.eClass().name ='complexe')]['['/][aSequenceFlow.source_node.nom/] :[aSequenceFlow.source_node.type/
 [/if][/for]=> ['['/][aSequenceFlow.tarjet_node.nom/] : [aSequenceFlow.tarjet_node.type/]] .
[/if]
[if (aSequenceFlow.source_node.eClass().name <> 'parallèle' and
aSequenceFlow.source_node.eClass().name <> 'exclusive' and
aSequenceFlow.tarjet_node.eClass().name <> 'complexe' and aSequenceFlow.source_node.eClass().name <> 'inclusive' )]
rl ['['/][aSequenceFlow.nom/]] : ['['/][aSequenceFlow.source_node.nom/] : [aSequenceFlow.source_node.type/]] => ['['/][aSequenceFl
[/if]
[/for]
```

FIG. 4.16 : code acceleo partie 2

On fait une boucle pour parcourir tous les flux de séquence dans le diagramme afin de faire les spécifications en Maude, nous déclarons une variable de type fleux de séquence "aSequenceFlow" car nous allons l'utiliser à l'intérieur de la boucle pour accéder aux attributs nom, type, source et tarjet (destination) de fleux de séquence ; on distingue 5 cas dans la boucle :

- Cas 1: Si la source de flux est une passerelle "exclusive" alors nous avons choisi le nom de rôle (rl) par le nom de la source de flux qui est "exclusive" nous définissons la partie droite du rôle par le nom puis le type de source de flux pour la partie gauche nous définissons par le nom et le type de destination pour obtenir un résultat comme celui-ci :

  rl [exclusive] : [f3 : exclusive] => [E6 : évènementFin] .
- Cas 2: Si la source de flux est une passerelle "inclusive" le même déclarations de "exclusive".
- Cas 3: Si la source du flux est une passerelle "parallèle" alors nous avons choisi le nom de rôle (rl) par le nom de la source du flux qui est parallèle nous définissons la partie droite du rôle par le nom puis le type de source du flux pour la partie gauche nous définissons la liste de nom et le type de destinations en utilise la boucle «for» pour passer à tous les flux de diagramme.pour obtenir un tel résultat rl [parallèle] : [f1 : parallèle] => [fréquenceCardiaque : tacheEnvoiMessage] [axymétriePouls : tacheEnvoiMessage] [tensionArtérielle : tacheEnvoiMessage] .
- Cas 4: Si la source du flux est une passerelle "complexe" alors nous avons choisi le nom de rôle (rl) par le nom de la destination du flux qui est complexe on distingue la partie droite du rôle la liste des nom et type de source en utilisant la boucle for. La partie gauche est définie par nom et type de destination pour obtenir un résultat comme celui-ci :

  rl [complexe] :[fréquenceCardiaque : tacheEnvoiMessage] [axymétriePouls : tacheEnvoiMessage] [tensionArtérielle : tacheEnvoiMessage]=> [f2 : complexe].
- Cas 5: Si la source du flux de séquence n'est pas (exclusive, inclusive, parallèle) la destination n'est pas (complexe) alors le nom de rôle (rl) est le nom du flux de séquence. Nous définissons la partie droite du rôle par le nom et le type de source. La partie gauche est définie par nom et type de destination.

- partie 3

```
[for (aMessageFlow : fleux_message | acollaboration.eContents(fleux_message))]
rl ['['/][aMessageFlow.nom/]] : ['['/][aMessageFlow.source.nom/] : [aMessageFlow.source.type/]] => ['['/][aMessageFlow.tarjet.nom/
[/for]
endm
rew ['['/] E1 : EvenMoniteurDebut ] ['['/]E5 : EvenMoniteurDebut] ['['/]E3 : EvenMoniteurDebut] ['['/]E8 : EvenMoniteurDebut] ['['
[/file]
[/template]
```

FIG. 4.17 : code acceleo partie 3

Pour générer le code Maude des flux de messages : une boucle pour parcourir tous les flux de messages de diagramme puis nommer le rôle (rl) par le nom des flux de messages on définit la partie droite du rôle par le nom et le type de source du message. Le flux pour le côté gauche définit le nom et le type de destination pour déclarer la liste des événements de départ (le nom et le type).

4.6 Cas étudier

4.6.1 Description de cas etudier

Nous avons choisi in cas d'étude dans le domaine dans le domaine des soins de santé, et son contenu est que nous suivons les indicateurs du patient ou d'une personne normale, pour collecter les informations que nous obtenons de différents capteurs : le capteur de fréquence cardiaque, le capteur de tension artérielle et le capteur de asymetrie du pouls, puis nous collectons tous les indicateurs que nous avons obtenus. Nous évaluons ses indicateurs par le moteur de la machine. Après cela, nous avons deux options. Si les indicateurs sont normaux, nous ne faisons rien, le travail s'arrête.S'ils ne sont pas bons, nous ferons appel à une aide humaine pour intervention, il peut s'agir d'appeler directement une ambulance ou d'appeler les médecins qui suivent le patient afin de traiter le patient dans les meilleurs délais.

4.6.2 Modélisation de l'application Soins de santé dans notre environnement

Le capteur de l fréquence cardiaque lit les valeurs du patient à intervalles réguliers à l'aide d'une minuterie dans le pool "CadiaqueAppariel" et les envoie au pool "centralProcess".

Le capteur de l la tension artérielle lit les valeurs du patient à intervalles

réguliers dans le pool "TensionArterielAppariel" et les envoie au pool "centralProcess".

Le capteur de lecture de la cohérence des pouls lit les valeurs du patient à intervalles réguliers dans la pool "PoulsAppariel" et les envoie à la pool "centraleProcess". le processus central "centralProcess" collecte les valeurs du patient via "tache reception message", qui reçoit les valeurs d'autres pool puis les passe ensemble dans la passerelle complexe afin d'analyser les valeurs dans la tache scripte"evaluer les mesures".

Si les résultats sont bons, il termine le processus par l'évenement fin, si Si les résultats sont mauvais, lance le pool d'assistance aux patients, qui contient une tâche manuelle qu'une personne fera, pas un mécanisme et le processus se termine

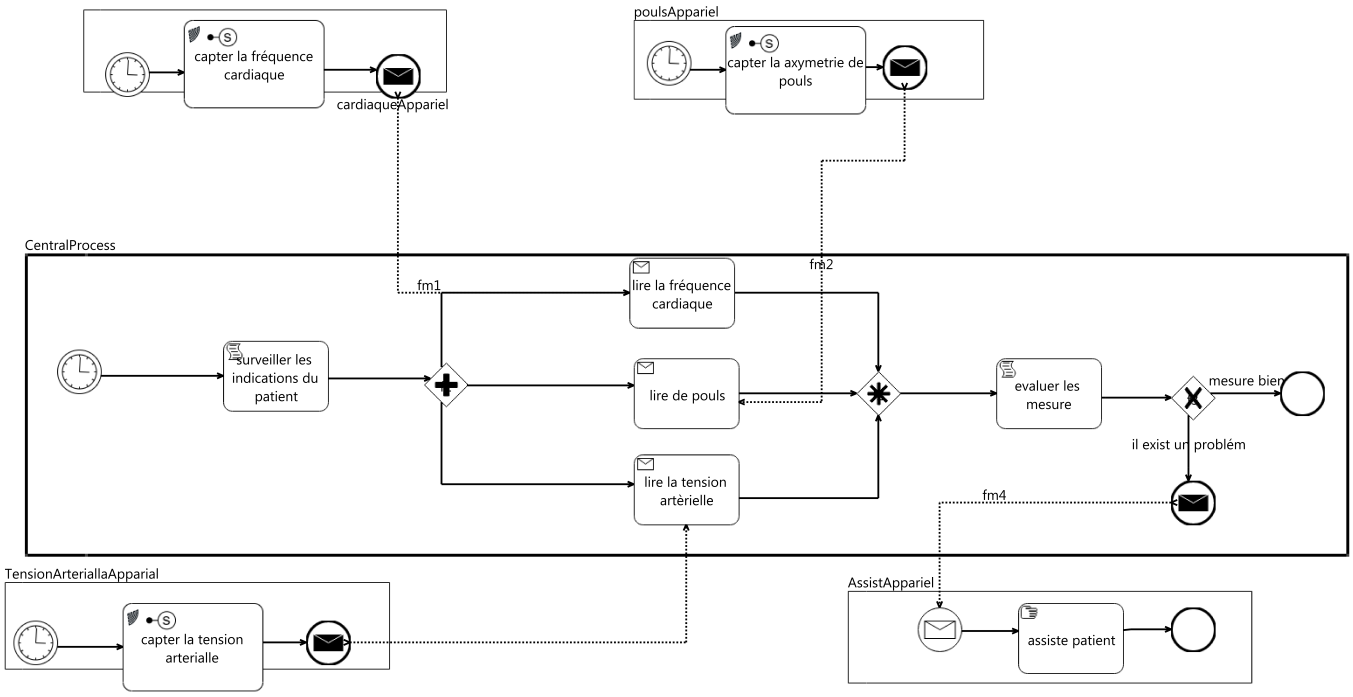


Fig. 4.18 : Le processus soins santé créé dans l'environnement graphique proposé.

4.6.3 Transformation de modèle crée en Maude

Pour obtenir le code Maude de notre diagramme nous créons un code dans Acceleo et exécutons la configuration afin d'obtenir le code Maude de notre modèle.lors de l'exécution de la configuration comme suit, nous choisissons le projet Acceleo que nous avons créé et sa classe principale, puis nous choisissons notre diagramme et le projet où il existe Enfin, nous

cliquons sur Run.Comme le montre l'image suivante :

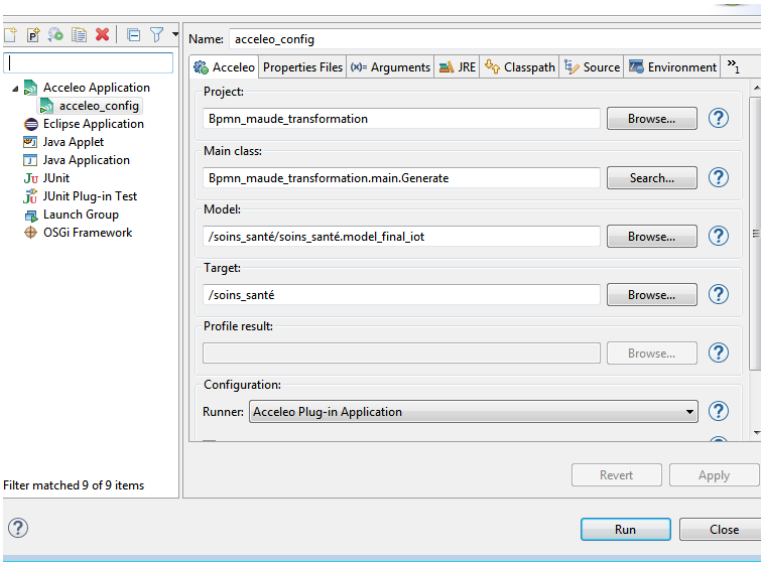


FIG. 4.19 : la configuration accleo.

Un fichier de type ".maude" apparaîtra avec notre diagramme de soins de santé on y retrouve le code Maude généré,, Comme le montre l'image suivante Fig. 4.20 : le resultat de configuration Accleo.

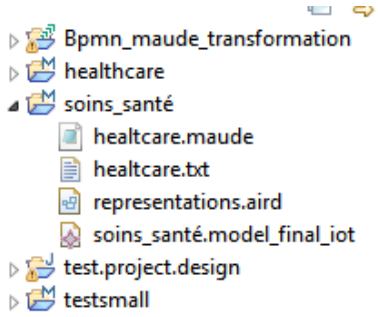


FIG. 4.20 : le résultat de configuration accleo.

Finalement nous avons obtenu le code Maude de notre diagramme de soins de santé.

```
in BPMN_MAUDE.maude
mod HEALTCARE is
including Basic-BPMN-IdO .
ops E5 E10 E1 E3 E8 : -> DebutEventNom .
ops E2 E4 E6 E7 E9 E11 : -> FinEventNom .

ops T1 T2 T3 T4 T5 T6 T7 T8 T9 : -> ActivitéNom .
ops f1 f2 f3 : -> passerelleNom .

rl [sf1] : [E1 : EvenMoniteurDebut] => [T1 : SensorT] .
rl [sf2] : [T1 : SensorT] => [E2 : EventMsgFin] .
rl [sf3] : [E3 : EvenMoniteurDebut] => [T2 : SensorT] .
rl [sf4] : [T2 : SensorT] => [E4 : EventMsgFin] .
rl [sf5] : [E5 : EvenMoniteurDebut] => [T3 : ScriptT] .
rl [sf6] : [T3 : ScriptT] => [f1 : parallèle] .
rl [f1] : [f1 : parallèle] =>[T4 : tache_reception_message][T5 : tache_reception_message][T6 : tache_reception_message] .
rl [f1] : [f1 : parallèle] =>[T4 : tache_reception_message][T5 : tache_reception_message][T6 : tache_reception_message] .
rl [f1] : [f1 : parallèle] =>[T4 : tache_reception_message][T5 : tache_reception_message][T6 : tache_reception_message] .
rl [f2] : [T5 :RecoiMsgT] [T4 :RecoiMsgT] [T6 :RecoiMsgT]  =>[f2 : complexe] .
rl [f2] : [T5 :RecoiMsgT] [T4 :RecoiMsgT] [T6 :RecoiMsgT]  =>[f2 : complexe] .
rl [f2] : [T5 :RecoiMsgT] [T4 :RecoiMsgT] [T6 :RecoiMsgT]  =>[f2 : complexe] .
rl [sf12] : [f2 : complexe] => [T7 : ScriptT] .
rl [sf13] : [T7 : ScriptT] => [f3 : exclusive] .
rl [f3] : [f3 : exclusive] => [E7 : évènement_fin_message] .
rl [f3] : [f3 : exclusive] => [E6 : évènement_fin] .
rl [sf16] : [E8 : EvenMoniteurDebut] => [T8 : SensorT] .
rl [sf17] : [T8 : SensorT] => [E9 : EventMsgFin] .
rl [sf18] : [E10 : EvenMsgDebut] => [T9 : ManualT] .
rl [sf19] : [T9 : ManualT] => [E11 : EventFin] .

rl [fm1] : [E2 : EventMsgFin] => [T4 : RecoiMsgT] .
rl [fm2] : [E4 : EventMsgFin] => [T5 : RecoiMsgT] .
rl [fm3] : [E9 : EventMsgFin] => [T6 : RecoiMsgT] .
rl [fm4] : [E7 : EventMsgFin] => [E10 : EvenMsgDebut] .
rew [ E1 : EvenMoniteurDebut ] [E5 : EvenMoniteurDebut] [E3 : EvenMoniteurDebut] [E8 : EvenMoniteurDebut] .
```

Fig. 4.21 : Le code Maude générer.

4.7 Écxution dans maude

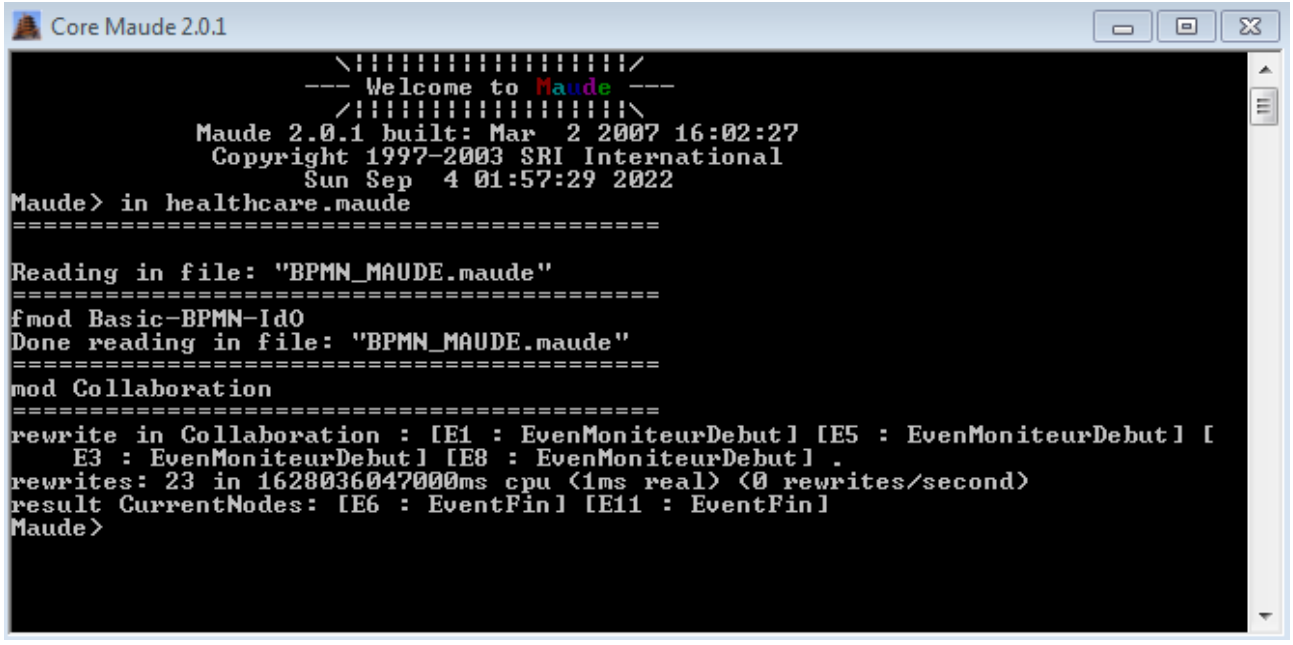


Fig. 4.22 : Le resultat d'éxécution de code maude.

4.8 Conclusion

Dans ce chapitre, nous avons proposé un outil d'environnement pour la modélisation d'applications IdO en utilisant BPMN2.0.Plus précisément, nous avons méta-modélisé un sous ensemble du langage BPMN adapté à la modélisation des applications IdO. nous avons utilisé la framework EMF et Sirius pour créer le metamodel et l'éditeur graphique. puis traduit les modèles BPMN étendus dans les spécifications Maude équivalentes a l'aide d'un transformation M2T en utilisant le langage Acceleo, afin de permettre la validation et l'analyse les modeles en Maude.

Conclusion générale

Conclusion générale

Ces dernières années, l'Internet des objets a émergé, qui a eu une empreinte proéminente dans de nombreux domaines, ce qui a suscité un grand intérêt des développeurs pour développer cette technologie et l'une des étapes les plus importantes du développement est la modélisation. C'est ce qui nous a poussé à essayer à créer un éditeur graphique utilisant un extension de BPMN comme langage de modélisation afin de faciliter la modélisation des applications Internet par les concepteurs.

Alors Dans ce travail, nous avons proposé un modèle d'extension de BPMN pour modéliser les applications IdO, en utilisant la plateforme Eclipse Modeling Tools en utilisant ses technologies EMF et Sirus pour le développement d'un environnement graphique. puis en l'exprimant dans Maud à l'aide du langage de transformation Accele pour la vérification et l'analyse des diagrammes.

Dans les travaux futurs, nous espérons améliorer l'extension bpmn d'IdO pour développer un outil de modélisation plus pratique qui inclut toutes les fonctionnalités et les besoins de modélisation des applications IdO.

Bibliographie

[1] K. Chandrakanth, J. Venkatesh, J. Uma Mahesh, Naganjaneyulu, S. Chandrakanth, K. Venkatesh, J. uma mahesh, Mahesh, and K. Naganjaneyulu, "Internet of things," *ijiacs*, vol. 8616, 10 2014.

[2] H. Sundmaeker, P. Guillemin, P. Friess, and S. Woelfflé, *Vision and Challenges for Realizing the Internet of Things*. 04 2010.

[3] P. Xiao, *Designing Embedded Systems and the Internet of Things (IoT) with the ARM mbed*. Wiley, 2018.

[4] Q. F. Hassan, *Internet of things A to Z : technologies and applications*. John Wiley & Sons, 2018.

[5] H. Ghayvat, S. Mukhopadhyay, X. Gui, and N. Suryadevara, "Wsn- and iot-based smart homes and their extension to smart buildings," *Sensors*, vol. 15, no. 5, pp. 10350–10379, 2015.

[6] P. Bellini, P. Nesi, and G. Pantaleo, "Iot-enabled smart cities : A review of concepts, frameworks and key technologies," *Applied Sciences*, vol. 12, no. 3, p. 1607, 2022.

[7] M. Jia, A. Komeily, Y. Wang, and R. S. Srinivasan, "Adopting internet of things for the development of smart buildings : A review of enabling technologies and applications," *Automation in Construction*, vol. 101, pp. 111–126, 2019.

[8] R. A. Rayan, C. Tsagkaris, and R. B. Iryna, "The internet of things for healthcare : applications, selected cases and challenges," in *IoT in Healthcare and Ambient Assisted Living*, pp. 1–15, Springer, 2021.

[9] V. K. Quy, N. V. Hau, D. V. Anh, N. M. Quy, N. T. Ban, S. Lanza, G. Randazzo, and A. Muzirafuti, "Iot-enabled smart agriculture : Architecture, applications, and challenges," *Applied Sciences*, vol. 12, no. 7, p. 3396, 2022.

[10] "Advantages and disadvantages of (iot) - javatpoin." https://www.javatpoint.com/iot-advantage-and-disadvantage.Page consultée le 11 juin 2022, 2022.

[11] S. Vashi, J. Ram, J. Modi, S. Verma, and C. Prakash, "Internet of things (iot) : A vision, architectural elements, and security issues," in *2017 international conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC)*, pp. 492–496, IEEE, 2017.

[12] M. Nikolaidou, G.-D. Kapos, A. Tsadimas, V. Dalakas, and D. Anagnostopoulos, "Simulating sysml models : Overview and challenges," in *2015 10th System of Systems Engineering Conference (SoSE)*, pp. 328–333, IEEE, 2015.

[13] B. Morin, N. Harrand, and F. Fleurey, "Model-based software engineering to tame the iot jungle," *IEEE Software*, vol. 34, no. 1, pp. 30–36, 2017.

[14] OMG, "Business Process Model and Notation (BPMN), Version 2.0," January 2011.

[15] M. Geller and A. Meneses, "Modelling iot systems with uml : A case study for monitoring and predicting power consumption," *American Journal of Engineering and Applied Sciences*, vol. 14, pp. 81–93, 01 2021.

[16] B. Tekinerdogan, Ö. Köksal, and T. Çelik, "Data distribution service-based architecture design for the internet of things systems," in *Connected Environments for the Internet of Things*, pp. 269–285, Springer, 2017.

[17] WFMC, "Workflow Management Coalition Terminology and Glossary (WFMC-TC-1011)," tech. rep., Workflow Management Coalition, Brussels, 1996.

[18] P.-A. Masse, *Conception contextuelle des interactions entre un modèle de processus opérationnel et des modèles de processus supports*. PhD thesis, Ecole nationale supérieure Mines-Télécom Atlantique Bretagne Pays de la Loire, 2019.

[19] A. H. Ter Hofstede and M. Weske, "Business process management : A survey," in *Proceedings of the 1st International Conference on Business Process Management, volume 2678 of LNCS*, Citeseer, 2003.

[20] N. A. Fedjkhi, "Une approche basée objectif pour la gestion des processus métier flexibles,"

[21] M. O. Kherbouche, *Contribution à la gestion de l'évolution des processus métiers*. PhD thesis, Université du Littoral Côté d'Opale, 2013.

[22] A. Simon, "Implémentation d'un éditeur bpmn au sein d'un outil de métamodélisation," *Mémoire de Master, Université de Naumer*, 2014.

[23] T. Vercruysse, "Modélisation et instanciation de processus sur des solutions techniquement hétérogènes," 2015.

[24] M. von Rosing, S. White, F. Cummins, and H. de Man, "Business process model and notation-bpmn.," 2015.

[25] T. Vercruysse, "Modélisation et instanciation de processus sur des solutions techniquement hétérogènes," 2015.

[26] A. Fleischmann, "Limitations of choreography specifications with bpmn," in *International Conference on Subject-Oriented Business Process Management*, pp. 203–216, Springer, 2020.

[27] S. A. White and D. Miers, *BPMN modeling and reference guide : understanding and using BPMN*. Future Strategies Inc., 2008.

[28] N. Brouns, S. Tata, H. Ludwig, E. Asensio, and P. Grefen, "Modeling iot-aware business processes - a state of the art report," 11 2018.

[29] T. De Meyer, *Integrating the Internet of Things into Business Process Management*. PhD thesis, Master's Thesis. University of Gent, 2016.

[30] G. Barbier, *Contribution de l'ingénierie dirigée par les modèles à la conception de modèles grande culture*. PhD thesis, Université Blaise Pascal-Clermont-Ferrand II, 2013.

[31] B. Combemale, "Ingénierie dirigée par les modèles (idm)–état de l'art," 2008.

[32] m. Rezali and l. Titi, *Un Environnement Graphique pour la Conception du Comportement de l'Internet des Objets : une Approche basée sur les Processus Métier*. PhD thesis, Université de Jijel, 2020.

[33] R. Boukhedenna, M. Mesbah, and E.-h. E. Kerkouche, *La co-évolution des modèles et des méta-modèles*. PhD thesis, université de jijel, 2021.

[34] E.-h. Kerkouche, "Modélisation multi-paradigme : Une approche basée sur la transformation de graphes," 2011.

[35] A. Harbouche, *Une approche dirigée par les modèles pour une conception flexible des systèmes distribués*. PhD thesis, Universite Mohamed Khider Biskra, 2018.

[36] F. Bacha, *Une approche MDA pour l'intégration de la personnalisation du contenu dans la conception et la génération des applications interactives*. PhD thesis, Université de Valenciennes et du Hainaut-Cambresis, 2013.

[37] M. Bouarioua, "Une approche basée transformation de graphes pour la génération de modèles de réseaux de petri analysables à partir de diagrammes uml," *UNIVERSITÉ CONSTANTINE*, vol. 2, 2013.

[38] K. Czarnecki and S. Helsen, "Classification of model transformation approaches," in *Proceedings of the 2nd OOPSLA Workshop on Generative Techniques in the Context of the Model Driven Architecture*, vol. 45, pp. 1–17, USA, 2003.

[39] A. Gargantini, E. Riccobene, and P. Scandurra, "Integrating formal methods with model-driven engineering," in *2009 Fourth International Conference on Software Engineering Advances*, pp. 86–92, IEEE, 2009.

[40] http://planetmde.org/. Page consultée le 26 aout 2022.

[41] "Eclipse modeling tools." https://www.eclipse.org/downloads/packages/release/2020-06/r/eclipse-modeling-tools. Page consultée le 26 aout 2022.

[42] J. Sprinkle, B. Rumpe, H. Vangheluwe, and G. Karsai, "3 metamodelling," in *Dagstuhl Workshop on Model-Based Engineering of Embedded Real- Time Systems*, pp. 57–76, Springer, 2007.

[43] D. Steinberg, F. Budinsky, E. Merks, and M. Paternostro, *EMF : eclipse modeling framework*. Pearson Education, 2008.

[44] J. Barzic, "Eclipse et ses plugins de modélisation(emf–gef– gmf)," *PDF créé le*, vol. 12, 2008.

[45] V. Vujović, M. Maksimović, and B. Perišić, "Comparative analysis of dsm graphical editor frameworks : Graphiti vs. sirius," in *23nd International Electrotechnical and Computer Science Conference ERK*, pp. 7–10, 2014.

[46] "What is sirius ?." https://www.eclipse.org/community/eclipse_newsletter/2013/november/article1.php. Page consultée le 26 aout 2022.

[47] V. Viyović, M. Maksimović, and B. Perisić, "Sirius : A rapid development of dsm graphical editor," in *IEEE 18th International Conference on Intelligent Engineering Systems INES 2014*, pp. 233–238, IEEE, 2014.

[48] I. B. Kara, "Design and implementation of the modelicaml code generator using acceleo 3. x," 2015.

[49] M. Clavel, F. Durán, S. Eker, P. Lincoln, N. Martı-Oliet, J. Meseguer, and J. F. Quesada, "Maude : Specification and programming in rewriting logic," *Theoretical Computer Science*, vol. 285, no. 2, pp. 187–243, 2002.

[50] Z. Nouri and T. Marir, "Spécification formelle des systèmes multi-agents-une approche basée sur le langage maude," 2020.

[51] M. Clavel, F. Duran, S. Eker, P. Lincoln, N. Marti-Oliet, J. Meseguer, and J. F. Quesada, "The maude system," in *International Conference on Rewriting Techniques and Applications*, pp. 240–243, Springer, 1999.

[52] F. Bellour and C. E. Djaoui, *Spécification mode de diagramme global d'interaction*. PhD thesis, Université de Jijel, 2019.

[53] A. Riesco, "Test-case generation for maude functional modules," in *International Workshop on Algebraic Development Techniques*, pp. 287–301, Springer, 2010.