Simulations multi-agent pour les villes intelligentes : une architecture multi-environnement temporelle, spatiale et organisationnelle. Apports pour l'anticipation

Tahina Vololona Eulalie Ralitera

UNIVERSITÉ DE LA RÉUNION

Thèse de Doctorat de l'Université de la Réunion

Spécialité

Informatique

présentée par

Tahina Vololona Eulalie Ralitera

pour obtenir le grade de

Docteur de l'Université de la Réunion

Simulations Multi-Agent pour les Villes Intelligentes : une Architecture Multi-Environnement Temporelle, Spatiale et Organisationnelle. Apports pour l'Anticipation.

Soutenue publiquement le 04 Septembre 2020 au Laboratoire d'Informatique de Paris 6 (LIP6), Sorbonne Université, devant le jury composé de

Pr. Florence Sedes	**Présidente du jury**
Dr. HDR. Zahia Guessoum	**Rapporteure**
Pr. Laurent Vercouter	**Rapporteur**
Pr. Amal El Fallah Seghrouchni	**Examinatrice**
Pr. Olivier Boissier	**Examinateur**
Pr. Rémy Courdier	**Directeur de thèse**
Dr. Denis Payet	**Co-encadrant**

Cette thèse a reçu le soutien financier de la Région Réunion, de l'Union Européenne (Fonds Européen de Développement Régional - FEDER) dans le cadre du Programme Opérationnel de Coopération Territoriale, de la fondation L'Oréal dans le cadre des bourses l'Oréal-UNESCO pour les femmes et la science et de la Mairie de Saint-Denis La Réunion.

Abstract

The multiagent simulation is a promising approach for smart city design and planning. In this context, we focus on the example of recharging electric vehicles on public charging points. This example illustrates a problem of managing limited and shared resources in time and space.

Rolland-May [83] defines three main dimensions that should be integrated by the system : the space, the organisation and the time. In multi-agent simulations, the spatial dimension and the social dimension are the subject of numerous proposals in the literature. In opposite, time remains subject to very few studies and consideration. In addition, if a lot of research deals with spatial and organisational consideration in the agent's reasoning, the time consideration, as a system dynamic, is often overlooked. This highlights two aspects to which we want to contribute :

— the need for interaction support to exchange spatial, social and temporal information ;
— the need for reasoning that takes this exchanged spatial, temporal and organisational information into account.

Thought this thesis, our first objective aim at making the multiagent simulation paradigm evolve in order to consider time as a new medium of interaction, in the same way as the spatial environment or the organisational environment. For that purpose, we draw on existing approaches that are commonly used for modelling the space and organisations. Our model is called Agent-Group-Environment-Time (AGRET). It is an extension of the generic organisational model Agent-Group-Rôle (AGR) [32] and its variant Agent-Group-Rôle-Environnement (AGRE) [32].The originality of our approach is that it integrates the temporal dimension as an environment, in the same way as the spatial environment and the social environment. This time environment is used to support the exchange and the storage of time information. It complements the simulation scheduler which manages the simulation activation cycle. The implementation of this new interaction environment brings new possibilities. One of these possibilities is the use of temporal, spatial and social information, perceived through the environments, to optimise the agent's reasoning. In this context, we choose to focus on anticipatory reasoning which is particularly interesting in the context of the smart city. This anticipatory reasoning increases the realism of the simulation by showing a cognitive capacity that is specific to humans. It also improves the agent's decision mechanism by choosing a more relevant behaviour that takes into account the agent's temporal, spatial and social activation context. This anticipatory reasoning is based on information about the past, the present and the future, which the agent perceives through the temporal environment. The inclusion of future information in the anticipative reasoning is an original feature of this approach. This functionality is made possible by the temporal environment, which allows storing and perceiving information on the temporal dimension.

To summarise, our contributions are both about time. Our first contribution is about the representation of time as an environment. In the multi-agent level, we propose an interaction support for the exchange and storage of information on space, time and organisation. Our second contribution is about temporal reasoning. We propose an anticipative reasoning based on the perception of spatial, temporal and social environments. In particular, we exploit the visibility of the future dimension of time that is allowed by the temporal environment.

In the example of electric vehicles recharge, the integration of our approaches allows, at the collective level, the optimisation of the recharge distribution in space and time. We show this through an implementation on a multi-agent simulation model called SkuadCityModel. More generally, at the level of the smart city, the implementation of our contributions allows the optimisation of resource management in space and time.

Résumé

La simulation multi-agent est une approche prometteuse pour la construction et la planification des villes intelligentes. Dans ce contexte, nous nous basons sur l'exemple du rechargement de véhicules électriques avec des bornes de recharge publiques. Cet exemple illustre une problématique de gestion de ressources partagées et limitées dans le temps et dans l'espace.

Rolland-May [83] définit trois principales dimensions que doit intégrer le système : l'espace, l'organisation et le temps. Dans les simulations multi-agent, la dimension spatiale et la dimension sociale font l'objet de nombreuses propositions dans la littérature. Contrairement à cela, la considération du temps comme une dynamique du système reste sujette à très peu d'étude et de considération. De plus, si beaucoup de travaux de recherche traitent de la considération spatiale et organisationnelle dans le raisonnement de l'agent, la considération de la dynamique temporelle est souvent négligée. Cela met en évidence deux aspects sur lesquels nous aimerions contribuer :

— le besoin de support d'interaction pour échanger des informations spatiale, sociale et temporelle ;
— le besoin de raisonnement qui prend en compte ces informations spatiale, temporelle et organisationnelle échangées.

À travers cette thèse, notre premier objectif consiste à faire évoluer le paradigme de simulation multi-agent de manière à considérer le temps comme un nouveau milieu d'interaction. Pour cela, nous nous basons sur des approches existantes qui sont communément utilisées pour modéliser l'espace et les organisations. Notre modèle est appelé Agent-Groupe-Rôle-Environnement-Temps (AGRET). Il s'agit d'une extension du modèle générique d'organisation AGR et de sa variante AGRE [32]. L'originalité de notre approche consiste en la considération de la dimension temporelle comme un environnement, au même titre que l'environnement spatial et l'environnement social. Cet environnement temporel est utilisé comme support pour l'échange et le stockage d'informations temporelles. Il vient en complément à l'ordonnanceur de la simulation qui gère le cycle d'activation de la simulation. L'implémentation de ce nouveau milieu d'interaction apporte de nouvelles possibilités. Une de ces possibilités est l'usage des informations temporelles, spatiales et sociales, perçues à travers les environnements, pour optimiser le raisonnement de l'agent. Dans ce cadre, nous choisissons de nous concentrer sur le raisonnement anticipatif qui est particulièrement intéressant dans le contexte de la ville intelligente. Ce raisonnement anticipatif augmente le réalisme de la simulation en faisant transparaitre une capacité cognitive qui est propre à l'humain. Il permet également d'améliorer le mécanisme de décision de l'agent en choisissant un comportement plus pertinent qui prend en compte le contexte d'activation temporel, spatial et social de l'agent. Ce raisonnement anticipatif se base sur des informations sur le passé, sur le présent et sur le futur, que l'agent perçoit au niveau de l'environnement temporel. La prise en compte des informations futures dans le raisonnement anticipatif constitue une originalité de cette approche. Cette fonctionnalité est permise par l'environnement temporel qui rend possible le stockage et la perception d'informations sur la dimension temporelle.

Pour résumer, nos deux contributions relèvent du temps. Notre première contribution concerne la représentation du temps comme un environnement. Au niveau multi-agent, nous proposons un support d'interaction pour l'échange et le stockage d'informations sur l'espace, le temps et l'organisation. Notre deuxième contribution concerne le raisonnement temporel. Nous proposons un raisonnement anticipatif basé sur la perception de l'environnement spatial, de l'environnement temporel et de l'environnement social. Plus particulièrement, nous exploitons la visibilité sur la dimension future du temps qui est permise par l'environnement temporelle.

Dans l'exemple du rechargement des véhicules électriques, l'intégration de notre approche permet l'optimisation de la répartition des recharges dans l'espace et dans le temps. Nous montrons cela à travers une implémentation sur un modèle de simulation multi-agent appelé SkuadCityModel. Plus généralement, au niveau de la ville intelligente, l'implémentation de nos contributions permet l'optimisation de la gestion des ressources dans l'espace et dans le temps.

Remerciements

Bien plus qu'une simple expérience scientifique et intellectuelle, cette thèse a été une belle expérience de vie. J'ai le sentiment d'avoir beaucoup grandi et beaucoup appris, mentalement et humainement. À part les contributions scientifiques, ma thèse m'a notamment ouvert les yeux et m'a fait réfléchir sur l'humain, le monde et la société. Depuis, j'essaie d'adopter un mode de vie plus respectueux de l'environnement, de l'humain et du collectif. Je suis également plus engagée dans la lutte pour l'égalité. Tout cela n'aura été possible sans ces personnes envers qui je souhaiterais témoigner toute ma reconnaissance.

Merci à toi qui, depuis mon départ de Madagascar, m'as promis que tu ne me laisserais pas et que tu ne m'abandonnerais pas.

Merci à vous Professeur Rémy Courdier d'avoir dirigé ma thèse. Vous m'avez fait découvrir et aimer l'univers des Systèmes Multi-Agent. Vous avez cru en moi et m'avez soutenu depuis l'école d'ingénieurs. Merci pour la confiance que vous m'avez accordée même lors de mes prises de décisions "osées" et "risquées". Merci également pour votre accompagnement.

Merci à vous Docteur Denis Payet d'avoir co-encadré ma thèse. Vous êtes sans doute l'un des enseignants qui m'ont le plus marqué. J'admire vos qualités en tant qu'informaticien. Depuis votre participation à mon jury de sélection pour entrer en écoles d'ingénieurs, vous avez grandement contribué à mon orientation et mon choix de parcours.

Thanks to you Doctor Koen Van Dam. You actively participated in the supervision of my thesis. You have always been available and attentive when I needed it the most, even if you were not obliged to do so. Thank you for your support.

Merci à vous Professeur René Mandiau, Professeur François Guerrin, Docteur Jean Christophe Soulié. Vous étiez les membres de mon comité de suivi de thèse. Merci pour vos conseils et pour votre accompagnement.

Merci à vous Docteure Zahia Guessoum, Professeur Laurent Vercouter, Professeure Florence Sedes, Professeure Amal El Fallah Seghrouchni, Professeur Olivier Boissier. Vous avez accepté de participer au jury de ma thèse. Je suis honorée et je vous remercie pour le regard critique que vous apportez à mes travaux de recherche.

Merci au Laboratoire l'Informatique et de Mathématiques (LIM) de l'Université de La Réunion qui m'a accueilli depuis ma dernière année d'ingénieurs. J'ai pu effectuer ma thèse dans de bonnes conditions de travail. Merci également à Télécom Saint-Etienne et au Laboratoire Hubert Curien (LHC) qui m'a accueilli lors de ma dernière année de thèse. Merci à mes collègues qui ont su égayer mes journées et avec qui j'ai passé de très agréables moments.

Je remercie la fondation L'Oréal pour les Femmes et la Science, l'UNESCO ainsi que toutes les personnes qui ont participé au programme L'OREAL-UNESCO pour les femmes et la science. Cette bourse a été un véritable tremplin pour ma carrière scientifique et de manière plus générale pour ma vie en tant que femme.

Merci à toi No pour ces 3 années durant lesquelles tu n'as pas cessé de me soutenir. Tu es resté à mes côtés dans mes meilleurs comme dans mes pires moments. Merci de m'aimer même quand je ne le mérite pas. Cette thèse est en quelque sorte la nôtre.

Merci à ma famille qui a toujours cru en ma réussite, qui a cultivé mon enthousiasme et ma volonté à sortir des sentiers battus.

Merci à vous Amis Vrais et Précieux. Vous m'avez accueilli depuis mon arrivée à La Réunion et je vous ai considéré comme ma deuxième famille. Je vous suis reconnaissante pour tout ce que vous m'avez apporté.

Merci à La Famille qui me soutient et qui m'accepte telle que je suis.

Merci à vous, amis et amies malgaches de Saint-Etienne. Vous m'avez soutenu chacun, chacune à votre manière.

Merci à toi qui as relu et corrigé mes travaux. Merci à toi qui m'as aidé dans l'élaboration de mes supports de présentation. Merci à toi qui m'as aidé dans l'écriture de codes. Merci à chacun et chacune d'entre vous qui avez contribué à ma thèse. Je vous suis entièrement reconnaissante : du fond du cœur, merci.

Table des matières

Table des figures

Liste des tableaux

Glossaire

AGR Agent-Group-Rôle. 1, 5, 14, 37–40, 42, 45–48, 53, 63, 66, 134

AGRE Agent-Group-Rôle-Environnement. 1, 14, 37, 39, 40, 42, 45–48, 53, 63, 66, 134

AGRET Agent-Group-Rôle-Environnement-Temps. 2, 3, 5, 13, 14, 40, 42, 45, 47–49, 53, 55, 63–68, 71–73, 82, 83, 85, 106, 113, 116–118, 123, 134, 136, 139, 141

GPS Global Positioning System. 119

ICL Imperial College London. 18, 95

IRM4S Influence/Reaction Model for Simulation. 2, 35, 36, 47, 53, 54, 56, 62, 63, 70, 74, 135

LIM Laboratoire d'Informatique et de Mathématiques. 10, 86

MOISE Model of Organization for multI-agent SystEms. 37, 40, 45

OperA Organizations per Agents. 37

RIO Rôles Interactions Organisations. 37

SIG Système d'Information Géographique. 25, 50, 106, 119, 131

SKUAD Software Kit for Ubiquitous Agent Development. 86, 87

SMA Système Multi-Agent. 3, 9, 10, 13, 14, 18, 21, 22, 24, 26, 27, 29, 38–42, 45–47, 64, 66, 68, 72, 74, 83, 86, 87, 134, 136, 140

TEAMS Task Analysis Environment Modeling and Simulation. 37

TIC Technologies de l'Information et de la Communication. 9, 17, 21

Chapitre 1

Introduction

1.1 Contexte

1.1.1 Contexte général

Selon les Nations Unies, en 2018, plus de la moitié (55%) de la population mondiale vivait dans des espaces urbains. Ce pourcentage devrait atteindre les 68% d'ici 2050 [2]. Face à ce phénomène d'urbanisation croissante, les zones urbaines sont actuellement sujettes à des crises économiques et environnementales. Ces crises mettent une énorme pression sur la structure des villes et sur la gestion des ressources. Combiner simultanément compétitivité et développement durable devient alors un grand défi. Cela force les citoyens, les gouvernements et les parties prenantes à trouver des solutions techniques permettant de réduire ces problèmes urbains tout en prêtant une attention particulière à l'environnement et au développement durable.

En parallèle à cela, depuis les dernières décennies, les Technologies de l'Information et de la Communication (TIC) ont contribué à de grands changements au niveau des villes sur tous les aspects : économique, culturel, transport, communication, etc. Les villes deviennent de plus en plus numériques et basées sur l'information [98]. Ces tendances ont conduit à la popularité croissante du concept de ville intelligente ou "smart city". Cette dernière est considérée comme étant une solution aux problèmes et objectifs évoqués précédemment. Selon Giffinger [40], une ville intelligente est une ville performante en 6 caractéristiques (environnement, économie, mobilité, citoyen, habitat, gouvernance) construite sur la combinaison "intelligente" de dotations et d'activités de citoyens auto décisifs, indépendants et conscients.

La ville est un système complexe caractérisé par :
- Des micro-interactions humains-humains et humains-environnement ;
- Une émergence de phénomènes inattendus qui découlent du comportement d'unités indépendantes ;
- Des dynamiques non linéaires : cela signifie qu'il est difficile de prédire la sortie du système à partir de ses entrées ;
- Des boucles de rétroaction.

Elle se compose de différents sous-ensembles, dont les citoyens qui peuvent avoir des comportements divers et variés. Bien que souvent négligés, ces citoyens jouent un rôle déterminant dans la mise en œuvre de la ville intelligente. En effet, le concept de la ville intelligente fait passer la conception, la planification et la gouvernance de la ville d'une logique descendante ou "top-down" à une logique ascendante ou "bottom-up". Il s'agit d'une approche plus inclusive des citoyens. Désormais, dans les villes intelligentes, ces derniers ne sont plus considérés comme de simples utilisateurs du système, mais comme faisant partie intégrante de celui-ci. L'approche ascendante amène les citoyens à s'engager en faveur du développement urbain de manière pro-active. De plus, l'avènement des réseaux sociaux et de l'open data a contribué fortement à l'accélération de la mise en place de cette approche. Ils facilitent grandement la participation des citoyens à la construction de la ville intelligente et des services numériques qui s'y rapportent.

L'approche multi-agent est une approche prometteuse [42] [57] [84] permettant de modéliser ces types de systèmes. C'est une approche ascendante. Elle fait l'objet depuis longue date de recherches en intelligence artificielle distribuée. Appliqués au système de la ville, les SMA permettent de décomposer cette dernière en plusieurs sous-ensembles plus simples, gérés par plusieurs entités

autonomes, sociales, réactives et pro-actives appelées agents. Les SMA permettent également de modéliser la non-linéarité et favorisent l'émergence des normes et des protocoles nécessaires pour les villes intelligentes [84].

Le domaine des SMA mobilise des scientifiques issus majoritairement de l'informatique, des sciences cognitives et des systèmes complexes. Il produit un large spectre de solutions dans des champs d'applications tels que le développement de systèmes informatiques décentralisés, la résolution collective de problème, le développement de systèmes médiatisés ou encore la simulation de phénomènes complexes. C'est ce dernier champ d'application qui nous intéresse dans le cadre de nos travaux. Nous nous focalisons particulièrement sur les modèles de simulation multi-agent. Ces modèles de simulation servent à analyser le fonctionnement de scénario choisi, afin de déduire des enseignements pratiques de gestion opérationnelle dans le cadre de la conception et de la planification de villes intelligentes.

1.1.2 Contexte applicatif

Ce travail de thèse s'inscrit dans la continuité des travaux étudiés précédemment par le groupe de travail Systèmes Collectifs Adaptatifs du Laboratoire d'Informatique et de Mathématiques (LIM) de l'Université de La Réunion. Notre équipe de recherche est spécialisée depuis plus de vingt (20) ans dans le paradigme des SMA, avec un accent particulier sur les aspects de simulations. Néanmoins, la politique actuelle de l'équipe s'oriente plus vers les approches de type hybrides. Cela veut dire qu'elle ne s'intéresse plus uniquement à la simulation, mais aussi à l'application dans un environnement réel. Un des sous-thèmes développés consiste en l'application des simulations multi-agent dans le domaine des villes intelligentes et des îles intelligentes. Il s'agit d'un domaine en émergence dans notre équipe et sur lequel nous travaillons en collaboration avec des chercheurs de l'Imperial College London (ICL) et avec la mairie de Saint-Denis, La Réunion. Notre objectif est de développer des outils d'aide à la décision pour la mise en place, la planification et la gestion des villes intelligentes. Les problématiques abordées sont en lien avec la mobilité intelligente. Il s'agit d'un des six (6) domaines clés de la ville intelligente selon la définition de Giffinger [40]. Nous appliquons notre approche sur un modèle de simulation multi-agent de flux de déplacement de véhicules électriques sur un territoire. Nous développons ce modèle au sein même de notre équipe. Il s'appelle SkuadCityModel [3]. Il se base sur un modèle développé par les chercheurs de l'ICL appelé SmartCityModel [8] et sur lequel nous avons également contribué [76]. L'objectif initial de l'étude était de concevoir un modèle de simulation assez générique pour être facilement adaptable d'un territoire à un autre. Par exemple : d'une ville à une île et vice-versa. Dans ce cadre, nous avons mené des expérimentations sur la ville de Londres et l'île de La Réunion. Le choix des deux territoires se base sur leurs caractéristiques physiques et sociales très différentes, mais complémentaires. En effet, Londres et La Réunion représentent deux extrêmes, dans le sens où contrairement à Londres, La Réunion est un territoire à très forte contrainte géographique. Nous décrivons les résultats de ces premières expérimentations dans les articles suivants : [76] [75]. Les problématiques abordées dans le cadre de cette thèse découlent des limites détectées lors de ces expérimentations. L'exemple sur lequel nous nous basons est le rechargement des véhicules électriques avec des bornes de recharge publiques. Dans ce contexte, nous appliquons SkuadCityModel à la simulation des flux de déplacements quotidiens de véhicules électriques sur un territoire. Dans l'ensemble (au niveau collectif), l'objectif principal est d'optimiser la répartition, dans le temps et dans l'espace, du rechargement des véhicules électriques et de l'occupation des bornes de recharge électrique. Cet exemple illustre une problématique de gestion de ressources partagées et limitées dans l'espace et dans le temps. Cette dernière fait partie des problèmes à l'origine même de la création du concept de ville intelligente.

La recharge de véhicules électriques implique principalement deux entités : le conducteur de voiture électrique et la borne de recharge. Ces deux entités sont autonomes et indépendantes. Dans SkuadCityModel, comme dans la réalité, un automobiliste possède un emploi du temps dont il a plus ou moins connaissance à l'avance. Cet emploi du temps se compose de différentes activités de la vie quotidienne nécessitant des déplacements : aller travailler, aller se divertir, rentrer chez soi ou faire ses courses. Ces activités sont situées dans le temps et dans l'espace. Cela revient à dire qu'à chaque activité, nous pouvons associer une composante spatiale (le conducteur va travailler à un endroit précis), et une composante temporelle (le conducteur part travailler à une heure précise). L'objectif principal de l'automobiliste est d'accomplir l'ensemble des activités qu'il a prévues dans son emploi du temps. Pour ce faire, il se déplace en utilisant sa voiture électrique. Cependant,

ce dernier possède une autonomie de batterie limitée. Par conséquent, le rechargement devient une contrainte. Le conducteur optimise alors au maximum le rechargement, de manière à ce qu'il puisse respecter son emploi du temps. Cette optimisation consiste à choisir le bon moment pour se recharger et à minimiser la durée du rechargement. Cette durée de rechargement inclut la durée d'attente au niveau de la borne de recharge.

Pour la borne de recharge, l'objectif principal est d'optimiser son taux d'occupation. Le nombre de bornes de recharge est limité. En France par exemple, le ratio de véhicules par point de charge est de 5,7 [1]. Ce nombre peut varier en fonction de la situation géographique de la borne. Par conséquent, la disponibilité d'une borne n'est garantie ni à tout moment ni à tout endroit, car d'autres véhicules peuvent l'utiliser.

Cette situation a des répercussions tant au niveau de la borne qu'au niveau de l'automobiliste. Pour la borne, une forte demande de rechargement augmente le nombre de véhicules en attente (la longueur de la file d'attente). Elle est donc contrainte à trouver un moyen de bien gérer la répartition du rechargement des véhicules dans sa file d'attente. Cependant, il s'agit d'une tâche difficile, car la borne n'a pas connaissance ni de l'état de chaque véhicule ni de l'emploi du temps de l'automobiliste qui le conduit. Elle ne connaît donc pas la disponibilité de leur conducteur. Du côté de l'automobiliste, l'occupation des bornes et l'augmentation de la longueur de la file d'attente multiplient le temps nécessaire pour effectuer le rechargement. Elle réduit également le nombre de créneaux de rechargement disponibles. Cela constitue une contrainte qui pourrait perturber son emploi du temps et l'empêcher d'atteindre son objectif. L'automobiliste devra donc prendre en compte les informations concernant l'emploi du temps et l'état de la borne dans ses décisions. Cependant, tout comme pour la borne, aucun support n'est prévu pour que l'automobiliste puisse accéder à la totalité des informations qui sont détenues par la borne.

Nous avons donc deux obstacles au bon déroulement du rechargement. D'un côté nous avons la limitation du nombre de bornes disponibles qui sont des ressources partagées et de l'autre côté la gestion du temps (durée de rechargement et créneaux disponibles) pour le chargement du véhicule. Nous estimons qu'il est possible de limiter l'impact de ces obstacles sur leurs objectifs. Pour cela, nous proposons de faire collaborer toutes les entités du système au sein d'un même collectif. L'idée est de permettre à chaque entité de s'engager en faveur de l'objectif collectif et de tirer profit de ce collectif pour mieux atteindre son propre objectif. Cette démarche est totalement en accord avec l'un des principes de base même du concept de ville intelligente que nous avons expliqué précédemment. Ce collectif permet l'émergence d'un objectif commun qui est l'optimisation du rechargement. Pour la borne cet objectif permet d'améliorer la répartition des recharges. Pour l'automobiliste, cet objectif permet une meilleure gestion des créneaux de recharge et minimise le temps nécessaire pour le chargement. Cela le conduit à satisfaire son objectif principal d'accomplir l'ensemble des activités qu'il a prévu dans son emploi du temps.

Comme expliqué précédemment, faire collaborer les différentes entités au sein d'un même collectif nécessite une interaction entre les entités. Cela revient à dire que ces entités doivent communiquer entre elles. Elles doivent aussi échanger des données ou informations leur permettant d'atteindre leurs objectifs. Cette interaction permet par exemple au conducteur de véhicule électrique de prendre connaissance de la localisation, de l'état et de l'emploi du temps de la borne. Ces informations peuvent être partagées par la borne elle-même. Cet échange d'information nécessite un support d'interaction.

Une fois les informations échangées, chaque entité pourra l'utiliser dans le cadre de son raisonnement afin d'adapter son comportement de manière à atteindre son objectif.

Cet exemple simple fait ressortir des aspects que nous considérons essentiels à gérer dans le domaine de la mobilité intelligente et de manière plus générale dans le domaine de la ville intelligente. Ces aspects sont :

- le besoin de support d'interaction permettant d'échanger des informations spatiales, sociales et temporelles ;
- le besoin d'un raisonnement qui prenne en compte les informations spatiales, sociales et temporelles échangées.

Ces aspects sont abordés dans la section 1.2 suivante en tant que problématiques principales de cette thèse.

1. https ://www.ecologique-solidaire.gouv.fr/sites/default/files/2019-07-Rapport-IRVE.pdf

1.2 Problématiques

La gestion de ressources partagées et limitées dans l'espace et dans le temps est une des problématiques centrales à l'origine du concept de la ville intelligente. Dans notre exemple, nous identifions deux types de ressources à gérer : la borne de recharge publique et le temps.

La borne de recharge publique est une ressource partagée et limitée, car le nombre de bornes est inférieur au nombre de véhicules électriques. L'accès à une borne est alors partagé entre plusieurs véhicules.

La deuxième ressource est le temps. Ce temps revêt deux aspects. La première est le temps de recharge. Il s'agit du temps d'attente avant de se recharger et pour se recharger. Ce temps est proportionnel à la longueur de la file d'attente au niveau de la borne. Le deuxième aspect est le nombre de créneaux de rechargement disponibles parmi lesquels l'automobiliste peut faire son choix. La gestion du temps consacré au rechargement est essentielle si le conducteur veut se conformer à son objectif principal qui est de respecter son emploi du temps. Dans ce sens, contrairement à la première ressource, le temps n'est pas seulement une ressource à partager, il est aussi une ressource à optimiser.

D'une manière plus générale, dans un système tel que la ville intelligente, nous retrouvons ces deux types de gestion de ressource [93]. Le premier est la gestion d'un accès concurrent à une ressource limitée dans l'espace et le temps. Par exemple : la gestion du rechargement de véhicules électriques avec des bornes publiques, la gestion des places de parking, la gestion de l'accès à des véhicules libre-service comme les vélos, les voitures ou les trottinettes. Le deuxième est une optimisation de l'emploi d'une ressource dans l'espace et le temps. Par exemple : l'optimisation du temps d'attente pour les automobilistes au niveau de la borne, l'optimisation du temps de recherche et du trajet effectué pour la recherche de place de parking, l'optimisation de l'utilisation de l'énergie en situation de quartier, îlots à énergie zéro, ou dans des territoires insulaires, comme La Réunion. Nous tenons à bien souligner l'importance de la prise en compte, au même titre, de ces deux dimensions, espace et temps, dans les solutions que nous proposons. Dans les simulations multi-agent, nous relevons alors deux problématiques principales sur lesquelles nous souhaitons contribuer :

1.2.1 Un besoin en support d'interaction

Comme mentionné en début de ce manuscrit, le concept de ville intelligente, et donc de la mobilité intelligente, se base sur une intelligence participative. Il s'agit d'une intelligence qui émerge de la participation de chaque entité du système afin de satisfaire un objectif commun. En d'autres termes, chacun doit adopter un comportement qui tend à satisfaire un but commun.

Nous avons également vu précédemment que la richesse des interactions entre entités joue un rôle clé dans la mise en place d'un tel système. Pour permettre ces échanges, nous proposons de mieux intégrer les informations temporelles dans le système de perception d'informations qui influence le comportement des agents.

1.2.2 Un besoin en raisonnement

Cette deuxième problématique vient en complément à celle citée précédemment. En effet, il est nécessaire que les informations spatiales, temporelles et sociales échangées soient prises en compte dans le processus de choix de comportement au sein même de chaque agent. Le but est de permettre aux agents de les exploiter afin de choisir un comportement qui convient, le mieux possible, à la fois à ses objectifs personnels et aux objectifs collectifs. Pour ce faire, il est nécessaire de mettre en place un mécanisme de raisonnement. Dans notre cas, nous manipulons des informations spatio-temporelles. Nous avons donc besoin d'un raisonnement spatio-temporel qui s'appuie sur les échanges d'informations spatiales et temporelles.

Notre proposition doit alors tenir compte de ces aspects et du contexte applicatif de la thèse qui est les villes intelligentes. Nous tenons également à préciser notre intérêt pour les approches distribuées. Cela revient à dire que le contrôle ne doit pas être centralisé au sein d'une nouvelle entité ou une autre entité déjà existante. Chaque entité participe collégialement à l'ensemble.

1.3 Contributions

Selon Reynaud [81], l'environnement urbain est un environnement partiellement observable, fortement multi-agent, stochastique, séquentiel, dynamique, continu et inconnu. Les environnements de ce type figurent parmi les environnements les plus complexes à modéliser et à étudier. Ainsi, le type de simulation que nous envisageons dans le cadre applicatif de cette thèse, en raison de la complexité de l'environnement étudié, représente déjà à lui seul un intérêt de recherche et un défi à part entière.

Nous avons décrit dans la section 1.2 précédente deux aspects dans la mise en œuvre de la ville intelligente et plus précisément de la mobilité intelligente, sur lesquels nous nous sommes penchés. Nous synthétisons dans cette section nos contributions relatives à ces aspects.

1.3.1 L'environnement comme support d'interaction : l'environnement temporel et le modèle AGRET

Un des supports les plus communément utilisés pour faire interagir les agents dans les SMA est l'environnement. La relation entre un agent et son environnement se traduit par le fait qu'un agent perçoit son environnement grâce à des capteurs et agit sur celui-ci grâce à des effecteurs. Cela se fait selon un modèle d'interaction. Dans les simulations multi-agent, selon Odell et al. [68], l'environnement se classe en deux types : l'environnement physique et l'environnement de communication.

L'environnement physique fournit les principes et processus qui gouvernent et supportent l'existence physique d'une population d'entités. Il permet alors le partage et la perception d'informations spatiales.

L'environnement de communication comme son nom l'indique fournit les principes, processus et structures qui permettent aux agents de transmettre des informations via une infrastructure [68]. L'environnement social est un type particulier d'environnement de communication. Il offre les bases essentielles pour les interactions, les coutumes, les normes, les valeurs, les engagements, les dépendances, etc., qui constituent une société d'agents. Ces principes peuvent être des langages de communications, des protocoles d'interactions, des stratégies de coordination, des politiques sociales, des cultures, etc.

Ces deux types d'environnements permettent de gérer deux dimensions de l'information : la dimension spatiale et la dimension sociale. Cependant, nous pensons qu'il existe une troisième dimension de l'information qui, dans notre contexte, est aussi importante que ces deux citées précédemment : le temps. La dimension temporelle a longtemps été négligée et très peu prise en compte en tant que dynamique du modèle dans les SMA, et ce malgré son importance au même titre que la dimension spatiale ou la dimension sociale. Pourtant, comme nous l'avons expliqué un peu plus haut, le temps est d'une importance capitale dans notre système. Par exemple : le temps est une ressource à partager et à optimiser, le emploi du temps est composé d'activités situées dans l'espace et dans le temps, etc. Le temps devient alors une contrainte importante du système. Par ailleurs, si nous revenons sur l'application au domaine de la ville intelligente, Rolland-May [83] précise bien que l'étude d'un système territorial doit intégrer simultanément trois dimensions :

— la dimension temporelle ;
— la dimension spatiale ;
— la dimension organisationnelle (sociale).

À cela, Nigon et al. [64] rajoute que les différentes caractéristiques des villes intelligentes partagent un trait commun : elles dépendent toutes du contexte. L'accès ou la prévision de données contextuelles et l'extraction d'informations pertinentes à partir de celles-ci sont toujours la clé pour faire progresser l'efficacité des fonctionnalités des villes intelligentes. Dans les plateformes d'exécution SMA, ce contexte occupe une place capitale dans le cycle d'activation de l'agent. En effet, ce cycle consiste en trois phases : la perception du contexte, la décision et la phase d'action. De manière générale, le contexte est perçu au niveau des environnements. La plupart du temps, ce contexte perçu est spatial et/ou social. À notre connaissance, la perception du contexte temporel par les agents n'est pas une pratique courante dans les SMA.

Notre première contribution consiste donc à faire évoluer le paradigme des simulations multi-agent, de manière à considérer le temps comme un nouveau milieu d'interaction, au même titre que l'environnement spatial et l'environnement organisationnel. Nous appelons ce nouveau milieu

d'interaction "environnement temporel". Pour mettre cela en œuvre, nous nous basons sur des modèles existants, communément utilisés dans les SMA. Il s'agit du modèle générique d'organisation AGR (Agent-Groupe-Rôle) et de son extension AGRE (Agent-Groupe-Rôle-Environnement). Nous nous appuyons sur ces derniers pour proposer une nouvelle extension que nous appelons AGRET (Agent-Groupe-Rôle-Environnement-Temps). L'originalité du modèle AGRET est qu'il intègre simultanément les trois dimensions : spatiale, temporelle et organisationnelle en tant qu'environnement. Cette contribution répond à la problématique du besoin de support d'interaction pour échanger des informations spatiales, temporelles et organisationnelles.

1.3.2 Un raisonnement anticipatif basé sur le modèle AGRET

La mise en place de l'environnement temporelle offre de nouvelles possibilités. L'une d'entre elles est la prise en compte des informations échangées au niveau du raisonnement de l'agent. Plus particulièrement, elle permet l'exploitation des informations temporelles au niveau du raisonnement de l'agent. Nous nous intéressons particulièrement au raisonnement anticipatif. Ce raisonnement anticipatif permet à chaque agent de remettre en question et d'optimiser ses décisions présentes et futures. L'objectif est que chacun puisse choisir un comportement qui tende à satisfaire aux objectifs individuels et collectifs fixés.

Ce raisonnement anticipatif se base sur des informations sur le passé, sur le présent et sur le futur planifié, que l'agent perçoit au niveau de l'environnement temporel, l'environnement spatial et l'environnement social. La prise en compte des informations futures dans le raisonnement anticipatif constitue une originalité de cette approche. Cette fonctionnalité est permise par l'environnement temporel qui rend possible le stockage et la perception des informations sur la dimension temporelle.

Pour mettre en oeuvre ce raisonnement anticipatif, nous nous basons sur les approches d'anticipations existantes [25] [81]. Ces approches intègrent, la plupart du temps, un modèle prédictif et un modèle de décision, à l'intérieur de l'agent. Dans beaucoup de modèles existants [25] [81], les agents n'ont aucune information sur ce que les autres agents du système prévoient de faire. Ils sont donc contraints d'essayer de deviner ce qui va se passer dans le futur en réalisant, par exemple, des microsimulations. Nous proposons donc d'améliorer ce système en introduisant un environnement temporel. Ce dernier offre une nouvelle dimension d'expression et de partage entre les agents. La nature temporelle de cette dimension a vocation de capter les projets individuels et de les diffuser sur le collectif. L'analyse prédictive réalisée par les agents peut alors opérer par perception de ce nouvel environnement commun, plutôt que par des calculs de simulation reproduits par chacun d'eux.

Notre deuxième contribution consiste alors à se servir de l'échange d'informations spatiale, temporelle et sociale qui est permis par le modèle AGRET comme substrat dans le cadre d'un raisonnement anticipatif.

1.4 Plan de la thèse

Ce manuscrit est structuré comme suit :

Le chapitre 2 présente les principes de bases de cette thèse. Nous y abordons notamment les problèmes liés à la faible prise en compte de la dimension temps dans les simulations multi-agent. Nous axons notre réflexion sur deux aspects du temps simulé : la représentation du temps et sur le raisonnement temporel, plus particulièrement sur l'anticipation.

Le chapitre 3 introduit notre première contribution : une approche multi-environnement temporelle, spatiale et organisationnelle. L'objectif est de faire évoluer le paradigme agent, de manière à considérer le temps comme un nouveau milieu d'interaction, au même titre que l'environnement spatial et l'environnement social. Ce milieu s'appelle l'environnement temporel. Le tout est intégré au sein d'un modèle nommé AGRET, est une extension du modèle générique d'organisation AGR et de sa variante intégrant l'environnement spatial AGRE.

La deuxième contribution de la thèse s'appuie sur la première. Nous nous servons de ce nouveau système doté d'un environnement temporel comme substrat afin de répondre à une problématique d'anticipation. Nous décrivons cela dans le chapitre 4.

Dans le chapitre 5, nous proposons une implémentation de notre solution sur la plateforme SimSKUAD. En complément à cela, nous présentons, dans le chapitre 6, une évaluation de nos contributions du point de vue de l'utilisateur.

Nous terminons ce manuscrit avec la conclusion dans le chapitre 7. Nous y reviendrons sur les grandes lignes des contributions de la thèse ainsi que les différentes perspectives dans le cadre de la continuité de nos travaux.

Chapitre 2

Considérations spatiale, organisationnelle et temporelle dans les simulations multi-agent pour les villes intelligentes

Ce chapitre présente les notions de base relatives aux problématiques principales de cette thèse. Nous nous intéressons aux simulations multi-agent pour la modélisation et la planification de villes intelligentes. Plus particulièrement, nous nous focalisons sur les simulations de flux de déplacement de véhicules électriques dans le cadre de la mobilité intelligente. Dans le chapitre 1, nous sommes partis de l'exemple du rechargement de véhicules électriques pour dégager la problématique principale ainsi que les différents besoins sur lesquels nous axons nos travaux. Nous avons choisi de nous focaliser sur l'anticipation. Nous avons notamment détecté un besoin en prise en compte de la dimension future du temps dans le raisonnement anticipatif de l'agent. Ce besoin est lié à un autre besoin : un besoin en support d'interaction permettant l'échange d'informations sur les trois dimensions du temps : passé, présent et futur.

Nous commençons par introduire dans la section 2.1 les différents concepts de bases sur lesquels se reposent nos propositions. Il s'agit de la ville intelligente (2.1.1) et la simulation multi-agent (2.1.2). Nous montrons notamment l'intérêt d'utiliser une approche multi-agent dans les villes intelligentes. Nous montrons également l'importance de la dimension temporelle au même titre que la dimension spatiale dans ce contexte. Nous décrivons par la suite, dans 2.1.3, les particularités de l'environnement urbain qui constitue notre cadre applicatif. Cela justifie notre choix de nous pencher sur l'étude du raisonnement anticipatif au sein de l'agent dans les simulations multi-agent. Nous abordons cet aspect dans 2.1.4.

Dans 2.2, nous nous focalisons un peu plus en détail sur la prise en compte des dimensions spatiale, organisationnelle et temporelle dans les simulations multi-agent. Il s'agit des trois dimensions que doit simultanément intégrer l'étude d'un système territorial selon Rolland-May [83]. Dans ce cadre, nous commençons, dans 2.2.1, par une comparaison entre la dimension spatiale et la dimension temporelle afin d'en ressortir l'intérêt de les considérer au même titre. Une fois cela démontré, nous effectuons dans 2.2.2, une analyse plus poussée de la prise en compte des trois dimensions : spatiale, organisationnelle et temporelle, dans les simulations multi-agent. Nous démontrons que comparée aux deux autres dimensions, l'importance de la dimension temporelle, en tant que dynamique du modèle, n'est pas toujours facile à cerner. Il s'agit d'une dimension qui est souvent négligée. Nous nous concentrons alors plus sur deux aspects de cette dimension temporelle : la représentation du temps et le raisonnement temporel. Nous nous focalisons sur l'anticipation qui est un type particulier de raisonnement temporel lié à la position future dans la représentation linéaire du temps. Nous menons alors une analyse de différentes approches classiques communément utilisées dans la littérature. Nous nous intéressons particulièrement à la modélisation de l'environnement (2.2.2.1), à l'ordonnancement de la simulation, aux modèles d'interactions permettant la relation entre l'agent et l'environnement et aux modèles d'organisation (2.2.2.2). L'objectif est d'identifier les problèmes ouverts et de définir les méthodes pour les lever afin de proposer un ensemble de solutions qui tient compte des propriétés et caractéristiques du cadre applicatif : la

Domaine clé		Aspect associé
Français	Anglais	
Économie intelligente	Smart economy	Industrie
Mobilité intelligente	Smart mobility	Logistiques et infrastructures
Environnement durable	Smart environment	Efficacité et développement durable
Éco citoyen	Smart people	Education
Habitat intelligent	Smart living	Sécurité et qualité de vie
Gouvernance durable	Smart governance	E-démocratie

TABLE 2.1 – 6 Domaines clés de la ville intelligente selon Giffinger [40]

ville intelligente. Ces propositions sont introduites dans 2.3 où nous terminons le chapitre par un résumé du positionnement de nos contributions.

2.1 Les simulations multi-agent pour les villes intelligentes

2.1.1 La ville intelligente (smart city)

La ville intelligente est une solution aux problèmes d'urbanisation croissante s'accompagnant de crises économiques et environnementales auxquelles sont sujettes les villes actuellement. Le concept se base sur l'utilisation des TIC [28] pour une gouvernance participative et une gestion plus éclairée des ressources limitées dans l'espace et dans le temps. Cela doit permettre de combiner compétitivité et développement durable. Ces deux notions : participative et optimisation de la gestion des ressources limitées dans l'espace et le temps sont deux notions importantes sur lesquelles nous basons nos choix et nos exemples.

L'expression "smart city" a commencé à prendre de l'ampleur dans les années 90. À cette époque, les chercheurs mettaient beaucoup l'accent sur l'usage de la technologie, sur l'innovation et sur le rôle que peut jouer la mondialisation dans le processus d'urbanisation [39]. À ce jour, il n'existe pas encore de standard permettant de définir de façon universelle ce qu'est une ville intelligente ou "smart city". Néanmoins, nous pouvons retrouver dans la littérature quelques travaux d'analyse comparative de systèmes d'indicateurs de villes dites intelligentes et de classifications de différentes définitions [56] [97]. Nous remarquons cependant que beaucoup de ces définitions négligent la place des citoyens dans le système [43]. Au mieux, ces derniers y sont mentionnés marginalement : en tant que consommateurs, dont les habitudes sont scrutées et régentées par des systèmes techniques. Pourtant, nous pensons que les citoyens occupent une place centrale dans la mise en œuvre d'une ville intelligente. L'intelligence dont il est sujet ici est notamment une forme d'intelligence collective qui émerge des interactions entre les citoyens et le système. Dans ce contexte, les nouveaux outils numériques facilitent la participation des citoyens à l'élaboration des politiques, à la planification et à la budgétisation. Cela pourrait aider les villes à prendre des décisions plus intelligentes et plus démocratiques [86]. Dans ce contexte, les modèles de simulation que nous développons servent d'outils d'aide à la décision. Ils permettent aux décideurs d'évaluer les projets en relation avec la ville intelligente et leurs possibles conséquences, en amont de leur réalisation.

La participation des citoyens est un aspect clé dans la ville intelligente. Ces citoyens échangent des informations et adoptent des comportements qui tendent à satisfaire des objectifs collectifs. Nous pensons alors que l'une des définitions les plus complètes de ce qu'est une ville intelligente est celle donnée par Giffinger [40], dans le sens où elle implique le citoyen dans le processus. Giffinger divise la ville intelligente en six (6) domaines clés. Chaque domaine est associé à un aspect de la vie urbaine comme le montre le tableau 2.1. Bien qu'ils soient distincts, ces 6 domaines sont interconnectés entre eux et constituent un ensemble cohérent de critères que la ville se doit de remplir pour que l'on puisse la qualifier de "smart" ou intelligente.

Cette thèse se focalise sur la mobilité intelligente. Plus précisément, nous appliquons nos contributions à la simulation de flux de déplacement de transports individuels en véhicules électriques (cf Chapitre 5). Néanmoins, les solutions proposées sont assez génériques pour être applicables et adaptables aux autres domaines clés.

Comme mentionné dans le chapitre 1 précédent, nous nous intéressons particulièrement aux simulations multi-agent. La simulation multi-agent est une approche prometteuse pour la modélisation et la simulation de villes intelligentes [42] [57] [84] [63]. Elle utilise une logique ascendante et

rentre parfaitement dans le cadre des approches utilisant l'intelligence collective. Ses principes sont tout à fait en accord avec les principes de base de la ville intelligente que nous avons énoncés plus haut. Dans la sous-section 2.1.2 suivante, nous abordons plus en détail ce qu'est une simulation multi-agent, comment nous l'appliquons à notre contexte et quelles sont les problématiques qui s'y rapportent.

2.1.2 La simulation multi-agent

Drogoul [26] définit une simulation comme une démarche scientifique qui consiste à réaliser une représentation artificielle, appelée modèle, d'un phénomène réel que l'on désire étudier. La simulation permet d'observer le comportement de cette reproduction en faisant varier certains paramètres. Elle permet également d'induire ce qui se passerait dans la réalité sous l'influence de variations analogues. Une simulation a donc pour objet d'imiter les phénomènes qui se produisent au cours du temps dans un système ou un processus du monde réel [7]. Pour cela, le simulateur produit une évolution du modèle en y faisant s'écouler virtuellement le temps [70]. Cette thèse s'intéresse particulièrement à la simulation multi-agent qui est un type particulier de simulation utilisant les méthodes et algorithmes des SMA.

Un SMA est un ensemble de processus autonomes et organisés appelés agents, plongés dans un environnement commun et capables d'interagir afin de réaliser des tâches complexes [16]. Il est notamment composé des éléments suivants [29] :

- Un environnement E, c'est-à-dire un espace disposant généralement d'une métrique ;
- Un ensemble d'objets O. Ces objets sont situés. C'est-à-dire que, pour tout objet, il est possible, à un moment donné, d'associer une position dans E. Ces objets sont passifs. Ils peuvent être perçus, créés, détruits et modifiés par les agents ;
- Un ensemble A d'agents, qui sont des objets particuliers ($A \subseteq O$), lesquels représentent les entités actives du système ;
- Un ensemble de relations R qui unissent des objets (et donc des agents) entre eux ;
- Un ensemble d'opérations Op permettant aux agents de A de percevoir, produire, consommer, transformer et manipuler des objets de O ;
- Des opérateurs chargés de représenter l'application de ces opérations et la réaction du monde à cette tentative de modification, que l'on appelle les lois de l'univers.

Grâce à sa capacité à traiter des types d'agents très différents, la simulation multi-agent touche un large spectre de domaines d'application. Dans notre contexte, nous l'appliquons au domaine de la ville intelligente et plus particulièrement à celui de la mobilité intelligente.

Selon Roscia et al. [84], les SMA permettent de modéliser la non-linéarité et favorisent l'émergence des normes et des protocoles nécessaires pour les villes intelligentes. La littérature le démontre à travers plusieurs travaux de recherches [42] [57] [84] et plus particulièrement, dans le domaine de la simulation de flux de mobilité dans les transports dans les villes [89] [52] [101].

Dans ce cadre, une vie humaine consiste en une succession d'activités telles que le travail, les activités de consommation, la pratique de loisirs, etc [54]. Ces activités sont situées dans l'espace et dans le temps et y sont souvent contraintes. Cela forme l'emploi du temps journalier et est permis par des déplacements utilisant des moyens de transport et de communication. **Le temps tout comme l'espace joue alors un rôle important dans cette mobilité quotidienne**.

Comme mentionné dans le chapitre 1 précédent, nous implémentons nos travaux sur un modèle de simulation de flux de déplacement de véhicules électriques sur le territoire appelé SkuadCityModel, basé sur un autre modèle appelé SmartCityModel.

Les modèles de simulation multi-agent SmartCityModel et SkuadCityModel. SmartCityModel est un modèle de simulation multi-agent implémenté sur Repast Simphony [66]. Il est utilisé dans le cadre de différentes études de cas autour des systèmes urbains. Il a déjà été utilisé, par exemple, pour simuler les interactions entre l'utilisation des sols, les transports et la demande de recharge des véhicules électriques [10], la demande résidentielle en électricité et en chaleur [11]. Il a aussi été utilisé pour estimer la flexibilité de la demande de recharge des véhicules électriques [9] et pour modéliser l'utilisation des infrastructures d'eau et d'assainissement dans les environnements urbains [1]. Dans le cadre de cette thèse, SmartCityModel est utilisé pour simuler les flux de déplacement de transport en voitures électriques, en fonction du calendrier d'activité des agents afin de mieux gérer les infrastructures de recharge. Ces travaux ont fait l'objet d'une première publication en collaboration avec les chercheurs de l'Imperial College London (ICL) [76]. Cependant,

afin d'être en adéquation avec les stratégies et les objectifs de notre équipe de recherche, nous avons décidé de basculer sur une implémentation sur SKUAD. En effet, notre équipe s'intéresse aux simulations hybrides. Nous projetons de mener des expérimentations et de mettre en oeuvre nos travaux de recherche en milieu réel. Nous ne comptons donc pas rester uniquement dans un contexte simulé. Dans ce cadre, SKUAD nous semble plus intéressante dans le sens où c'est une plateforme bimodale que nous développons en interne. Cela veut dire que SKUAD fonctionne à la fois en mode simulé et en milieu réel. Contrairement à cela, Repast Simphony fonctionne uniquement en mode simulé. Toutefois, une réimplémentation de nos contributions sur Repast Simphony, dans le cadre de SmartCityModel, reste une perspective. Cela permettrait de démontrer que nos propositions sont assez génériques et indépendantes de la plateforme de simulation.

Le mode simulé de SKUAD est appelé SimSKUAD. SkuadCityModel est une implémentation de SmartCityModel sur SimSKUAD [3]. Dans la première version du modèle, les agents sont des conducteurs de véhicule. Les véhicules et les bornes de recharge électrique sont des objets de l'environnement. Comme dans la réalité, les automobilistes possèdent un emploi du temps et adoptent des comportements qui tendent à réaliser les activités (travail, loisir, rechargement, etc.) qui y sont prévues.

Definition 2.1.1. Emploi du temps

L'emploi du temps des agents est composé d'un ensemble d'activités qui peuvent être de deux types :

- Les activités cycliques comme le travail, la pratique de loisir, etc.
- Les activités ponctuelles dont certaines viennent "perturber" l'enchaînement des activités qui sont déjà prévues dans l'emploi du temps. Par exemple : le rechargement de son véhicule électrique lorsque le niveau de batterie atteint un certain seuil.

Le modèle de simulation utilise une approche basée sur les activités (activity-based approach) [49]. Cela veut dire que les déplacements résultent de la demande en activités personnelles que les individus doivent ou souhaitent effectuer. Ces déplacements sont faits par le moyen de véhicules dont les voitures électriques. Une voiture électrique dispose d'une charge de batterie limitée. De plus, le nombre de bornes de recharge publiques est fortement inférieur au nombre de véhicules. La disponibilité d'une borne de recharge à tout moment (dans le temps) et à tout endroit (dans l'espace) n'est donc pas garantie. Le rechargement devient alors une contrainte.

Dans ces types de modèles de simulations basés sur les activités, l'emploi du temps contient des informations sur le contexte d'activation temporel de l'agent. D'une manière générale, l'emploi du temps contient l'ensemble des informations temporelles qui permettent de définir la dynamique d'activation temporelle des agents. La plupart du temps, ces informations sont exploitées uniquement pour la gestion du cycle d'activation de la simulation et sont accessibles uniquement par l'ordonnanceur de la simulation. Nous décrivons son fonctionnement dans 2.2.2.2. Nous pensons que ces informations temporelles pourraient être exploitées, d'une manière plus poussée, au niveau du raisonnement temporel de l'agent. Dans ce cadre, différents problèmes très souvent issus du milieu industriel sont associés au raisonnement temporel : la planification, l'ordonnancement de tâches, le suivi de processus, le diagnostic, la prédiction. Pour des raisons que nous évoquons dans la sous-section 2.1.3 suivante, nous choisissons de nous focaliser sur l'anticipation.

2.1.3 L'environnement urbain et la simulation multi-agent

L'environnement urbain qui constitue notre cadre applicatif possède des caractéristiques spécifiques. Dans les simulations multi-agent, ces caractéristiques de l'environnement sont d'une importance centrale puisqu'elles posent les premières contraintes du modèle de simulation [81]. Selon la classification établie par Russell and Norvig [85] et Reynaud [81], un environnement urbain possède les caractéristiques suivantes :

1. **Un environnement urbain est partiellement inconnu** : en fonction de l'angle de vue, l'environnement urbain peut être considéré comme partiellement inconnu. Cela signifie que l'agent ne connaît ni les conséquences exactes de ses actions ni son fonctionnement. Néanmoins, il est possible d'anticiper et de prédire l'état de l'environnement et de l'agent ainsi que les conséquences des actions de ce dernier. Pour ce faire, l'agent doit être doté d'une capacité d'anticipation.

2. **Un environnement urbain est partiellement observable** : dans une ville, l'agent n'a pas accès à la totalité de l'état de l'environnement. Ses capacités de perception sont limitées.

Par exemple, ses capteurs ne lui permettent de voir que des objets qui se trouvent à une certaine distance de lui, de n'entendre que les sons provenant d'une source suffisamment proche de sa position. Ainsi, intégrer une capacité à anticiper ce qu'il ne peut pas percevoir ou ce qu'il ne connaît pas lui permettra d'optimiser son comportement. Dans SmartCityModel [8] et SkuadCityModel [3], la capacité d'anticipation des agents se limite à la détermination du trajet correspondant au chemin le plus court allant d'un point à un autre. Un des objectifs de cette thèse consiste à améliorer ce raisonnement anticipatif au niveau des agents du système afin de le rendre plus réaliste.

3. **Un environnement urbain est stochastique** : du point de vue d'un agent, l'état courant et l'action qu'il exécute ne suffisent pas pour déterminer l'état suivant de l'environnement. Ce dernier est impacté par d'autres événements non entièrement connaissables par des agents (météo, embouteillages, travaux, etc.), voire par des interactions externes. Pouvoir anticiper ces différents événements permettrait d'optimiser le comportement des agents.

4. **Un environnement urbain est séquentiel** : l'environnement urbain ne peut être réduit en épisodes atomiques pendant lesquels un agent reçoit un percept puis exécute une action unique. De plus, dans ce type d'environnement, la décision courante est susceptible d'affecter les décisions futures : les actions à court-terme peuvent avoir des conséquences sur le long-terme. Les agents vont donc devoir gérer une mémoire d'une manière ou d'une autre. Cette mémoire n'est pas forcément intégrée au sein des agents, elle pourrait être externe. Dans notre proposition, cette mémoire est en partie intégrée au niveau de l'environnement.

5. **Un environnement urbain est fortement dynamique** : l'environnement urbain peut être modifié pendant qu'un agent délibère. La réactivité des agents, ainsi que leur rapidité de délibération deviennent des éléments clés.

6. **Un environnement urbain est à états continus** : l'état interne des agents varie de manière continue, de même que leurs vitesses et leurs positions spatiales et temporelles.

7. **Un environnement urbain est fortement multi-agent** : le concept de ville intelligente est considéré comme étant une solution aux problèmes liés à la forte urbanisation évoqués dans le chapitre 1. Dans ce contexte, le nombre d'agents rentrant en jeu dans une simulation varie considérablement selon l'échelle de territoire considéré. Par exemple, à l'échelle d'une grande ville comme Shanghai ou Pékin, ce nombre peut facilement dépasser les 20 000 000. Les espaces urbains sont densément peuplés. Nous sommes alors dans le cas d'environnements fortement multi-agent. Cela complique la tâche des agents, en posant des questions de coopération, de concurrence et de communication. De plus, la capacité du système à passer à l'échelle une contrainte importante.

À cela nous rajoutons les caractéristiques suivantes :

1. **Un environnement urbain intelligent est fortement instrumenté** : La ville intelligente dispose d'un environnement fortement instrumenté. Cela voudrait dire qu'il est doté d'un ensemble de capteurs, d'actionneurs et d'outils permettant le partage et la récolte de données. Ces données sont larges et concernent différents domaines : le transport, l'éducation, la gouvernance, etc. Dans les simulations multi-agent, elles permettent aux agents de disposer d'un ensemble d'informations riches et variées sur leur contexte d'activation. Une manière de les exploiter consiste à les prendre en compte dans le processus de décision. Cependant, malgré les avantages que cette richesse d'information puisse apporter notamment en termes de réalisme et de précision, cette quantité phénoménale d'information mise à disposition de l'agent peut très vite faire exploser le nombre de choix et de possibilités qui s'offrent à l'agent. Cela peut perturber la capacité de discernement de ce dernier. Dans ce cadre, doter les agents d'une capacité d'anticipation permettrait une prise de décision plus pertinente. En effet, la mise en place d'un raisonnement anticipatif au sein de l'agent optimiserait son choix de comportement. Il n'effectuerait ainsi que les actions véritablement nécessaires [29].

2. **Un environnement urbain intelligent dépend des données** : Actuellement, il n'existe pas encore de standard universel permettant de définir ce qu'est une smart city. Dans leur article [50], Ismagilova et al. dressent un état de l'art des différentes définitions du terme et

des sous-domaines. Plusieurs définitions et principes mettent en avant la place importante qu'occupe l'information dans la ville intelligente. Par exemple, Lee and Lee [55] définissent une ville intelligente comme une ville qui développe et gère une variété de services innovants. Ces derniers fournissent des informations à tous les citoyens sur tous les aspects de la vie urbaine par le biais d'applications interactives et basées sur les TIC. Calderoni et al. [14] la définissent comme un contexte urbain performant, dans lequel les citoyens sont plus conscients et mieux intégrés dans la vie urbaine grâce à un système d'information intelligent [14]. En effet, les villes actuelles deviennent de plus en plus numériques et de plus en plus basées sur l'information [98]. Par conséquent, l'économie et les conditions sociales actuelles dépendent de plus en plus des connaissances et des informations numériques [92]. Il devient alors difficile de parler de smart city, sans aborder les problématiques relatives aux échanges et à l'exploitation des données et des informations. Selon Heppenstall et al. [45], le manque de puissance au niveau des machines et le manque de données ont constitué une entrave au paradigme SMA, jusqu'à l'avènement récent du big data et de la puissance de calcul et de stockage. Dans notre cadre, les données peuvent aider à la compréhension et à la modélisation de la mobilité dans les transports, en tant que système complexe, sur plusieurs niveaux. Par exemple :

— Au niveau de l'**initialisation** : utilisation des données lors de la phase d'initialisation de la simulation pour paramétrer la simulation ;
— Au niveau de la **calibration** et de la **validation** de la simulation : des étapes où actuellement il n'existe pas encore de standard et où les données peuvent être d'une très grande utilité ;
— Au niveau du **comportement des agents** : les données apportent de la précision et améliorent le réalisme des comportements dans un modèle basé sur les individus ;
— Pour l'**extraction de paramètres pertinents** que l'on utilisera dans la simulation à partir de large entrepôt de données.

Dans cette thèse, nous choisissons de nous focaliser particulièrement sur l'utilisation des données au niveau du comportement des agents, afin d'y apporter plus de précision et de réalisme. Ces données sont exploitées par les agents dans le cadre de leur raisonnement anticipatif. En effet, il existe de nombreuses capacités considérées comme cognitives. Cependant, dans notre contexte, l'anticipation parait plus intéressante que les autres. Pezzulo et al. [72] déclarent même que seuls les systèmes cognitifs dotés d'un mécanisme d'anticipation peuvent être crédibles, adaptatifs et interagir correctement avec leur environnement et les autres systèmes autonomes. Nous souhaitons alors intégrer, dans nos agents, des capacités d'anticipation afin d'améliorer la précision et le réalisme de leurs comportements.

2.1.4 L'anticipation

Definition 2.1.2. Anticipation

L'anticipation occupe une place majeure [81] parmi un éventail des processus généralement considérés comme cognitifs et dans lesquels la maîtrise humaine est indéniable. Támos [91] définit un système d'anticipation dans les systèmes artificiels comme un système contenant un modèle prédictif de lui-même et/ou de son environnement qui lui permet de changer son état actuel à un instant selon les modèles de prédictions se rapportant à un instant ultérieur. Selon lui, l'anticipation résulte d'une relation de modélisation qui permet de passer du système réel à son modèle prédictif.

Une définition plus précise est également donnée : "un système d'anticipation S_2 est un système contenant un modèle d'un système S_1 avec lequel il interagit. Ce modèle est un modèle prédictif. Son état présent donne des informations sur des états futurs de S_1. De plus, l'état présent du modèle peut causer un changement d'état dans d'autres sous-systèmes de S_2 ; ces sous-systèmes sont (a) impliqués dans l'interaction de S_2 sur S_1, et (b) ils n'affectent pas le modèle de S_1 (c'est-à-dire qu'ils n'y sont pas liés). En général, on peut considérer les changements d'états de S_2 découlant du modèle comme une adaptation, ou une pré-adaptation, de S_2 par rapport à ses interactions avec S_1".

Butz et al. [12] définissent quatre (4) classes principales d'anticipation : l'anticipation implicite, l'anticipation par récompense, l'anticipation sensorielle et l'anticipation d'état.

L'anticipation implicite. L'anticipation implicite fait référence à un type d'anticipation qui se base sur des comportements préprogrammés inclus directement dans les agents, dans les structures

et les interactions entre les capteurs, les actionneurs, et les algorithmes qui les relient. L'agent ne réalise aucune prédiction. Cependant, le modélisateur construit sa structure comportementale de manière anticipative. Pour cela, il introduit, dans le module décisionnel, des notions permettant à l'agent de réaliser une certaine forme d'anticipation. Le processus décisionnel prend alors directement en compte les stimuli internes et externes. Ce type d'anticipation nous intéresse peu. En effet, compte tenu du nombre élevé de choix possibles pouvant s'offrir à l'agent, il serait compliqué de préprogrammer tous les comportements possibles de manière à anticiper tous les phénomènes qui pourraient se produire dans un environnement urbain comme les villes intelligentes.

L'anticipation d'état. Dans ce type d'anticipation, les prédictions sur les états futurs influencent directement la prise de décision comportementale actuelle. Ce système dispose d'un modèle prédictif et d'un modèle de décision. Il utilise directement une prédiction des états futurs de l'environnement dans le processus de décision. Dans l'approche que nous proposons, nous gardons cette même structuration. Cependant, nous apportons des améliorations au niveau du modèle prédictif.

Ce type d'anticipation est souvent utilisé pour éviter des états futurs de l'environnement considérés comme indésirables [22]. Beaucoup de travaux dans le domaine des SMA utilisent cette classe d'anticipation. Un exemple est l'approche utilisée par Doniec et al. [25] qui se focalise sur une architecture d'agents anticipateurs proposée par Davidsson [22]. Elle comporte deux couches : une couche d'anticipation et une couche réactive. Ces deux couches fonctionnent de manière concurrente. La couche d'anticipation utilise un modèle de l'environnement ainsi qu'un anticipateur. Ils permettent de simuler plus rapidement l'environnement et de générer des prédictions. Lorsque l'anticipateur détecte un état indésirable, il agit sur la couche réactive pour modifier le comportement de l'agent. Au niveau de l'agent, ce concept d'anticipation implique alors un processus de raisonnement en deux phases : une phase de prédiction de l'état futur du SMA et une phase d'analyse de ces prédictions afin que l'agent puisse modifier son comportement actuel. Le système et son modèle prédictif n'avancent pas sur la même échelle de temps. En effet, le modèle prédictif avance plus rapidement de manière à ce que son observation permette de prédire l'état dynamique du système. Le processus de décision consiste à partitionner l'espace des états du système en deux régions [91]. L'une d'entre elles regroupe les états désirés et l'autre les états non désirés. Tant que la prise de décision du modèle permet de rester dans un état désiré, aucune action n'est prise par le modèle au travers des effecteurs. Dès que la prise de décision conduit le modèle dans un état non désiré, les effecteurs sont activés de manière à changer la dynamique du système pour que le système ne se retrouve pas dans un état non désiré. Une autre approche, proposée par Reynaud [81] utilise un modèle de décision plus poussé, basé sur la composition de comportements. Dans son implémentation il utilise le même modèle de décision que celui véritablement utilisé par la simulation. De même, le modèle prédictif utilisé est le modèle de simulation lui-même. Ce type d'approche est intéressant, car il présente divers avantages notamment en termes de réalisme et de précision. Cependant, sa principale limite est son coût élevé en temps de calcul et la gourmandise en ressources du modèle de prédiction.

L'anticipation par récompense (payoff). L'anticipation par récompense (payoff) est une approche dans laquelle l'agent prend en compte dans son processus décisionnel des prédictions concernant les récompenses ou bénéfices des différentes actions qu'il est possible d'exécuter. Le système estime ainsi le bénéfice attendu pour chaque action possible et utilise une fonction de maximisation comme critère de décision. L'influence des prédictions futures sur le comportement se limite aux prévisions des récompenses. Les systèmes de classeurs (anticipatory classifier system) [46], ainsi que les systèmes d'apprentissage par renforcement [88] [53] sont des exemples d'utilisation typique de ces prédictions. Il s'agit d'une stratégie d'anticipation basique dans le sens où elle n'envisage que les effets des actions [81]. Dans notre approche, nous nous inspirons de quelques concepts utilisés dans ce type d'approche : notamment la prise en compte des prévisions futures dans le processus décisionnel et l'utilisation d'une fonction de maximisation (dans notre cas : une fonction de minimisation des coûts) comme critère de décision.

L'anticipation sensorielle L'anticipation sensorielle est une classe d'anticipation où les prédictions futures influencent le (pré-)traitement sensoriel. Les prédictions sont étendues aux états futurs de l'environnement, ainsi qu'aux futurs stimuli. Ces prédictions n'impactent pas directement le comportement des agents, mais modifient la manière dont les capteurs traitent les perceptions

[34]. Ainsi, les stimuli attendus peuvent être traités plus rapidement que ceux qui sont imprévus. Les prédictions ont une influence sur le comportement en modifiant les processus de perception du système. La prise de décision quant à elle reste uniquement guidée par l'état et les actions antérieurs du système. L'anticipation sensorielle est fortement reliée aux mécanismes d'attention sélective et permet de simuler des états mentaux tels que la curiosité, ou la distraction (par exemple en filtrant un certain nombre de perceptions) [17]. Dans le même style, certains travaux se focalisent sur les phénomènes de surprise et prévision pour gérer l'anticipation [73]. Les agents hystérétiques utilisent leur expérience passée pour anticiper sur l'avenir [29]. À partir des perceptions ou des informations portant sur l'état actuel du monde, des modèles des actions que l'agent peut accomplir et des lois du monde auxquelles il croit et des croyances factuelles supplémentaires, le système prévisionnel fournit des attentes. En d'autres termes, des percepts ou informations qui devront "normalement" être corroborés par les percepts et informations provenant du système perceptif. La possibilité de modification du processus de perception du système nous semble être une fonctionnalité intéressante. Nous nous en inspirons dans l'approche que nous proposons.

2.2 Considérations spatiale, organisationnelle et temporelle

L'exploitation des informations spatiales ou organisationnelles est classique dans le raisonnement d'anticipatif dans les simulations multi-agent. Dans SmartCityModel et SkuadCityModel, cette démarche consiste par exemple à utiliser la minimisation de la distance prévisionnelle entre la localisation actuelle du véhicule et de sa destination comme critère de choix entre plusieurs destinations possibles. Comme nous sommes dans le cadre de l'application aux déplacements de véhicules électriques, la pente est également un critère important, car elle impacte sur la consommation du véhicule électrique. Un autre critère de choix consiste donc à prendre en compte la valeur prévisionnelle de la pente dans le mécanisme de décision.

À travers cet exemple, nous illustrons la prise en compte des trois (3) dimensions de l'environnement spatial : abscisse, ordonnée (distance) et hauteur (pente) dans le processus d'anticipation. Cette considération semble très naturelle dans le sens où elle se fait automatiquement sans que nous y prêtions vraiment attention.

Maintenant, si nous faisons le parallèle avec le temps : la conception du temps linéaire que nous expliquons plus en détail dans 2.2.2.2, définie trois (3) positions de la dimension temporelle : le passé, le présent et le futur. Cependant, si la prise en compte des informations sur les trois dimensions de l'espace dans le raisonnement anticipatif dans les simulations multi-agent semble intuitive, celle des trois dimensions du temps l'est moins. À notre connaissance, la majorité des mécanismes d'anticipation utilise dans leur modèle prédictif des informations positionnées uniquement sur le passé et sur le présent. Ces informations sont utilisées pour générer des prédictions.

Pourtant, la prise en compte des informations sur les trois (3) dimensions du temps : le passé, le présent et le futur planifié dans le raisonnement anticipatif est naturelle chez l'humain. Les exemples suivants illustrent ce processus :

- Certaines de nos activités comme le fait d'aller au travail ou à l'école sont cycliques/répétitives. Cela nous permet déjà d'avoir une visibilité sur les possibles occurrences futures de ces activités et de les prendre en compte dans notre raisonnement.
- Depuis l'avènement des réseaux sociaux, nous avons accès à diverses informations sur le passé, sur le présent et sur le futur planifié, partagées par d'autres personnes. Si nous prenons l'exemple de Facebook, nous pouvons par exemple connaître à peu près combien de personnes et quelles personnes ont répondu "sera présent" lors d'un événement qui se déroulera prochainement. Nous avons également accès à des retours d'expériences sur des activités passées, etc.
- La plupart d'entre nous avons un emploi du temps qui nous permet d'avoir plus ou moins connaissance de nos projets individuels, de ce que nous devons faire ou ce que nous prévoyons de faire dans le futur. Nous pouvons également avoir connaissance des projets de nos proches. Afin de ne pas l'oublier et pour s'y conformer, nous notons notre emploi du temps dans un agenda par exemple. Cet agenda nous donne concrètement une visibilité sur nos activités passées, présentes, et futures planifiées. Nous prenons en compte toutes ces informations dans notre raisonnement.

Ainsi, dans la réalité, nous prenons en compte naturellement dans notre raisonnement, toutes ces informations stockées, partagées ou échangées sur le passé, le présent et le futur planifié. Cependant, dans une simulation multi-agent, le déroulement du même processus est loin de se conformer

à cette réalité. Dans le contexte actuel des simulations multi-agent, du point de vue de l'agent, la prise en compte des informations positionnées sur le passé et le présent est très courante. Contrairement à cela, la prise en compte de celles positionnées sur le futur, planifié par les agents, comme les projets individuels des agents du système, l'est moins. En effet, la plupart du temps, les informations positionnées sur le futur sont souvent considérées comme inexistantes. Dans certains cas, elles existent, mais aucun support n'est prévu pour permettre à l'agent de les percevoir. L'agent ne va donc pas les prendre en compte dans son processus de raisonnement. Son raisonnement anticipatif consiste à rechercher des comportements à réaliser uniquement en fonction des informations qu'il a sur le présent et sur le passé. Il fait comme s'il ne connaissait rien du futur et que les seules informations qu'il pourrait connaître sur le futur sont les prédictions qui résultent de son raisonnement anticipatif basé sur le passé et sur le présent.

Les plans ou prévisions ne sont pas utilisés dans le processus de raisonnement, car aucun support natif n'a été défini pour traiter l'avenir.

Dans d'autres domaines comme l'apprentissage par renforcement en machine learning [88], nous pouvons retrouver des travaux montrant l'utilisation des trois positions du temps (passé, présent et futur) dans le processus d'anticipation ou de planification. Par exemple, dans la modélisation à base de chaîne de Markov, un système dynamique à temps discret $(x_t)_{t\in\mathbb{N}} \in X$, où X est l'espace d'états tel que $\mathbb{P}(x_{t+1} = x|x_t, x_{t-1}, ..., x_0) = \mathbb{P}(x_{t+1} = x|x_t)$. Toute l'information pertinente pour la prédiction du futur est contenue dans l'état présent (propriété de Markov). Ainsi, il est tout à fait possible que l'information pertinente contenue dans l'état présent soit le cumul des informations disponibles sur le passé, le présent et le futur. Cependant, cette notion de futur est différente de celle dont nous nous intéressons dans le cadre de cette thèse. Ce futur est en réalité un futur antérieur. Un futur antérieur qui est certain et qui a déjà eu lieu. En effet, l'apprentissage du modèle est effectué sur la base de données existantes. Ainsi, si nous prenons comme référence l'horloge réelle, toutes ces données se situent sur le présent et sur le passé. Cependant, il est possible de procéder à un changement de référence et de prendre comme référence une information positionnée sur le passé. Dans ce cas, cette position qui est passée par rapport à notre horloge réelle est positionnée sur le présent par rapport à la nouvelle référence. Nous notons cette position t. Ainsi, par rapport à cette nouvelle référence, toutes les informations antérieures à t se situent dans le passé et toutes les informations postérieures à t sont des informations futures. Ces informations futures se distinguent par leur certitude. Par rapport à l'horloge réelle, ce sont des informations sur des événements qui ont déjà eu lieu. Ce sont donc des informations certaines. Par exemple : par rapport à notre horloge réelle, nous avons constaté que dans le passé, l'automobiliste part faire ses courses lorsqu'il constate que son frigo est vide. Si l'on prend comme référence l'instant où le frigo a été vide, l'activité "faire ses courses" est positionnée sur le futur. Il s'agit d'un futur qui est certain, car il a déjà eu lieu.

Contrairement à cela, la notion d'information future à laquelle nous nous intéressons, dans le cadre de cette thèse, est positionnée dans un futur par rapport à notre horloge réelle. Ces informations correspondent à des souhaits, des projets que l'agent souhaite entreprendre, un état que l'agent veut atteindre dans le futur. Contrairement à la notion de futur évoqué précédemment, il s'agit d'informations à titre indicatif donc incertain. Par exemple : l'automobiliste constate que son frigo est vide. Demain, il compte aller au supermarché pour faire ses courses. La réalisation de l'activité "faire les courses" est incertaine. En fonction de l'enchaînement des événements, il y a une probabilité pour que cette activité ne soit pas réalisée. Cependant, l'agent peut déjà prendre en compte cette information future dans son raisonnement anticipatif, à titre indicatif. Il va par exemple planifier un rechargement de son véhicule demain sur une borne publique située au supermarché pendant qu'il fait ses courses.

Dans les SMA, la démarche actuelle qui ignore complètement les échanges d'informations sur cette position future du temps parait absurde. Cependant, elle a toujours été tolérée, car cette connaissance de l'anticipation est portée par les modélisateurs et les concepteurs de modèles. Ces derniers prévoient ce comportement dans les agents, de manière implicite ou par effet de bord, sans que les agents n'aient besoin de l'exprimer. Ainsi, cette connaissance qui est mal maîtrisée au sein de la simulation peut introduire un biais et des imprécisions. Ces derniers peuvent être tolérables à petite échelle, mais peuvent devenir critiques à l'échelle d'une ville.

Par conséquent, nous pensons qu'il est important d'introduire dans le modèle simulé une visibilité sur la dimension future du temps sur laquelle pourra s'appuyer un certain potentiel d'anticipation comme c'est le cas dans la vie réelle. Dans beaucoup de modèles de simulation multi-agent, et plus particulièrement dans les modèles basés sur les activités, ces informations sur le futur existent au sein du système, mais ne sont pas accessibles par les agents. En effet, comme mentionné précé-

demment, dans les modèles utilisant une approche basée sur les activités, les agents disposent d'un emploi du temps plus ou moins défini à l'avance par eux même. Cet emploi du temps contient des informations sur les activités passées, présentes, mais également sur les activités futures planifiées par les agents. Cependant, cet emploi du temps n'est pas représenté de manière à ce que les informations qu'il contient soient accessibles par les autres agents du système. Nous pensons alors qu'il serait utile de mettre en place, au niveau du système, un support permettant l'échange (partage et accès) d'informations sur le passé, le présent et le futur planifié par les agents. Le partage et l'accès à ces informations permettraient à l'agent de prendre en compte ces trois positions du temps au niveau de son raisonnement. Cette mise en place d'un support d'interaction temporel au niveau du modèle de simulation multi-agent constitue notre première proposition. Son originalité réside sur le fait qu'elle permet une visibilité sur la dimension future du temps. Nous exploitons cette visibilité sur la dimension future pour améliorer le raisonnement temporel de l'agent, et plus particulièrement son raisonnement anticipatif. Cela constitue notre deuxième contribution.

D'une manière générale, dans les simulations multi-agent, l'espace est modélisé sous forme d'environnement. Cet environnement est appelé environnement physique et permet l'échange d'informations spatiales. En effet, l'environnement physique constitue un support d'interaction permettant aux agents d'agir sur la dimension spatiale et de percevoir des informations sur leur contexte d'activation spatiale. Ces informations spatiales riches, perceptibles par l'agent, peuvent être utilisées au niveau de leur raisonnement et leur permettent, par exemple, d'optimiser leur comportement sur la base d'informations spatiales.

SmartCityModel et SkuadCityModel intègrent une prise en compte très détaillée de l'espace. Ce dernier est modélisé sous forme d'environnement physique défini à partir de données réelles sous forme de Système d'Information Géographique (SIG) et de statistiques. Cet environnement physique est disposé en différentes couches correspondant chacune à un groupe de localisation de zones d'activités. Exemple : une couche correspondant à la localisation des zones résidentielles, une couche correspondant à la localisation des zones industrielles, une couche correspondant à la localisation des zones commerciales, etc. Les agents exploitent ces données, par exemple, pour choisir leur trajectoire et pour éditer leur emploi du temps sur la base de données spatiales réelles.

Face à cette considération naturelle et précise de l'espace, la considération du temps, en tant que dynamique du modèle, reste assez pauvre. En effet, dans les simulations multi-agent, le temps est souvent abordé uniquement sous l'angle technique, en tant que mécanique du système géré par l'ordonnanceur de la simulation. L'ordonnanceur de la simulation se sert des données temporelles pour la gestion du cycle d'activation de la simulation. Dans la plupart des cas, les informations temporelles sont détenues et exploitées exclusivement par l'ordonnanceur de la simulation. Ce fonctionnement ne permet pas l'accès aux informations temporelles par les agents. Nous abordons cet aspect plus en détail dans 2.2.2.2. Nous pensons que cette dimension temporelle pourrait être exploitée de manière plus poussée, au même titre que la dimension spatiale et la dimension sociale.

En effet, un environnement urbain est situé dans l'espace et dans le temps. La ville et les activités humaines occupent également le temps et l'espace. De même, la donnée occupe plusieurs dimensions dont une des plus importantes est la dimension temporelle [4]. Cependant, en analysant les simulations multi-agent utilisées pour modéliser ces villes intelligentes, nous nous rendons compte que si la considération de l'espace en tant que dynamique du modèle semble naturelle, celle du temps l'est moins. La dimension temporelle est souvent sujette à très peu d'attention et de considération, de par sa complexité, mais surtout parce qu'en tant que dynamique du modèle, il n'est pas toujours évident de comprendre son importance comparée à celle de la dimension spatiale [99]. Ainsi, dans les systèmes multi-agent, la notion de la prise en compte du temps est parfois évoquée sans toutefois être au cœur de la problématique. Notre premier objectif est alors de démontrer la nécessité d'intégrer naturellement une dimension temporelle, au même titre que la dimension spatiale dans les simulations multi-agent. Dans ce cadre, nous présentons dans 2.2.1, une étude comparative faisant analogie entre l'espace et le temps.

2.2.1 Analogies et divergences entre le temps et l'espace

Cette section montre quelques similitudes et différences entre le temps et l'espace. L'objectif est de démontrer l'intérêt de prendre en compte le temps au même titre que l'espace dans les simulations multi-agent pour les villes intelligentes.

Dans son livre, Callender [15] cite trois caractéristiques temporelles qui différencient le temps de l'espace : la dimensionnalité unique du temps, la différence de métrique entre le temps et l'espace

et l'absence de ce que nous pourrions appeler la "mobilité libre".

- Dimensionalité : le temps est constitué d'une variable à une dimension avec une topologie plus pauvre que celle de l'espace puisqu'un seul nombre suffit à déterminer une date.
- Différence en termes de métrique : en physique relativiste, la composante du temps se distingue des composantes spatiales par un célèbre signe moins.
- Mobilité : l'une des différences les plus évidentes entre le temps et l'espace est que nous avons une mobilité relativement libre dans les directions spatiales, mais pas dans le temps. Si dans l'espace, on peut se mouvoir vers n'importe quel endroit et revenir au point initial, la position dans le temps est imposée. Au niveau de l'horloge réelle, repasser par le même instant, tout comme rétroagir sur un élément passé, est impossible/interdit. Ces contraintes sont plus souples en simulation. En effet, en fonction des objectifs du modélisateur et de l'utilisateur, la rétroaction sur un élément dans le passé est possible en simulation. Par exemple : dans le cas où la simulation est utilisée pour effectuer des tests afin d'évaluer les conséquences de différents paramétrages ou événements passés sur l'état actuel et l'état futur de la simulation. Il est également possible de rejouer une simulation, c'est-à-dire de revenir dans le passé et repasser par le même instant.

Cependant, malgré ces divergences, espace et temps possèdent des propriétés similaires et peuvent être même complémentaires.

Selon l'étude comparative faite par Schlesinger [87], toutes les propriétés que le temps a en vertu du fait qu'il est un continuum sont également valables pour l'espace puisque l'espace est aussi un continuum. Dans ce contexte, espace et temps possèdent plusieurs propriétés similaires que les continus possèdent en général. Les événements, processus et objets, qui occupent l'espace et le temps, ont également toutes les propriétés qui caractérisent les occupants du continuum en général. Nous citons quelques propriétés communes entre le temps et l'espace :

- **Espace et temps sont discrétisables** : le temps peut être par exemple découpé en pas de temps, l'équivalent dans l'espace peut être la discrétisation en cellules ;
- **La durée dans le temps est similaire à la superficie dans l'espace** : la relation entre le temps occupé par une chose pendant toute son existence et le reste du temps est aussi fixe que la relation entre la région que la chose recouvre pendant toute son existence et le reste de l'espace. Si l'on considère le temps comme une durée, rien ne peut occuper deux temps au même endroit. De même, si l'on considère la position en termes de surface, rien ne peut occuper deux endroits en même temps. Il existe également un horizon temporel et un horizon spatial.
- **Il existe une position dans le temps et une position dans l'espace** : il existe une position dans l'espace, son équivalent dans le temps est aussi une position dans le temps. Ainsi simultané dans le temps équivaut à situé sur le même point dans l'espace.

Espace et temps possèdent également un certain nombre de similitudes en termes de représentation. Ainsi, aux termes à référence spatiale "ici" et "là" peuvent correspondre respectivement dans le temps "maintenant" et "alors"("alors" pouvant faire référence à un temps du passé ou du futur). Tandis que spatialement il existe des relations euclidiennes ou topologiques, le temps fait l'objet d'une mesure quantitative : une échelle d'intervalle ou de rapport selon les cas, et qualitative : une échelle ordinale (avant, pendant, après). Implicitement, le mouvement des aiguilles de nos montres nous incite à assimiler le temps à un flux composé d'instants infiniment proches parcourus les uns après les autres. L'heure lue sur l'horloge lie l'espace au temps par la position spatiale des aiguilles. Alors que les cadrans confortaient une conception cyclique du temps, l'affichage numérique renforce une conception linéaire accentuée encore par la technique puisqu'au remontage quotidien s'oppose la pile. Dans cette représentation spatiale du temps qui en permet la mesure, le temps est constitué d'une variable à une dimension. Sa topologie est plus pauvre que celle de l'espace puisque comme nous avons vu précédemment, un seul nombre suffit à déterminer une date. Selon la conception adoptée, le temps peut être représenté comme une courbe unidimensionnelle orientée qui peut être ramenée soit à un segment de droite, soit à un cercle, exprimant respectivement un temps linéaire ou cyclique. Cette représentation du temps relie fortement ce dernier à l'espace. Ce lien semble même naturel à l'homme. Ainsi, lorsque l'on dit que Saint-Etienne est situé à 3 h de Paris en TGV, l'on se rend compte d'une réalité bien plus riche qu'en faisant simplement allusion aux 400 km qui séparent les deux villes.

Malheureusement, malgré cette complémentarité et cette similarité, dans les SMA, espace et temps ne sont pas considérés de la même manière. L'importance accordée à ces deux dimensions n'est pas la même dans le sens où la dimension temporelle n'est pas explicitement modélisée en

tant que dynamique au niveau du système contrairement à la dimension spatiale. Pourtant temps et espace font partie des trois dimensions occupant une place centrale dans l'étude d'un système territorial comme la ville intelligente [83].

Dans la suite de ce chapitre, nous analysons alors plus en détail la prise en compte de ces trois dimensions : spatiale, organisationnelle et temporelle, dans les simulations multi-agent .

2.2.2 Modélisation de l'espace, des organisations et du temps dans les simulations multi-agent

2.2.2.1 L'environnement

Une manière assez courante de modéliser l'espace et les organisations dans les simulations multi-agent consiste à les représenter sous forme d'environnements. Ces environnements agissent comme des supports d'interaction permettant l'échange d'informations entre les agents et entre les agents et le système.

L'environnement se définit comme étant un "espace" disposant généralement d'une métrique [32]. L'environnement dans les SMA englobe tous les éléments non-agents [96]. Il fournit les conditions sous lesquelles une entité (agent ou objet) existe [68].

De manière générale, l'environnement est classé en deux types [68] :

— L'environnement physique qui est la représentation de l'espace physique. Pour ce type d'environnement, la métrique est spatiale ;

— L'environnement de communication, dont l'environnement social (organisation) en est un cas particulier.

La relation entre un agent et son environnement se traduit par le fait qu'un agent perçoit son environnement grâce à des capteurs et agit sur celui-ci grâce à des effecteurs. Cette relation entre agent et environnement se fait selon un modèle d'interaction. Cela fait donc intervenir deux notions : le milieu d'interaction (l'environnement) et le modèle d'interaction (cf 2.2.2.2).

La dimension organisationnelle : l'environnement de communication. Une façon de prendre en compte la dimension organisationnelle dans les simulations multi-agent est de la modéliser sous la forme d'un environnement. Ce type d'environnement est de type environnement de communication. Il fournit les principes, les processus et les structures qui permettent aux agents de transmettre des informations via une infrastructure [68].

Les principes de l'environnement de communication offrent les bases essentielles pour les interactions, les coutumes, les normes, les valeurs, les engagements, les dépendances, etc., qui constituent une société d'agents. Ces principes peuvent être des langages de communication, des protocoles d'interaction, des stratégies de coordination, des politiques sociales, des cultures, etc.

Dans le cadre de cette thèse, notre proposition repose sur un cas particulier d'environnement de communication qui est l'environnement social. Il s'agit d'un environnement dans lequel les agents interagissent de manière coordonnée [68] selon un modèle d'organisations.

La dimension spatiale : l'environnement physique. Dans beaucoup de simulations multi-agent, la dimension spatiale est modélisée, sous la forme d'un environnement. Cet environnement est appelé environnement physique. Il fournit les principes et processus qui gouvernent et supportent l'existence physique d'une population d'entités. Un environnement physique s'exprime formellement comme un tuple [69] :

$$Environnement =< Etat_e, process_e > \tag{2.1}$$

Où $Etat_e$ est un ensemble de valeurs qui définissent complètement l'environnement. Il inclut également les agents et les objets qui y sont présents. $process_e$ est un mécanisme autonome qui change l'état $Etat_e$ de l'environnement. Ces processus servent à mettre en œuvre le principe environnemental et le rendent actif. C'est-à-dire que l'environnement a son propre processus et peut changer d'état indépendamment des actions des agents qui y sont situés. Ils regroupent un ensemble de règles, contraintes et politiques qui régissent et supportent l'existence physique des agents et des objets. Par exemple, le champ gravitationnel est un principe qui peut être mis en œuvre avec un processus qui attire les entités d'une manière bien définie. De manière générale, les principes de base d'un environnement physique peuvent inclure [95] [85] :

— L'accessibilité : Dans quelle mesure l'environnement est-il connu et disponible pour l'agent ?

— Le déterminisme : Dans quelle mesure l'agent peut-il prédire des événements dans l'environnement ?
— La diversité : Dans quelle mesure les entités de l'environnement sont-elles homogènes ou hétérogènes ?
— La contrôlabilité : Dans quelle mesure l'agent peut-il modifier son environnement ?
— la volatilité : Dans quelle mesure l'environnement peut-il changer pendant que l'agent délibère ?
— La temporalité : Le temps est-il divisé de manière clairement définie ?
— La localité : L'agent a-t-il un emplacement distinct dans l'environnement qui peut ou peut ne pas être identique à l'emplacement d'autres agents partageant le même environnement ? Comment est-elle exprimée (par exemple, système de coordonnées, métrique de distance, positionnement relatif) ?

La modélisation de la dimension spatiale et de la dimension sociale sous forme d'environnements permet l'échange et le stockage d'informations spatiales et sociales. Les agents peuvent alors percevoir leur contexte d'activation spatiale et leur contexte d'activation sociale. Par conséquent, ils peuvent également, prendre en compte les informations spatiales et sociales dans leur raisonnement anticipatif.

2.2.2.2 L'ordonnancement de la simulation

Zeigler [100] définit formellement une base de temps comme un ensemble ordonné d'indexation d'événements qui modélise le flux du temps réel. Ce temps est appelé le temps logique ou le temps simulé. Il est différent du temps physique qui est le temps astronomique sur lequel nous n'avons aucune emprise. Lorsque nous considérons des événements se produisant dans le monde réel, en temps réel, nous nous référons à une variable de temps telle que mesurée par une horloge réelle. Le temps physique, également appelé temps métrique ou temps mural, est mesuré en ticks d'horloges physiques. Le temps logique ou temps simulé, quant à lui, est mesuré en ticks d'une horloge qui est en quelque sorte intégrée dans un modèle. Dans cette thèse, nous nous intéressons essentiellement au temps simulé.

Une simulation a pour objet d'imiter les phénomènes qui se produisent au cours du temps dans un système ou un processus du monde réel [7]. Pour cela, le simulateur produit une évolution du modèle en y faisant s'écouler virtuellement le temps. Cette mécanique est gérée par une entité du simulateur appelé l'ordonnanceur. Ainsi, **contrairement à l'espace et aux organisations, dans la plupart des simulations multi-agent, le temps simulé n'est pas modélisé comme un milieu d'interaction (environnement), mais comme une simple mécanique du système**. Contrairement aux environnements qui permettent les échanges et le stockage d'informations spatiales et sociales, cette mécanique n'intègre pas de support permettant l'échange et la manipulation d'informations temporelles.

Afin de permettre l'exploitation et la manipulation des informations temporelles par les agents, au même titre que les informations spatiales et sociales, nous décidons de nous pencher sur deux aspects de l'étude du temps simulé selon Carron [16] :

— La représentation du temps ;
— La manipulation du temps ou raisonnement temporel.

La représentation du temps peut être implicite ou explicite et traite de sa topologie, de sa structure ainsi que des approches qui permettent de le modéliser. Le raisonnement temporel quant à lui est lié à la manipulation de ces représentations afin d'effectuer des calculs, conduire des raisonnements à partir de celles-ci. En intelligence artificielle, le raisonnement temporel équivaut à un raisonnement sur ce qui change au cours du temps. Cela correspond au raisonnement sur le temps. Cependant, il convient également de souligner l'existence dans des domaines spécifiques et spécialisés (systèmes temps réel, parallélisme, etc.) d'un raisonnement dans le temps qui s'intéresse au temps nécessaire pour le calcul. Cela ne nous intéresse pas puisque le temps de calcul n'est pas la préoccupation primordiale des systèmes sur lesquels nous nous penchons [16]. Différents problèmes très souvent issus du milieu industriel sont associés au raisonnement temporel : la planification, l'ordonnancement de tâches, le suivi de processus, le diagnostic, la prédiction. Dans notre contexte, nous choisissons de nous focaliser sur l'anticipation. Les bases relatives à l'anticipation ont déjà été introduites dans 2.1.4, dans le paragraphe suivant, nous nous concentrons essentiellement sur la représentation du temps.

La représentation du temps dans les simulations multi-agent Selon Klein [54], il coexiste deux conceptions du temps fondamentalement différentes. La première, linéaire, amène à situer un événement sur l'axe des temps donnés par le calendrier et peut définir par exemple : différentes étapes allant de la naissance jusqu'à la mort. La seconde, cyclique, consiste à situer des phénomènes et processus répétitifs, comme les déplacements quotidiens des habitants d'un territoire.

L'approche de conception du temps linéaire délimite trois positions comme illustrées sur la figure 2.1 :

— un passé, caractérisé par l'histoire, la mesure de ce qui passe et les souvenirs ;
— un présent, défini par l'action ;
— un futur, caractérisé par des prévisions et des anticipations pour approcher une attente incertaine.

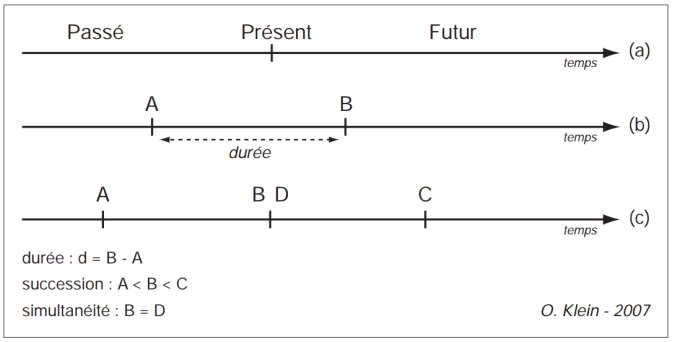


FIGURE 2.1 – Conception linéaire du temps [54]

De cette vision découlent trois caractéristiques du temps : la durée, la succession et la simultanéité. La durée caractérise un espace-temps qui s'écoule par rapport à un phénomène entre deux limites observées (le début et la fin). La succession est un ensemble d'événements ou de phénomènes occupant dans le temps des moments voisins, mais distincts de manière à présenter un ordre. La simultanéité dépeint des événements distincts rapportés à un même moment.

Dans les SMA, la simultanéité est liée à la gestion de l'*environnement* qui est le milieu dans lequel les actions concurrentes se réalisent. La succession et la durée quant à elles sont gérées par l' *ordonnanceur* de la simulation. **Dans notre première contribution qui relève de la représentation du temps, nous distinguons bien ce qui relève de l'environnement de ce qui relève de l'ordonnanceur.** Ces deux aspects sont complémentaires et forment un tout cohérent. Nous y revenons plus en détail dans le chapitre 3.

Par ailleurs, la conception du temps cyclique s'appuie sur le principe de "l'éternel retour". Selon cette conception, le temps présente des apparences de retours périodiques de situations qui peuvent être considérées comme des cycles (Figure 2.2). Un exemple est le temps astronomique qui est marqué par la répétition et l'alternance de saisons, de lever et de coucher du soleil, la rotation de la terre autour du soleil marquant un cycle annuel, etc.

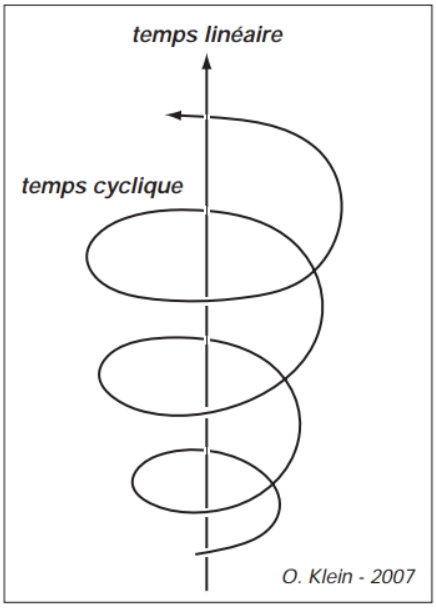


FIGURE 2.2 – Conception cyclique du temps [54]

Cette caractéristique cyclique du temps est généralement traduite par la période. Il s'agit d'un paramètre que nous prenons également en compte dans les détails de notre contribution.

Une autre notion intéressante dans le contexte du temps simulé est la notion d'événements. Selon Klein [54], un événement qualifie tout changement pouvant se produire sur une courte période et perturbant le cours du temps. Dans son approche, l'événement est en lien avec les deux conceptions explicitées précédemment. Il caractérise un fait qui peut survenir à un moment donné, pouvant former une rupture dans le cours des choses de manière relativement soudaine. Un événement peut alors être défini simplement comme une courte période de temps au cours de laquelle apparaît une nouveauté. Il dénote un élément temporel localisé qui rompt le cycle quotidien. L'événement est généralement localisé dans des espaces limités par rapport à l'ensemble considéré. Par conséquent, d'un point de vue géométrique, il peut alors être défini par un point dans l'espace-temps repéré par une localisation associée à un moment et matérialisé par le quadruplet (x, y, z, t).

Ces différentes conceptions du temps coexistent et se matérialisent dans l'espace par la juxtaposition d'échelles temporelles. Ce sont des temps longs et linéaires comme les temps géologiques ou historiques de la ville, des temps courts et cycliques, succession des saisons, temps quotidiens. À ces types de temps se superposent des événements dont les périodes sont plus ou moins longues (jour, semaine ou année). De la même manière, le support de représentation du temps que nous proposons dans notre contribution tient compte de ces différentes conceptions du temps et des propriétés qui en découlent.

Les approches d'ordonnancement du temps dans les simulations multi-agent. Dans un modèle de simulation multi-agent, l'ordonnancement du temps est géré par une entité du simulateur appelé l'*ordonnanceur* (scheduler). À chaque progression du temps simulé, ce dernier déclenche l'ensemble des actions éligibles. Le simulateur fait alors progresser à nouveau le temps simulé et ainsi de suite. Un des intérêts des simulations est que le temps simulé peut progresser beaucoup plus vite que le temps réel. Cela permet d'observer plus rapidement l'évolution du modèle étudié et de déduire des propriétés du système réel imité.

L'ordonnanceur gère notamment deux des trois concepts mentionnés dans la section précédente : la succession et la durée. Pour cela, différentes approches peuvent être utilisées. Les plus communes sont l'approche à pas de temps constant (time-stepped approach), l'approche événementielle (event-driven approach) et l'approche hybride ou mixte (mixed approach). Nous décrivons brièvement ces trois approches classiques. Nous décrivons également une autre approche appelée modèle à temporalité sur laquelle se base notre première contribution. Une étude comparative des différentes approches est faite par la suite dans 2.3.2.2.

1. **L'approche à pas de temps constant** est l'approche la plus couramment utilisée dans

les simulations multi-agent [70]. Dans ce type d'approche, l'ordonnanceur fait avancer le temps simulé en l'incrémentant d'une valeur fixe appelée pas de temps (time-step) [35]. À chaque avancement du temps, l'ordonnanceur déclenche le code de comportement de tous les agents.

Le temps simulé peut alors être représenté graphiquement par un axe discrétisé par des intervalles fixes (Figure 2.3). Cette représentation du temps fait référence à une conception du temps linéaire et/ou cyclique.

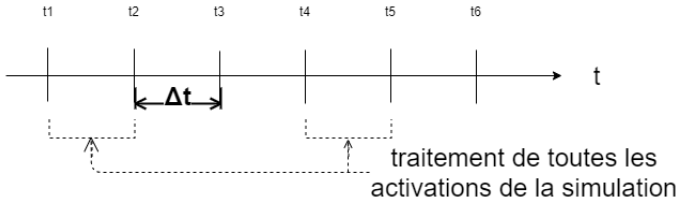


FIGURE 2.3 – Axe de temps simulé : Approche à pas de temps constant [70].

2. **L'approche événementielle** : Dans ce type d'approche, le temps simulé est établi à partir d'informations déclaratives appelées événements. Un événement est la description d'une condition d'activation associée à un ensemble d'agents [70]. La simulation consiste alors en l'exécution d'une liste ordonnée d'événements. L'axe temporel de la simulation est continu, mais l'état du système change de manière discrète à des moments précis décrits par les événements. Cet axe de temps peut être représenté comme une liste chaînée d'événements qui ponctuent l'axe temporel et qui sont inéquitablement espacés (cf Figure 2.4).

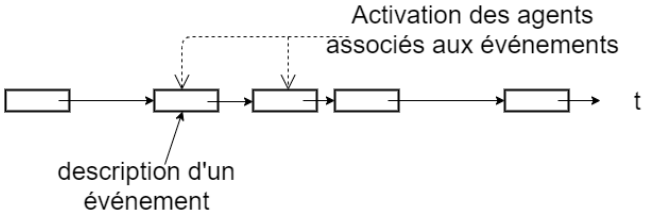


FIGURE 2.4 – Axe de temps simulé approche événementielle.

3. **L'approche hybride** : Ce type d'approche propose de diviser le modèle de simulation en sous-modèles et d'associer, à chaque sous-modèle, le type d'ordonnanceur le plus approprié. C'est l'approche qui est utilisée dans MASON ([58]) et SWARM [62]. Par définition, dans ce type d'approche il n'y a pas d'axe temporel global. Le temps simulé universel résulte de la combinaison des axes temporels de tous les ordonnanceurs utilisés dans la composition.
4. **Le modèle à temporalité** [71] : Dans ce type d'approche, l'agent décrit lui-même sa propre dynamique temporelle. Nous qualifions donc cette approche d'expressive dans le sens où l'ordonnancement de la simulation est basé sur les besoins en termes d'activation temporelle, directement exprimés par les agents qui composent le modèle à simuler. Ces besoins sont exprimés à l'aide d'une structure de données appelée temporalité. Cette temporalité est porteuse de différentes informations relatives à la dynamique d'activation temporelle de l'agent.

   Une temporalité t est définie formellement par :

$$t = \{id, d, f, p, v\} \tag{2.2}$$

   — id : l'identifiant de la temporalité. Ce paramètre permet à l'agent d'identifier le contexte dans lequel il est activé ;
   — $[d, f]$: l'intervalle de temps durant lequel la temporalité est applicable ;
   — p : la période qui définit la série d'occurrences (p = 0 si l'action est ponctuelle) ;
   — v : la variabilité, c'est-à-dire la précision en dessous de laquelle l'occurrence temporelle reste valide.

   Une temporalité provoquera le déclenchement de l'exécution du code du comportement de l'agent à toutes les dates x vérifiant $x = d + p * k$, avec k un entier tel que $0 \leq k \leq n$, et n

le plus grand entier vérifiant $(d + p * n) = f$. On nomme cet ensemble de dates : dates de déclenchement de la temporalité.

Au niveau de l'ordonnanceur, le traitement des différentes temporalités fait intervenir deux structures de données comme illustrées sur la figure 2.5 :

— Slot temporel : élément d'ancrage correspondant à un point de l'axe du temps sur lequel le simulateur va être amené à déclencher l'exécution du code du comportement d'un agent ;

— Tempo : structure regroupant l'ensemble des temporalités qui à un instant donné sont situées sur un même slot temporel et qui ont une même valeur de période. C'est cette valeur de période qui caractérise le tempo en traduisant le fait que les temporalités qu'il comporte ont toutes le même "rythme".

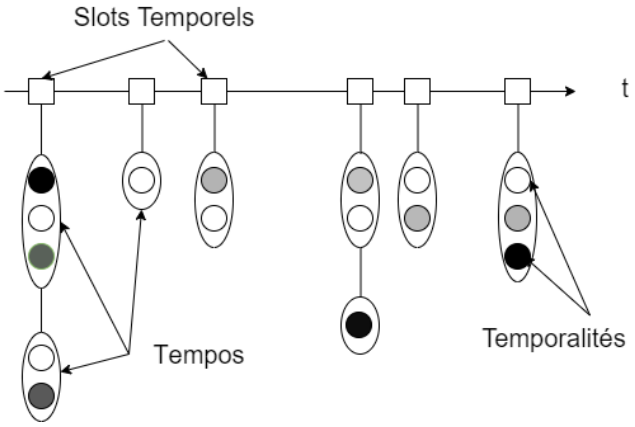


FIGURE 2.5 – Axe de temps simulé modèle à temporalité [70]

Avant de lancer une simulation, l'expérimentateur peut définir les contraintes temporelles qu'il souhaite imposer. Cette possibilité d'avoir la main sur la structuration du temps simulé constitue une grosse différence comparée à l'approche événementielle décrite précédemment. Pour cela, il doit configurer les deux propriétés de régulation suivantes :

— le pas de temps minimum : qui sert à indiquer au simulateur que deux dates de déclenchement distinctes doivent être séparées d'une durée au moins égale à la valeur de ce pas de temps. Si une telle situation se produit lors de l'analyse des temporalités, l'ordonnanceur exploitera le paramètre de variabilité de chacune des temporalités pour déterminer si elles doivent être écartées l'une de l'autre ou bien regroupées sur la même date. Pour chaque insertion, l'ordonnanceur exploite le paramètre de variabilité des temporalités pour optimiser l'organisation des slots temporels et des tempos ;

— la période par défaut : cette valeur sera utilisée pour affecter automatiquement des temporalités aux agents qui n'expriment aucun besoin particulier au cours de la phase d'initialisation des temporalités.

L'ordonnanceur fait progresser le temps simulé en l'emmenant successivement sur chacune des positions signalées par un slot temporel. Sur chaque slot temporel, tous les tempos sont traités en produisant l'exécution de tous les comportements associés aux temporalités qu'ils comportent. Après le traitement d'un tempo, et grâce à la valeur de sa période, on peut calculer le prochain slot temporel sur lequel le rattacher. On obtient ainsi, en une seule opération, la prochaine date de déclenchement de toutes les temporalités portées par le tempo. Une fois que tous les tempos sont traités, on passe au slot temporel suivant et ainsi de suite jusqu'à ce que l'on atteigne la date de fin de la simulation.

Plusieurs autres avantages du modèle à temporalité, comparé aux approches classiques d'ordonnancement du temps, sont cités dans [71]. Dans ce manuscrit, nous en retiendrons deux (2) :

— Cette approche permet un ajustement homogène de l'ordonnancement couvrant tous les cas de figure se situant entre deux extrêmes :
  — Dans les cas où la simulation est composée d'agents "simples" en termes d'activation : Il suffit que ces derniers ne définissent aucune temporalité et le système leur affectera automatiquement une temporalité de base calée sur la période par défaut indiquée

par l'expérimentateur. Ainsi, si tous les agents sont dans ce cas de figure, on retombe automatiquement sur un mode d'ordonnancement "à pas de temps de constant" ;
— Dans le cas contraire où tous les agents sont complexes et expriment chacun un grand nombre de temporalités, on se retrouve avec un ordonnancement de type évènementiel. De ce fait, sans opérer de changement de forme, le modèle à temporalité est adapté aux situations types correspondant à ces deux formes d'ordonnancement.

— Le modèle à temporalité a pour granularité les activités. Dans cette approche les besoins temporels sont exprimés au niveau des activités de l'agent. Il est possible de rattacher une temporalité spécifique à chaque activité distincte. Cela est totalement en phase avec le contexte applicatif de cette thèse qui lui aussi utilise une approche basée sur les activités (activity-based approach).

Cette analyse de la représentation de la dimension temporelle et la comparaison avec celle de la dimension spatiale et de la dimension organisationnelle fait ressortir une grande lacune au niveau de la prise en compte du temps dans les simulations multi-agent. Nous remarquons que même si certaines approches de considération du temps comme le modèle à temporalité permettent à l'agent de s'exprimer (de partager des informations sur la dynamique temporelle), aucun support d'interaction n'est prévu pour l'échange d'informations temporelles. Le mot "échange" ici désigne la nécessité d'un partage d'informations et d'un accès à ces informations partagées. En d'autres termes, un agent partage une information temporelle et le rend accessible par un autre agent. Il peut également accéder à une information partagée par d'autres agents. Malheureusement, dans la plupart des approches d'ordonnancement du temps mentionnées dans la littérature, nous faisons face aux limites suivantes :

— soit le concept même de partage d'information temporelle est inexistant. Ce qui est le cas de l'approche à pas de temps constant où aucun support n'est prévu pour ce type d'échange ;
— soit les agents sont amenés à partager des informations sur leurs dynamiques d'activations temporelles, mais aucun support d'interaction n'est prévu pour l'accès à ces informations. Cela est dû au fait que ces informations sont destinées uniquement à la gestion du cycle d'activation des agents de la simulation au niveau de l'ordonnanceur. C'est le cas des approches dites expressives comme l'approche événementielle ou du modèle à temporalité.

Actuellement, dans la plupart des simulations multi-agent que nous rencontrons dans la littérature, la dimension temporelle sert uniquement dans les mécaniques d'ordonnancement. Ces mécaniques permettent à l'ordonnanceur de la simulation de déclencher les comportements des agents et de gérer le cycle d'activation de la simulation. L'ordonnanceur de la simulation est donc le seul à détenir les informations temporelles. Il est également le seul à y avoir accès. Les informations temporelles sont rarement utilisées au niveau du raisonnement de l'agent, car aucun support n'a été expressément prévu pour les échanges d'informations temporelles. L'agent n'est donc pas acteur de son comportement temporel, il subit le temps et son raisonnement se base uniquement sur le spatial. Ce raisonnement consiste par exemple à calculer l'itinéraire le plus court (spatialement) pour partir d'un lieu à un autre et à choisir sa destination en fonction de cela. Cela vient alors renforcer notre problématique concernant le besoin en support d'interaction. En effet, il est courant de rencontrer des modèles de simulations multi-agent qui intègrent des supports d'interactions sociales et spatiales. Cependant, il n'en existe, à notre connaissance, aucun qui intègre explicitement un support d'interaction temporelle. L'originalité de notre première contribution découle de ce besoin de support d'interaction temporelle. Ce support d'interaction prend la forme d'un environnement. Nous l'appelons l'environnement temporel. Sa structure se base sur l'approche d'ordonnancement de type modèle à temporalité. Nous justifions ce choix par le fait que le modèle à temporalité est une approche dite expressive. Comme expliqué auparavant, cela voudrait dire qu'elle permet aux agents d'exprimer et de partager leur dynamique d'activation temporelle. Le modèle à temporalité permet également la structuration des informations temporelles (tempos, slots, temporalités, etc.) et présente différents autres avantages notamment en termes de passage à l'échelle. En effet, le modèle supporte mieux la montée en charge qui est un objectif difficilement tenable par un modèle événementiel. Il présente une mécanique de gestion qui est beaucoup plus malléable en termes d'ajustement du temps d'exécution, un avantage comparable à celui du modèle à pas de temps constant. Nous abordons cet aspect plus en détail dans [78], [71] et [77].

La mise en place de ce nouveau support d'interaction de nature temporel amène de nouvelles possibilités qui viennent bouleverser le processus de raisonnement de l'agent et plus particulièrement au niveau de son raisonnement anticipatif. Ce raisonnement anticipatif se base la plupart du

temps sur le spatial. Désormais, la mise en place d'un support d'interaction permettant un échange d'informations temporelles permet de compléter le raisonnement spatial par un raisonnement temporel. Notre deuxième contribution consiste alors à l'optimisation du raisonnement temporel au niveau de l'agent, exploitant la visibilité sur la dimension future du temps qui est permise par l'environnement temporel. Ce raisonnement anticipatif optimisé prend en compte les informations temporelles sur le futur planifié, en plus des informations sur le présent et le passé ainsi que des informations sur la dimension spatiale et sociale en tant que dynamique du système.

Les modèles d'interaction entre l'agent et l'environnement Dans cette thèse, le modèle d'interaction influe énormément sur nos choix conceptuels et techniques. Dans cette sous-section, nous nous concentrons sur deux types de modèles d'interaction, le modèle action/réaction qui dans notre cas est utilisé dans SmartCityModel, et le modèle influence/réaction qui est utilisé dans SkuadCityModel.

Modèle Action/Réaction. Le cycle comportemental d'un agent a se résume en trois phases : une phase de perception, une phase de délibération et une phase d'action. Ce cycle est représenté par la fonction $Behaviour_a$. Dans une approche classique de type action/réaction, ce cycle comportemental aboutit à la création d'un nouvel état global de l'environnement. C'est-à-dire que le résultat de l'action d'un agent se concrétise directement par la transformation de l'environnement sous-tendue par cette action [32]. Si nous prenons le formalisme de Genesereth and Nilsson [38], le cycle comportemental d'un agent a est représenté par une fonction

$$Behaviour_a : \Sigma \mapsto \Sigma \tag{2.3}$$

Cette fonction correspond à l'application successive de trois (3) fonctions :
- $Percept_a : \Sigma \mapsto P_a$, qui calcule un percept P_a à partir de l'état du système Σ.
- $Mem_a : P_a \times S_a \mapsto S_a$, qui calcule le nouvel état interne S_a de l'agent a.
- $Decision_a : P_a \times S_a \mapsto \Sigma$, qui modifie le monde suivant l'action de l'agent a.

Cette approche est fortement critiquée du fait qu'elle pose des difficultés à la fois techniques et conceptuelles. Elle ne permet pas de modéliser ou de traiter facilement la simultanéité [31]. Elle viole également la contrainte d'intégrité environnementale et dans certains cas, la propriété d'autonomie des agents du système. En effet, l'agent modifie directement son environnement et dans certains cas, modifie directement les variables d'un autre agent du système[61]. De plus, ce fort couplage entre agent, action et environnement rend le modèle extrêmement sensible à la manière dont il est implémenté. Par exemple, l'ordre dans lequel les agents sont activés influe sur la dynamique du système. Afin de pallier à ces différentes limites, Ferber and Müller [31] proposent le modèle d'interaction appelé modèle Influence/Réaction.

Modèle Influence/Réaction. Dans le modèle Influence/Réaction, le cycle comportemental d'un agent aboutit non pas à la modification directe de l'état global de l'environnement, mais à la production d'influences [31]. Cela veut dire qu'un agent produit des influences sur son environnement et non des actions au sens vu précédemment. Ces influences diffèrent des actions dans le sens où elles ne modifient pas directement l'environnement. Elles représentent le souhait d'un agent de le voir modifié d'une certaine façon, un moyen d'essayer de changer le cours des choses [61]. La distinction claire entre les deux niveaux de dynamique tient dans l'idée que le résultat effectif de cette tentative de modification ne peut être calculé sans connaître l'ensemble des influences produites au même instant. Il s'agit de bien distinguer les gestes produits par les agents, c'est-à-dire les influences (niveau agent), de ce qui se passe effectivement compte tenu des autres gestes, c'est-à-dire la réaction de l'environnement à ces influences (niveau multi-agent) comme le montre la figure 2.6.

La fonction behaviour, représentant le cycle comportemental devient :

$$Behaviour_a : \Gamma \mapsto \Gamma \tag{2.4}$$

Où S_a est l'état interne de l'agent.

Ce modèle introduit la notion d'état dynamique de l'environnement, noté $\delta \in \Delta$. Il s'agit d'une paire notée $< \sigma, \gamma >$ où $\sigma \in \Sigma$ est l'état de l'environnement (les variables environnementales) et $\gamma \in \Gamma$ décrit l'ensemble des influences.

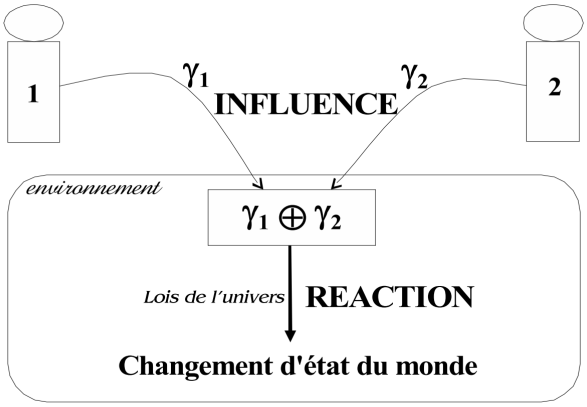


FIGURE 2.6 – Le principe influence/réaction [61]

Ainsi, grâce aux deux notions : influence et réaction, cette approche fait une claire distinction entre la dynamique du niveau agent de la dynamique du niveau multi-agent.

Au niveau agent, la fonction $Behaviour_a$ se décompose en trois (3) fonctions :

— $Percept_a : \Gamma \mapsto P_a$ qui transforme les influences Γ en percepts P_a

— $Mem_a : P_a \times S_a \mapsto S_a$, la fonction de mémorisation qui prend deux arguments : un percept et un état interne S_a de l'agent, et qui renvoie un nouvel état interne

— $Decision_a : S_a \mapsto \Gamma$, la fonction décision qui prend un état interne comme argument et qui indique quelle opération exécuter.

Pour articuler les niveaux agent et multi-agent, la dynamique globale est décomposée en deux fonctions :

— $Exec : \Sigma \times \Gamma \mapsto \Gamma$ qui produit les influences au niveau agent. $\gamma' = Exec(\sigma', \gamma)$

— $React : \Sigma \times \Gamma \mapsto \Sigma$ qui intègre la réaction du monde aux influences au niveau multi-agent. $\sigma' = React(\sigma, \gamma)$

Le modèle influence/réaction présenté par Ferber and Müller est un modèle général d'interaction pour les systèmes multi-agent. Il a l'avantage de bien séparer le niveau agent du niveau multi-agent. Il distingue également tout ce qui relève de l'agent de tout ce qui relève de l'environnement. Ainsi, il permet de représenter plus facilement la simultanéité. Le modèle influence/réaction respecte également la contrainte d'intégrité environnementale et la propriété d'autonomie des autres agents. Cependant, l'approche a été critiquée par Dávila and Tucci [23] du fait que le formalisme défini ne permet pas aux agents de percevoir des variables appartenant à Σ (l'état de l'environnement). En effet, les agents perçoivent ce qui les influence, mais ne sont pas influencés par l'état de l'environnement. Pour y remédier, Ferber et al. [32] proposent une "simplification" et une clarification du modèle dans le cadre de la simulation multi-agent. Ce modèle s'appelle IRM4S.

IRM4S. Comme dans le modèle influence/réaction, le cycle comportemental de l'agent aboutit à la production d'influences.

$$Behaviour_a : \Sigma \times \Gamma \mapsto \Gamma \tag{2.5}$$

Lorsque nous comparons cette fonction $Behaviour_a$ à celle dans le modèle influence/réaction, la première grande différence est la suivante : dans le modèle influence/réaction, Ferber and Müller modélise la perception locale et subjective de l'environnement par une influence $\gamma \in \Gamma$. La fonction $Percept$ se révèle alors très contraignante et entraîne une certaine confusion. Elle suppose notamment qu'un agent ne puisse percevoir des variables appartenant à Σ [24]. Cette perception des variables environnementales a été rajoutée dans IRM4S.

Dans IRM4S, au niveau agent, les trois (3) fonctions composant la fonction behaviour deviennent :

— $Percept_a : \Sigma \times \Gamma \mapsto P_a$. La perception des variables environnementales a été rajoutée. Néanmoins, Ferber et al. conservent la possibilité pour un agent de percevoir les influences. "Cette possibilité est en effet très intéressante, car elle permet de modéliser des perceptions qui intègrent la dynamique d'une situation ; le fait qu'une balle est en train de rouler par

exemple. Une telle perception est difficilement extrapolable à partir de données statiques appartenant à Σ".

— $Mem_a : P_a \times S_a \mapsto S_a$, la fonction de mémorisation prend deux arguments, un percept et un état interne S_a de l'agent, et renvoie un nouvel état interne

— $Decision_a : S_a \mapsto \Gamma$, la fonction décision qui prend un état interne comme argument et qui indique quelle opération exécuter.

Un des intérêts majeurs du principe influence/réaction est de bien distinguer ce qui est propre aux agents de ce qui se passe dans l'environnement. L'environnement possède alors une dynamique endogène et produit lui aussi des influences. L'évolution endogène de l'environnement découle aussi directement de l'état dynamique du système. Il est défini par $Natural_w : \Sigma \times \Gamma \mapsto \Gamma$.

Afin de clarifier l'application du principe influence/réaction, Ferber et al. utilisent une variable temporelle explicite. Les équations temporelles correspondant à ces trois (3) fonctions, à un instant t du temps, se décomposent comme suit :

La fonction $Behaviour_a : \Sigma \times \Gamma \mapsto \Gamma'$ se décompose en 3 fonctions qui s'appliquent séquentiellement :

$$p_a(t) = Perception_a(\sigma(t), \gamma(t)) \tag{2.6}$$

$$s_a(t + dt) = Memorization_a(p_a(t), s_a(t)) \tag{2.7}$$

$$\gamma'(t) = Decision(s_a(t + dt)) \tag{2.8}$$

$Natural_w$ qui correspond aux influences produites par l'environnement s'écrit alors comme suit :

$$\gamma'_w(t) = Natural_w(\sigma(t), \gamma(t)) \tag{2.9}$$

$Influence$ donne donc un ensemble $\gamma'(s)$ qui contient les influences déjà présentes dans le système et les influences de l'environnement et des agents :

$$\gamma'(t) = Influence(\sigma(t), \gamma(t)) = \gamma(t), \gamma'_w(t), \bigcup_a \gamma'_a(t) \tag{2.10}$$

au niveau multi-agent, ils définissent alors une fonction $Evolution$ telle que

$$\sigma(t + dt) =< \delta(t + dt), \gamma(t + dt) >= Evolution(< \sigma(t), \gamma(t) >) \tag{2.11}$$

Cette fonction se décompose en deux (2) nouvelles fonctions qui vont clarifier l'application du principe d'un point de vue temporel :

— $Influence : \Sigma \times \gamma \mapsto \gamma'$ définit globalement les influences produites au niveau micro par les agents et l'environnement, puis

— $Reaction : \Sigma \times \gamma' \mapsto \Sigma$, γ définit la manière dont le monde se transforme pour donner un nouvel état dynamique.

Ainsi, l'état dynamique de l'environnement $\delta(t)$ évolue en $\delta(t + dt)$ en appliquant les équations suivantes :

$$\gamma'(t) = Influence(\sigma(t), \gamma(t)) \tag{2.12}$$

$$< \sigma(t + dt), \gamma(t + dt) >= Reaction(\sigma(t), \gamma'(t)) \tag{2.13}$$

Une différence fondamentale par rapport au modèle influence/réaction c'est que ces deux fonctions sont instantanées du point de vue temporel. Cela veut dire que l'état du système est indéfini entre deux dynamiques distinctes.

La Figure 2.7 montre une représentation schématique du principe d'évolution d'un système modélisé avec le modèle IRM4S.

Pour tous les avantages évoqués précédemment, nous préconisons l'utilisation du modèle d'interaction de type influence/réaction, plus particulièrement, le modèle IRM4S. Nous y revenons plus en détail dans le chapitre 3.

Nous avons abordé dans 2.2.2.2 les concepts de bases relatives à la structuration de l'environnement temporel. Nous avons, par la suite, abordé dans 2.2.2.2, différents modèles d'interaction faisant le lien entre l'agent et l'environnement. En complément à cela et pour articuler et de structurer le système, nous nous penchons vers les modèles d'organisations. En effet, ces modèles décrivent une première structure, de nature organisationnelle, au niveau du système. Certaines de

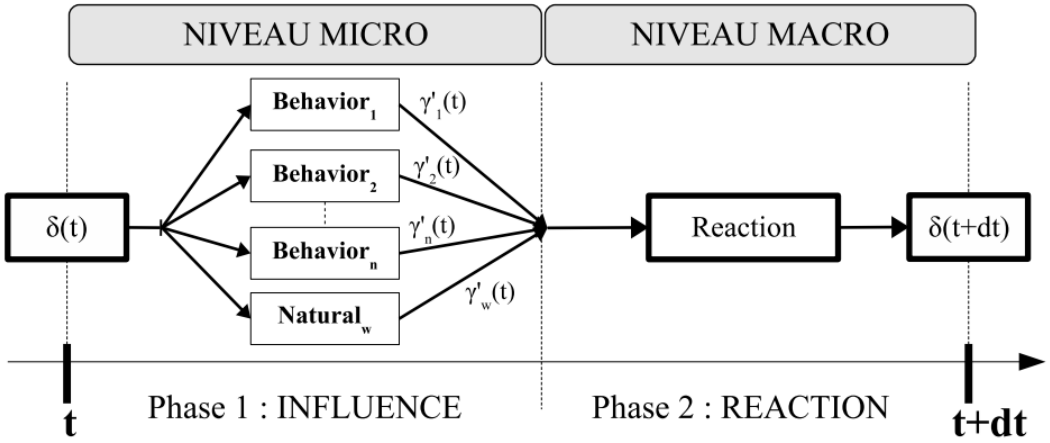


FIGURE 2.7 – Principe d'évolution d'un système modélisé avec le modèle IRM4S [61]

ces structures sont intéressantes dans le sens où elles intègrent déjà la notion d'environnement et disposent de méta-modèle UML bien défini et reconnu au niveau de la communauté des chercheurs dans le domaine des simulations multi-agent. C'est le cas de AGR (et de sa variante AGRE), qui est l'approche sur laquelle nous nous reposons. Notre objectif est de mettre en place un modèle qui permet de prendre en compte, au même titre, l'environnement spatial, l'environnement social et l'environnement temporel. Afin d'avoir un tout cohérent, le modèle mis en place doit également pouvoir fonctionner avec un modèle d'interaction de type influence/réaction. Tous ces besoins et critères influent énormément sur nos choix d'approches.

Les modèles d'organisation Il existe plusieurs modèles d'organisations dans la littérature. Certains d'entre ces modèles prennent en compte les aspects normatifs, c'est-à-dire la description des mécanismes d'interactions entre les différents agents dans l'organisation. Des exemples sont Model of Organization for multI-agent SystEms (MOISE) [44], ISLANDER [27] et Organizations per Agents (OperA) [51]. D'autres décrivent uniquement la structure organisationnelle c'est-à-dire à quelle organisation l'agent appartient, quels sont ces objectifs et ses fonctions. Des exemples sont AGR, Rôles Interactions Organisations (RIO) [60] et Task Analysis Environment Modeling and Simulation (TEAMS) [90]. Nous en décrivons un de chaque type : MOISE et AGR.

MOISE [44] MOISE et ses extensions : MOISE$^+$ [48] et MOISE^{Inst} [37] sont des modèles de description de SMA orienté organisations. Ils prennent en compte aussi bien les aspects structurels que les aspects fonctionnels. Ces modèles distinguent trois niveaux de spécification :

1. **Spécification structurelle** : le modèle s'articule autour des notions de groupes, de rôles et de relations. Les groupes se composent de sous-groupes et de rôles et définissent les lieux où l'interaction entre agents est possible. Les rôles représentent, pour les agents, les comportements possibles ou autorisés. L'ensemble des rôles se définit selon une hiérarchie, avec une relation d'héritage. Les relations représentent les liens entre agents jouant des rôles spécifiques. Les relations sont typées. Ces types peuvent être : connaissance, communication ou autorité. MOISE est un modèle multi-niveau. Chaque organisation est spécifiable au niveau :
   — individuel : comportements d'agents autorisés pour chaque rôle ;
   — social : interactions possibles entre rôles ;
   — collectif : structures résultant de ces rôles interconnectés.
   Les liens suivants peuvent être établis entre deux rôles :
   — lien d'accointance : pour structurer la représentation des autres ;
   — lien de communication : pour structurer les échanges d'informations ;
   — lien d'autorité : pour définir une structure de contrôle ;
   — lien de compatibilité : pour définir les rôles qui peuvent être joués par un même agent.
2. **Spécification fonctionnelle** : permet d'exprimer comment des buts collectifs sont décomposés par des plans et répartis sur les agents par des missions.

3. **Spécification normative ou déontique** : permet de relier la spécification structurelle et la spécification fonctionnelle. Les normes décrivent pour les agents, en fonction de leurs rôles, quelles sont les missions pour lesquelles ils peuvent s'engager.

Il s'agit d'un modèle permettant de spécifier de manière très complète les organisations, indépendamment des agents qui les instancient : les agents entrant dans le système peuvent choisir de jouer différents rôles, mais sont dès lors contraints par les normes de comportement et d'interaction liées à ce rôle.

Agent-Groupe-Rôle (AGR) [30] Agent-Groupe-Rôle (AGR) est un modèle générique d'organisations multi-agent aussi connu sous le nom du modèle Aalaadin [30]. Il s'agit d'un des modèles génériques d'organisations multi-agent les plus connus. Son organisation s'articule autour de trois notions comme le montre la figure 2.8 : agent, groupe, rôle.

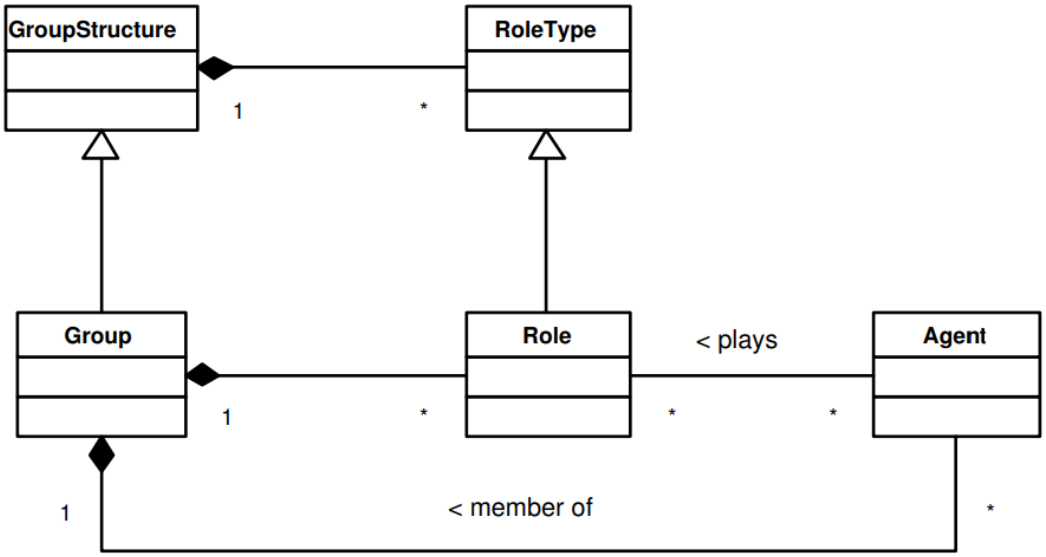


FIGURE 2.8 – Un méta-modèle UML de l'approche Agent, Groupe, Rôle (AGR) [32]

Definition 2.2.1. Agent. Un agent est une entité active et communiquante. Il joue un *rôle* dans un ou plusieurs groupes, peut tenir plusieurs rôles et peut être membre de plusieurs groupes.

Definition 2.2.2. Groupe. Un groupe est un ensemble d'agents partageant des caractéristiques communes. Un groupe est utilisé comme un contexte pour un modèle d'activités ou pour partitionner des organisations. Deux agents peuvent communiquer si et seulement s'ils appartiennent au même groupe. Un agent peut appartenir à plusieurs groupes. Cette fonctionnalité permet de définir des structures organisationnelles.

Definition 2.2.3. Rôle. Le rôle est la représentation abstraite d'une position fonctionnelle d'un agent dans un groupe. Un agent doit jouer un rôle dans un groupe et peut en jouer plusieurs dans ce même groupe. Les rôles sont locaux aux groupes et un rôle doit être demandé par un agent. Un même rôle peut être joué par plusieurs agents.

AGR a été critiquée par ces mêmes auteurs ([32]) du fait qu'elle ne prenne pas en compte le concept d'environnement. Pourtant, il s'agit d'un concept important dans le cadre des interactions dans les SMA et plus particulièrement dans le contexte d'agents situés.

En effet, selon Ferber et al. [32], les SMA centrés environnement, contrairement aux SMA standards sont construits selon les principes suivants [32] :

Principe 2.2.1. Le niveau organisationnel dans les SMA décrit le "quoi" et non le "comment". Cela voudrait dire que ce niveau impose une structure dans le modèle des activités des agents, mais ne décrit pas le comportement de ceux-ci.

Principe 2.2.2. Il n'existe aucune description d'agent et par conséquent aucun problème mental au niveau organisationnel. Le niveau organisationnel ne doit rien dire sur la façon dont les agents interprètent ce niveau.

Principe 2.2.3. Une organisation fournit un moyen de partitionner un SMA, chaque partition constitue un contexte d'interaction pour les agents.

Ainsi, si l'on considère un environnement comme étant un ensemble de conditions selon lesquelles une entité existe, les principes d'AGR ne fournissent qu'un support pour l'environnement social et non pour les environnements physiques. De plus, elles ne considèrent pas les entités autres que les agents.

Plusieurs extensions d'AGR ont été proposées afin d'intégrer les environnements aux groupes. Un exemple est le modèle proposé par Odell et al. [67] (cf figure 2.9) qui associe directement l'environnement aux groupes.

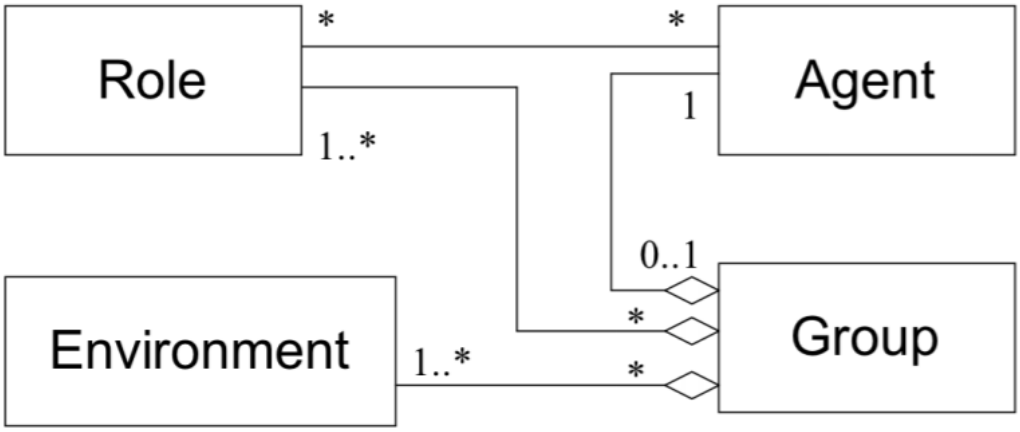


FIGURE 2.9 – Une extension d'AGR avec l'environnement proposé par Odell et al. [67]

Cependant, ce modèle ne résout pas réellement toutes les questions relatives à l'intégration des environnements aux organisations. Plusieurs problèmes concernant la relation qui existe entre l'agent et l'environnement restent non résolus [33]. Par exemple, le concept de corps (partie de l'agent qui exécute une action dans un environnement) n'est pas analysé proprement. Ferber et al. décident alors d'étendre AGR afin de rajouter l'environnement. Ils proposent donc AGRE dont le principe consiste à intégrer l'environnement aux organisations multi-agent. Un agent doit être pensé comme ayant deux parties : un esprit (ou cerveau) et un corps. Le corps peut être soit un "corps physique" dans un environnement physique, soit un "corps social" (un rôle) dans une organisation. Nous détaillons le fonctionnement de AGRE dans la partie 2.2.2.2.

L'implémentation de notre première contribution au niveau de notre plateforme de simulation SimSKUAD se fait sous la forme d'un environnement physique. Nous détaillons cela dans le chapitre 5.

Agent-Groupe-Rôle-Environnement (AGRE) AGRE [32] vient compléter l'aspect situé corporel à l'aspect situé social d'AGR. Dans AGRE, un agent possède un ensemble de *modes* qui peuvent être sociaux ou physiques. Les modes sociaux, sont considérés comme des interfaces sociaux permettant d'agir en groupe et sont appelés *rôles*. De la même manière, les modes physiques sont considérés comme des interfaces physiques permettant d'agir dans une *zone*. D'une manière plus générale, les modes sont des interfaces permettant d'agir dans les *espaces*.

Ainsi, AGRE permet de prendre en compte deux des trois dimensions citées par Rolland-May [83] : la dimension spatiale et la dimension organisationnelle. La figure 2.10 montre un méta-modèle UML de l'approche AGRE.

AGRE défini, en plus du concept d'agent, trois concepts généraux qui se déclinent en concepts spécifiques selon le type d'environnement (organisationnel ou spatial) :

Definition 2.2.4. Espace (space). Les espaces sont des domaines dans lesquels les agents sont situés. Dans AGRE, les espaces peuvent être physiques (géométriques) ou sociaux. Les espaces géométriques sont appelés zones (areas) et les espaces sociaux sont appelés groupes (groups).

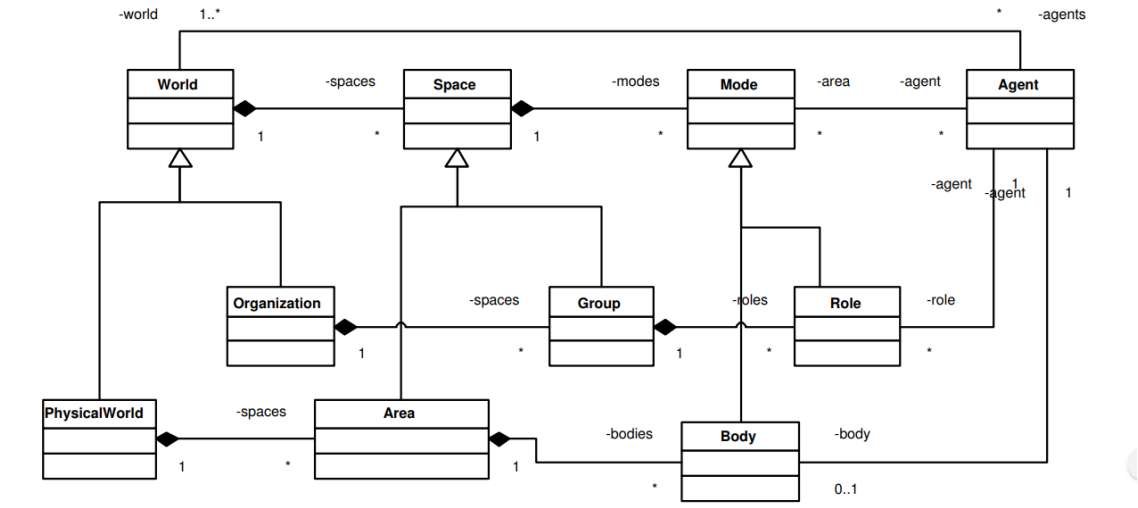


FIGURE 2.10 – Un méta-modèle UML de l'approche Agent, Groupe, Rôle, Environnement (AGRE) [32]

Definition 2.2.5. Mode (mode). Un mode est la manifestation d'un agent dans un domaine spécifique, son mode d'existence et son apparence dans un espace. Un mode décrit la position de l'agent et la manière dont il perçoit et agit dans un space. Les agents sont alors situés dans les spaces et peuvent y agir à travers les modes. Un mode dans une zone est appelé corps (body) tandis qu'un mode dans un groupe est appelé rôle (role).

Definition 2.2.6. Monde (worlds). Un monde est une collection de spaces de même type. AGRE définit deux types de mondes : les organisations qui représentent l'environnement social et qui sont composées de groupes et le monde physique qui représente l'environnement physique et qui est composé de zones. Néanmoins, Ferber et al. [32] évoquent la possibilité de considérer d'autres mondes comme des mondes pour l'affichage des agents de manière spécifique, des mondes constitués de places pour décrire la mobilité de l'agent, etc.

Dans cette thèse, nous choisissons de nous baser sur AGR et sur son extension AGRE pour les raisons évoquées précédemment et pour les suivantes :

- AGR est une des approches organisationnelles comportementalistes les plus connues des SMA. Elle a l'avantage d'être minimaliste et générique.
- AGR permet de décrire les organisations indépendamment de toute implémentation ou architecture interne des agents. Elle permet également d'intégrer facilement le modèle influence/réaction.
- Les notions organisationnelles utilisées prennent leur origine dans les organisations humaines comme celles que nous prenons en compte dans notre contexte applicatif.
- Nous nous intéressons particulièrement aux aspects structurels. Dans ce contexte, AGR et ses extensions répondent parfaitement à nos besoins. Nous pensons tout de même que d'autres modèles prenant en compte les aspects normatifs peuvent être tout aussi intéressants. Ainsi, même si nous nous basons essentiellement sur AGR, nous pensons que la possibilité d'intégrer l'ensemble de nos solutions au niveau d'autres modèles comme MOISE et ses extensions constituent une piste de recherche intéressante à explorer.

AGRE permet de prendre en compte deux des trois dimensions citées par Rolland-May [83] : la dimension organisationnelle et la dimension spatiale. Afin de prendre en compte la troisième dimension qui est la dimension temporelle, nous proposons d'enrichir le modèle AGRE en y ajoutant la dimension temps. L'objectif est de faire évoluer le paradigme agent de manière à considérer le temps comme un nouveau milieu d'interaction, au même titre que l'environnement spatial et l'environnement social. Nous appelons ce milieu **environnement temporel** et l'extension du modèle AGRE : **AGRET (AGRE+T) pour Agent-Groupe-Rôle-Environnement-Temps**. AGRET constitue notre première contribution, nous décrivons ce modèle dans la section 3.1 du chapitre 3 suivant.

Ferber et al. [33] critiquent AGRE et toutes les familles de modèles de type AGR (dont AGRET) pour le fait qu'ils ne fournissent pas de théorie de l'action qui prendrait en compte les actions

concurrentes. Afin d'y remédier, une des solutions proposées consiste en l'utilisation du modèle d'interaction de type influence-réaction [31] [33] pour l'interaction entre l'agent et l'environnement. Nous avons décrit ce modèle dans 2.2.2.2.

2.3 Positionnement et contributions

2.3.1 Synthèse

Dans ce chapitre, nous avons dressé un état de l'art du contexte général de la thèse et de son cadre applicatif. Cet état de l'art présente les bases théoriques relatives à l'ensemble des problématiques que nous avons évoquées dans 1.2. Il présente les notions de base générales et les concepts plus spécifiques sur lesquels s'appuient nos propositions.

Nous avons commencé par introduire le contexte applicatif de la thèse : la ville intelligente et ses caractéristiques. Ces caractéristiques montrent que les SMA figurent parmi les solutions intéressantes pour leur mise en œuvre et leur modélisation. Plus particulièrement, plusieurs travaux ont déjà été menés et ont fait leurs preuves, dans la modélisation des flux de déplacements de personnes au quotidien sur un territoire. Nous y avons par la suite introduit brièvement les deux modèles de simulations SmartCityModel et SkuadCityModel sur lesquels nous avons débuté nos expérimentations et desquels découle le choix des problématiques principales de la thèse. Les implémentations de nos approches, décrites dans le chapitre 5 se feront notamment sur l'un d'entre eux : SkuadCityModel, bien qu'une perspective d'implémentation est également envisagée sur SmartCityModel. Les premières études nous ont permis de relever ne limite importante que nous rencontrons souvent dans les simulations multi-agent : la faible prise en compte de la dimension temps comparée à la dimension spatiale et la dimension organisationnelle. Nous avons donc, dans la section 2.2.1, effectué une étude comparative entre le temps et l'espace. Notre objectif est de montrer l'intérêt de prendre en considération le temps au même titre que l'espace dans les simulations multi-agent. Plus particulièrement, nous nous focalisons sur le contexte applicatif de la ville intelligente où les trois dimensions : spatiale, organisationnelle et temporelle sont considérées comme indispensables. Une analyse plus poussée de la considération de ces trois dimensions dans les simulations multi-agent est faite dans la section 2.2.

De cette analyse, nous concluons que dans la plupart des simulations multi-agent, l'agent est acteur de son comportement spatial et organisationnel et non de son comportement temporel. En effet, une certaine maîtrise du comportement spatial et organisationnel est permise par l'existence de supports d'interactions spatiaux et organisationnels. Dans le cas de l'espace, le milieu d'interaction est l'environnement physique. Dans le cas de l'organisation, il s'agit de l'environnement social. Les interactions entre l'agent et ces différents environnements se font par le biais d'un modèle d'interaction. Nous en avons présenté deux : action/réaction et influence/réaction, en faisant ressortir les avantages et inconvénients de chaque approche. Ces avantages et inconvénients justifient notre choix de nous pencher sur un modèle de type influence/réaction que ce soit au niveau conceptuel ou au niveau technique. Selon les modèles d'interaction, non seulement les agents agissent ou exercent une influence sur leur environnement, mais ils y perçoivent également des informations.

Par ailleurs, les caractéristiques de la ville intelligente nous a également permis de justifier notre intérêt pour le raisonnement anticipatif dans les simulations multi-agent que nous avons présenté dans 2.1.4. Plus précisément, nous nous intéressons à la considération ou la prise en compte de la dimension future du temps dans le raisonnement anticipatif. Nous avons notamment remarqué que la prise en compte des trois dimensions de l'espace (abscisse, ordonnée et hauteur) se fait de manière très intuitive dans le raisonnement anticipatif. Contrairement à cela, la plupart des mécanismes d'anticipation dans les simulations multi-agent a tendance à négliger l'existence et l'importance de la troisième position de la dimension temps qui est le futur planifié. En d'autres termes, la plupart des raisonnements anticipatifs existants ne prennent pas en compte, ou du moins pas explicitement, les informations temporelles positionnées sur le futur planifié par les agents. Seules les informations sur le passé et sur le présent sont considérées. Cela va à l'encontre de ce qui se fait dans la réalité.

D'une manière plus générale, dans beaucoup de simulations multi-agent, les informations perçues concernent les contextes d'activations spatiales et organisationnelles des agents et sont prises en compte dans leur processus décisionnel. Selon Nigon et al. [64], les différentes caractéristiques des villes intelligentes partagent un trait commun : elles dépendent toutes du contexte. L'accès ou la perception de données contextuelles et l'extraction d'informations pertinentes à partir de celles-ci sont toujours la clé pour faire progresser l'efficacité des fonctionnalités des villes intelligentes.

Dans les villes intelligentes, ces informations concernent la dimension spatiale, la dimension sociale et la dimension temporelle. Cependant, comme mentionné précédemment, contrairement à la dimension spatiale ou organisationnelle, l'agent n'est pas acteur de son comportement temporel. Cela veut dire qu'il n'a aucune emprise sur le temps simulé : il le subit. En effet, dans les simulations multi-agent, le temps est souvent une mécanique du système. Son fonctionnement consiste en une horloge qui gère l'activation des agents et de la simulation. Cette considération assez pauvre du temps ne fournit pas de support permettant aux agents d'exploiter des informations temporelles dans leur processus de décision. Dans certains cas comme dans le modèle à temporalité, ces informations sont présentes dans la représentation du temps, mais ne sont pas exploitées dans le raisonnement temporel. En effet, les agents n'y ont pas accès, car il n'existe aucun support qui le permet. Notre première contribution que nous présentons dans le chapitre 3 relève donc de la représentation du temps au même titre que les dimensions spatiale et organisationnelle. Ce support est un milieu d'interaction que nous appelons environnement temporel. Le modèle d'interaction utilisé est de type influence-réactions. Pour articuler l'ensemble au niveau du système multi-agent, nous mettons en place un modèle appelé AGRET. Ce modèle se base notamment sur un modèle organisationnel très connu des SMA : AGR et sur sa variante AGRE. Cette représentation du temps sous forme d'environnement permet non seulement le partage des informations sur la dynamique temporelle des agents, ce qui était déjà le cas avec certaines approches d'ordonnancement comme l'approche événementielle ou le modèle à temporalité, elle permet également l'accès aux informations temporelles sur le passé, le présent et le futur planifié.

En complément à cela, nous proposons une approche d'anticipation qui se base sur notre première contribution relative à la représentation du temps sous forme d'environnement afin de fournir aux agents une visibilité sur la dimension future du temps. Cela permettra aux agents d'intégrer dans leur raisonnement anticipatif la prise en compte des informations positionnées sur le futur.

Pour résumer, nous nous intéressons à deux aspects du temps simulé :

1. La représentation du temps : nous avons notamment noté une considération pauvre du temps comparée à une prise en compte très poussée de l'espace. Notre proposition consiste à faire évoluer les simulations multi-agent de manière à considérer le temps comme un milieu d'interaction au même titre que l'espace et les organisations. Ce milieu d'interaction temporel vient en complément à l'ordonnanceur de la simulation ;
2. Le raisonnement temporel : nous constatons, dans la plupart des simulations, au niveau de l'agent, une capacité d'anticipation très limitée, éloignée de la réalité et ne prenant pas en compte les informations temporelles positionnées sur le futur. Nous proposons de faire évoluer les approches d'anticipation classique de manière à prendre en compte les informations temporelles sur le futur planifié.

Cette analyse bibliographique nous permet également de dresser une liste de quatre critères que l'ensemble des solutions que nous proposons dans les chapitres 3 et 4 devraient remplir :

1. intégrer une prise en compte de la dimension temporelle au même titre que les dimensions spatiales et organisationnelles ;
2. permettre aux agents de s'exprimer (partager des informations) sur la dimension temporelle comme sur la dimension spatiale ou la dimension organisationnelle ;
3. intégrer un support permettant aux agents de partager, de stocker et de percevoir des informations temporelles comme c'est le cas pour la dimension spatiale et la dimension organisationnelle. Ce support est un milieu d'interaction, plus précisément, un environnement. Le modèle d'interaction utilisé est de type influence/réaction ;
4. introduire dans le modèle simulé une visibilité sur la dimension future du temps sur laquelle pourra s'appuyer un certain potentiel d'anticipation comme c'est le cas dans la vie réelle.

2.3.2 Choix conceptuels et techniques

Nos choix conceptuels et techniques se font sur la base des critères cités précédemment. Nous introduisons brièvement ces choix dans cette sous-section. Des descriptions plus détaillées et plus précises seront données en début des chapitres 3, 4 et 5.

2.3.2.1 Un support d'interaction : Modélisation du temps sous forme d'environnement

L'environnement, qu'il soit physique ou social, occupe le temps et l'espace [68]. De même, son action est située à la fois dans l'espace et dans le temps [94] et y est contrainte. Il est donc possible d'associer à l'action, une position dans le temps et dans l'espace. Dans les simulations agents, la position de l'agent dans l'espace est associée à l'environnement physique tandis que sa position dans le temps est donnée par une horloge virtuelle gérée au niveau de l'ordonnanceur de la simulation. Vus sous cet angle, espace et temps sont considérés de la même manière, car la notion de position "existe" au niveau des deux dimensions. Cependant, par définition, un agent est autonome et acteur de son comportement. Ce comportement peut être spatial et/ou temporel. Au niveau spatial, l'environnement spatial offre aux agents une visibilité sur les informations spatiales et donc la possibilité d'agir/influencer et de percevoir l'espace selon différentes règles via ses capteurs. Contrairement à cela, le temps est géré au niveau de l'ordonnanceur. Son fonctionnement classique consiste en une mécanique qui fait écouler le temps selon des intervalles définis en fonction de l'approche d'ordonnancement utilisée. Par conséquent, aucune visibilité sur les informations temporelles ne s'offre à l'agent. Il s'agit d'une visibilité qui pourtant est existante dans la réalité. Par exemple : dans la réalité, les agents peuvent déjà avoir des informations sur ce qu'ils souhaitent faire dans le futur. De manière générale, une personne qui travaille est censée arriver à son bureau à une heure précise et à une fréquence régulière. Ces informations sont prises en compte dans leur raisonnement et dans leur choix de comportement. Dans les simulations agents, la mécanique actuelle de gestion du temps ne donne pas aux agents la possibilité d'avoir une visibilité sur ces informations qui sont portées par la dimension temps. Il existe certains travaux comme ceux de Zargayouna [99] qui traitent de la dimension spatio-temporelle et présentent des solutions alternatives pour la prise en compte de la variable temporelle dans le système. Il y a également ceux de Chaouche et al. [18] qui traitent de la planification et du guidage spatio-temporel. Cependant, d'une manière générale, la dimension temporelle dans les simulations multi-agent reste encore très peu exploitée. De plus, l'agent n'a aucune emprise vis-à-vis du temps, il le subit. Il n'est pas acteur de son comportement temporel et aucun raisonnement n'est fait ou ne peut être fait par rapport aux informations temporelles. L'objectif principal de notre première contribution est alors de faire évoluer le paradigme des simulations multi-agent afin de considérer le temps comme un nouveau milieu d'interaction, au même titre que l'espace et les organisations. Nous complétons donc les mécanismes de gestion du temps, au niveau de l'ordonnanceur de la simulation, par une représentation du temps sous forme d'environnement. Cette approche de représentation du temps doit alors pouvoir fournir un support permettant aux agents de partager et de percevoir des informations sur la dimension temps comme ce qui se fait déjà pour la dimension spatiale et la dimension organisationnelle.

Nous tenons à bien souligner que notre objectif n'est pas de proposer un modèle de représentation de connaissances temporelles, mais un support permettant aux agents de partager et de percevoir des informations temporelles. L'environnement temporel, comme nous le concevons, bien qu'il soit indépendant du fonctionnement de l'ordonnanceur de la simulation, vient compléter le rôle de celui-ci. Ainsi, l'ordonnancement du temps reste géré au niveau de l'ordonnanceur de la simulation tandis que le traitement et le stockage des informations temporelles sont gérés au niveau de l'environnement temporel.

2.3.2.2 Un support d'interaction temporelle structurée sur la base de l'approche de type modèle à temporalité

Dans les simulations multi-agent, il existe plusieurs approches d'ordonnancement du temps. La plus classique est l'approche à pas de temps constant. Elle a l'avantage d'être généralement facile à mettre en œuvre. De plus, comme tous les agents sont régis par la même fréquence d'activation, elle convient parfaitement à des modèles d'agents à comportements homogènes. Ce sont des modèles constitués d'une unique famille d'agents ayant les mêmes besoins en termes de fréquence d'activation. Cependant, ce type d'approche est souvent critiqué pour ses limites dans le cas de modèles composés d'agents à comportements hétérogènes, comme c'est souvent le cas dans les villes intelligentes. En effet, utiliser une valeur de pas de temps inapproprié peut avoir soit un effet de léthargie soit un effet d'hyperactivité des agents. Si l'agent est activé trop peu souvent, son action est ralentie par rapport à son fonctionnement normal. Contrairement à cela, si l'agent est activé trop fréquemment, ses actions vont être accélérées. Afin d'éviter ces incohérences, une technique consiste à définir une valeur de pas de temps égal au plus petit intervalle de temps exigé

dans l'ensemble des agents. Cependant, il n'est pas envisageable d'effectuer la démarche inverse. Si un agent peut ralentir volontairement son activité pour contrer son hyperactivité, il n'a pas la possibilité d'agir sur un intervalle temporel plus petit que celui imposé par l'ordonnanceur pour corriger un état léthargique. Il suffit également qu'un petit nombre d'agents ait besoin d'un pas de temps faible pour que la majorité des agents passent la plupart de leur temps à s'inactiver [70]. De plus, les modèles utilisant cette approche passent difficilement à l'échelle. En effet, il faut, à chaque pas de temps, activer tous les agents du système, même ceux qui n'ont en pas besoin. Cela entraine un gaspillage de ressources computationnelle et va donc à l'encontre des principes de la ville intelligente qui se veut être écologique. Fabien Michel [61] conclut donc qu'une discrétisation régulière du temps est inappropriée lorsque le modèle simulé nécessite de prendre en compte des actions d'agents fortement hétérogènes (du point de vue de la fréquence d'activation).

Par ailleurs, nous constatons également que l'approche à pas de temps constant présente aussi des limites en termes d'expressivité. En effet, elle ne permet pas à l'agent d'avoir une emprise sur le temps simulé. L'agent ne peut pas exprimer lui-même ses besoins en termes d'activation temporelle. Au contraire, le système lui impose une rythmique d'activation qui peut lui être ou ne pas lui être adaptée. De plus, cette approche n'offre aucun support permettant de partager ou de percevoir des informations sur la dynamique d'activation temporelle des agents.

L'approche événementielle permet de s'affranchir de certaines limites de l'approche de type à pas de temps constant. Elle convient notamment aux modèles composés d'agents hétérogènes et permet aux agents d'exprimer eux-mêmes leur besoin en termes d'activation temporelle (approche expressive). Cependant, cette approche est critiquée par Payet et al. [70] du fait qu'elle ne permet pas à l'utilisateur de la plateforme d'avoir une emprise sur le temps simulé. Aussi, ce dernier peut prendre des formes complexes et alourdir les calculs effectués par le simulateur. Galler [36] estime même qu'il est souvent préférable de choisir les approches à pas de temps constant pour leur simplicité.

Dans notre contexte, un des grands avantages de cette approche est le fait qu'elle permet aux agents d'exprimer leur dynamique d'activation temporelle. Cependant, malgré son expressivité, ce type d'approche, ne prévoit aucune structure permettant aux agents de percevoir les informations qu'ils expriment sur leur dynamique d'activation temporelle.

Concernant l'approche hybride, le manque de vision globale du temps pose des problèmes si l'on envisage de réaliser une analyse de la dimension temporelle de la simulation[70].

Le modèle à temporalité cumule les grands principes avantageux que l'on trouve dans les approches précédentes : la périodicité et la réductibilité du temps d'exécution des approches à pas de temps constant, la précision et la fugacité des approches événementielles et l'adaptabilité aux modèles complexes des approches composées. Elle présente également des avantages en termes de passage à l'échelle [70] [79]. De plus, sa structuration (temporalité, slot temporels, tempo) offre un support permettant aux agents de partager des informations sur leur dynamique d'activation temporelle. Cependant, la possibilité de perception de ces informations reste encore un champ à explorer. Par ailleurs, le modèle à temporalité a également l'avantage d'intégrer dans son fonctionnement les principes du modèle d'interaction influence/réaction dont les principaux avantages sont cités dans la section 2.2. C'est pour cela que dans le cadre de cette thèse, nous reprenons les principes de bases du modèle à temporalité dans la structuration de notre environnement temporel. Nous préconisons également une utilisation de l'approche de type modèle à temporalité au niveau de l'ordonnancement de la simulation. Néanmoins, comme précisé dans 2.3.2.1, l'environnement temporel et l'ordonnanceur de la simulation, bien qu'ils soient complémentaires fonctionnent de manière indépendante. En fonction des besoins, il est donc tout à fait possible de faire tourner une simulation utilisant un ordonnanceur fonctionnant avec une approche de type modèle à temporalité sans mettre en place un environnement temporel (comme c'est déjà le cas actuellement dans SkuadCityModel). Il est également envisageable de remplacer le type d'ordonnanceur utilisé et de le coupler avec un environnement temporel. Une description plus détaillée de l'approche est faite dans le prochain chapitre (Chapitre 3) et un exemple d'implémentation est présenté dans le chapitre 5.

Par ailleurs, selon Rolland-May [83], l'étude d'un système territorial doit intégrer simultanément trois dimensions :

— la dimension temporelle,
— la dimension spatiale,
— la dimension organisationnelle.

Dans les simulations multi-agent, la modélisation sous forme d'environnement est une des ap-

proches les plus communément utilisées pour représenter et exploiter les dimensions spatiales et organisationnelles [32]. Comme précisé précédemment, notre contribution consiste à faire évoluer le paradigme agent, de manière à considérer le temps comme un nouveau milieu d'interaction, au même titre que l'environnement spatial et l'environnement organisationnel. Nous appelons ce nouveau milieu d'interaction "environnement temporel". Pour structurer l'ensemble, nous nous basons sur des modèles existants, communément utilisés dans les SMA : le modèle générique d'organisation AGR (Agent-Groupe-Rôle) et son extension AGRE (Agent-Groupe-Rôle-Environnement). Nous en proposons notre propre extension que nous appelons AGRET (Agent-Groupe-Rôle-Environnement-Temps). Bien que dans la littérature, il existe d'autres approches toutes aussi intéressantes, nous avons choisi de nous pencher sur AGRE pour les raisons suivantes :

- AGR est une des approches organisationnelles les plus connues. Nous pesons partir d'une base solide et qui a déjà fait ses preuves dans le domaine des SMA.
- AGRE offre une prise en compte et au même titre de la dimension sociale et de la dimension spatiale en tant qu'environnements. Notre objectif est de considérer l'environnement temporel au même titre que l'environnement social et l'environnement spatial. AGRE nous offre cette possibilité.
- AGRE fonctionne avec un modèle d'interaction de type influence/réaction.
- AGRE définit un ensemble de notions de bases génériques (monde, espace, mode) qui se déclinent en notions spécifiques selon le type d'environnement (monde physique, surface, corps, organisation, groupe, rôle). La structure du modèle est également bien décrite à l'aide d'un diagramme UML. L'ajout de l'environnement temporel entraine une nouvelle déclinaison des notions génériques déjà en place. L'extension du modèle en ajoutant de la prise en compte de l'environnement temporel au même titre que les deux environnements déjà existants nous semble cohérente.

Toutefois, nous sommes conscients que d'autres approches comme MOISE [44] pourraient être tout aussi intéressantes. Une intégration de l'environnement temporel avec MOISE pourrait constituer une perspective intéressante.

Cette nouvelle considération de la dimension temporelle amène alors de nouvelles possibilités. Une d'entre elles constitue la deuxième contribution de la thèse. Elle s'appuie sur la première et illustre un raisonnement temporel qui en résulte. Notre proposition consiste à se servir de ce nouveau système doté d'un environnement temporel comme substrat dans le cadre d'une stratégie d'anticipation basée sur la perception de la dynamique temporelle.

2.3.2.3 Un raisonnement temporel : amélioration du raisonnement anticipatif par exploitation de la perception de l'environnement temporel

Nous avons introduit dans 2.1.2 les quatre classes d'anticipation selon Butz et al.. Chaque classe d'anticipation présente ses avantages et ses inconvénients. D'une manière générale, le raisonnement anticipatif dans la plupart des simulations multi-agent utilise uniquement les informations passés et présentes pour générer des prédictions sur le futur. Ce raisonnement semble éloigné de la réalité où nous anticipons sur la base d'informations passées, présentes, mais également sur la base du futur planifié. Notre approche est consiste alors à exploiter la visibilité sur la dimension future du temps offerte par la mise en place de l'environnement temporel dans le raisonnement anticipatif de l'agent.

Dans beaucoup d'approches d'anticipation que nous rencontrons dans la littérature dans les simulations multi-agent intègrent deux modèles : un modèle prédictif et un modèle de décision. Le modèle prédictif comprend obligatoirement un modèle des actions et un modèle des environnements. L'ensemble est appelé modèle du monde. Ce modèle a pour rôle de générer à sa sortie des prédictions d'un état ou d'un événement futur. Le modèle de décision quant à lui repose sur les prédictions générées par le modèle prédictif et propose un comportement ou une action.

L'approche d'anticipation que nous proposons diffère des approches classiques que nous pouvons voir actuellement dans la littérature. En effet, notre proposition exploite les percepts que les agents peuvent capter des différents environnements du système et plus particulièrement de l'environnement temporel dans notre modèle prédictif. C'est à ce niveau que se situe l'originalité de notre proposition. En effet, notre modèle de prédiction se sert du réglage de l'horizon temporel de perception pour percevoir des informations temporelles passées, présentes et futures. Dans ce cadre, la prise en compte du futur planifié dans le raisonnement anticipatif de l'agent est une nouveauté que nous introduisons dans cette thèse. L'agent couple ces informations temporelles avec des informations perçues au niveau d'autres environnements comme l'environnement spatial.

Il utilise le tout pour remettre en question son emploi du temps. L'objectif est de faire converger l'emploi du temps des agents, par ajustement successif, vers un emploi du temps plus pertinent. Nous y revenons plus en détail dans le chapitre 4.

2.3.3 Conclusion

L'objectif de cette thèse est de proposer une approche permettant d'optimiser la gestion de ressources partagées et limitées dans les villes intelligentes. Cette gestion se fait de manière distribuée. En d'autres termes, chaque entité du système participe par son comportement à la satisfaction de cet objectif collectif. Cependant, cette collaboration requiert que les composants du système interagissent pour échanger des informations utiles au collectif. Cela est permis par le biais d'un support d'interaction composé d'un milieu d'interaction et d'un modèle d'interaction. Une fois les informations partagées, il est nécessaire que chaque entité dispose d'un mécanisme interne lui permettant de traiter ces informations-là et de les prendre en compte lors de son choix de comportement. C'est à ce niveau qu'intervient le mécanisme de raisonnement. Afin d'avoir une idée plus précise concernant les approches déjà existantes relatives à ces problématiques et afin de déterminer leurs limites, nous avons établi un état de l'art. Cet état de l'art a fait ressortir deux aspects du temps sur lesquels nous basons nos contributions. Il s'agit de la représentation du temps et du raisonnement temporel.

Notre première contribution concerne la représentation du temps. Elle consiste à faire évoluer le paradigme agent de manière à considérer le temps comme un nouveau milieu d'interaction, au même titre que l'environnement spatial et l'environnement organisationnel. Nous appelons ce nouveau milieu d'interaction "environnement temporel" et nous le définissons de manière très simple, comme un espace dont la métrique est le temps. Ainsi, si dans les approches classiques, le mécanisme d'activation des agents se fait par lien direct entre l'agent et le sous-système que l'on appelle l'ordonnanceur. Désormais, dans notre proposition, l'environnement temporel vient s'interfacer entre l'agent et l'ordonnanceur de la simulation. Cette nouvelle approche casse le lien direct qui existe entre ces derniers. Le fonctionnement de l'environnement temporel vient alors compléter le fonctionnement de l'ordonnanceur de la simulation. Pour mettre tout cela en place, nous nous basons sur des modèles existants et qui sont déjà communément utilisés dans les SMA : le modèle générique d'organisation AGR et son extension AGRE. Nous détaillons ces notions ainsi que cette première contribution dans le chapitre 3 de ce manuscrit.

Cette nouvelle considération de la dimension temps amène alors de nouvelles possibilités en termes de raisonnement temporel. La deuxième contribution de la thèse s'appuie sur la première et illustre l'une de ces nouvelles possibilités. Nous nous servons de ce nouveau système doté d'un environnement temporel comme substrat de négociation afin de répondre à une problématique d'anticipation. L'approche que nous proposons se base sur un modèle de raisonnement sur la dynamique temps et prend en compte les trois positions dans le temps : passé, présent et futur. Nous y revenons plus en détail dans le chapitre 4.

De manière très concrète, si nous reprenons l'exemple du rechargement de véhicules électriques avec des bornes publiques décrites dans 1.1.2, l'objectif collectif est d'optimiser la répartition des rechargements dans l'espace et le temps. Au niveau individuel, l'objectif de l'automobiliste est d'optimiser son temps de rechargement. Celui de la borne est d'optimiser son occupation dans le temps.

L'environnement temporel permet alors aux automobilistes et aux bornes de recharge d'échanger des informations temporelles concernant leurs états et leurs activités. Ces informations temporelles peuvent être par exemple des informations concernant les automobilistes qui prévoient de se recharger à une borne à un instant donné, le taux d'occupation d'une borne ou la longueur prévisionnelle de la file d'attente au niveau d'une borne à un instant donné, etc. Ces informations temporelles, combinées aux informations spatiales et organisationnelles sont perçues et exploitées dans le cadre du raisonnement anticipatif intégré au sein de chaque composant du système (automobiliste ou borne). Ce raisonnement leur permet d'optimiser leur comportement de manière à satisfaire aux objectifs cités un peu plus haut. Ces objectifs émergent alors de l'interaction entre les agents par le biais des environnements : spatial, temporel et social.

Chapitre 3

AGRET : Un modèle intégrant l'environnement temporel dans les SMA au même titre que l'environnement spatial et l'environnement social

Nous introduisons ici notre première proposition. Cette contribution répond à la problématique du besoin en support d'interaction sur la dimension spatiale, organisationnelle et temporelle que nous avons décrit dans les chapitres 1 et 2 précédents. Cette solution est centrée sur une représentation du temps sous forme d'un environnement. Nous appelons cet environnement : l'environnement temporel [80]. Cela constitue notre contribution principale. L'environnement temporel est un "espace" dont la métrique est le temps. Nous tenons à bien souligner que notre objectif n'est pas de proposer un modèle de représentation de connaissances temporelles, mais un support permettant aux agents de partager et de percevoir des informations sur la dynamique temporelle.

Au niveau de la dimension temporelle, cet environnement temporel vient compléter le rôle que joue l'ordonnanceur de la simulation. L'ordonnanceur gère le cycle d'activation de la simulation tandis que l'environnement temporel agit comme un support permettant l'échange et le stockage d'informations temporelles.

Au niveau de la simulation, cet environnement temporel vient également en complément à l'environnement spatial et à l'environnement social qui existent déjà de manière classique dans les SMA. La relation entre l'agent et l'environnement se fait par influence-réaction.

L'ensemble est regroupé au sein d'un modèle que nous nommons AGRET (Agent-Group-Rôle-Environnement-Temps). AGRET est basé sur le modèle AGRE proposé par Ferber et al. [32] qui est une extension du modèle d'organisation multi-agent générique et connu AGR [30]. L'idée de base consiste à inclure la prise en compte de la dimension temps dans AGRE, au même titre que la dimension spatiale et la dimension organisationnelle qui y sont déjà.

Nous introduisons alors, dans la section 3.1 le modèle AGRET. Nous montrons par la suite dans 3.2 comment nous structurons cet environnement sur la base des principes du modèle à temporalité. Cela a également été abordé dans une de nos publications [80]. Dans 3.3, nous décrivons comment nous y intégrons le modèle d'interaction IRM4S. Nous y mettons notamment en valeur le changement que la mise en place de l'environnement temporel vient apporter au niveau du cycle d'activation de la simulation : les notions d'environnements contraints ou non contraints par le temps. Avant de terminer par la conclusion dans 3.5, nous présentons dans 3.4 un modèle de référence pour l'environnement temporel.

3.1 Agent-Groupe-Rôle-Environnement-Temps (AGRET)

La contribution principale de cette thèse repose sur la similitude et la complémentarité entre la dimension spatiale et la dimension temporelle que nous avons démontrées dans la section 2.2.1 du

chapitre 2 précédent. Notre objectif est de faire évoluer le paradigme agent, de manière à prendre en compte le temps comme un nouveau milieu d'interaction, au même titre que la dimension spatiale et la dimension sociale. Pour ce faire, nous nous reposons sur des approches communément utilisées dans la modélisation de l'espace et des organisations dans les simulations agents. Nous étendons, réadaptons et réappliquons ces approches à la dimension temps. L'approche que nous proposons ne remplace pas l'ordonnanceur de la simulation qui gère les mécaniques d'avancement du temps simulé et l'activation des agents et de la simulation. Au contraire, elle vient compléter le fonctionnement de ce dernier. L'environnement temporel vient s'interfacer entre l'agent et l'ordonnanceur de la simulation. L'ordonnanceur n'a pas d'équivalent au niveau spatial. Cela illustre justement les divergences qui peuvent exister entre le temps et l'espace physique à ce niveau. **L'environnement temporel gère la représentation du temps, les échanges et le stockage d'informations temporelles tandis que l'ordonnanceur de la simulation gère le cycle d'activation de la simulation**.

Dans leur article [32], Ferber et al. considèrent uniquement deux types de monde : les organisations (environnements sociaux) qui sont composées d'un ensemble de groupes, et le monde physique (environnements physiques) qui se compose de zones. Néanmoins, ils évoquent la possibilité d'existence d'autres types de monde et d'autres types d'espace qu'ils ne décrivent pas dans leur article [32]. Dans AGRET, nous reprenons les principes de base d'AGRE et nous rajoutons la dimension temps. Nous définissons alors un nouveau type d'espace qui est de type temporel et que nous appelons *agenda*. Comme dans AGR et AGRE, les agents sont situés dans les espaces et peuvent y agir à travers les modes. Si un mode dans une zone est appelé corps (body), un mode dans un groupe est appelé rôle (role), nous appelons un mode dans un agenda une **temporalité (temporality)**. Un nouveau type de monde vient également s'ajouter à ces deux types de monde déjà existants dans AGRE : le monde temporel (temporal world). La figure 3.1 montre un diagramme UML simplifié qui représente les relations entre mondes, espaces, zones, groupes, agendas, modes, corps, rôles et temporalités.

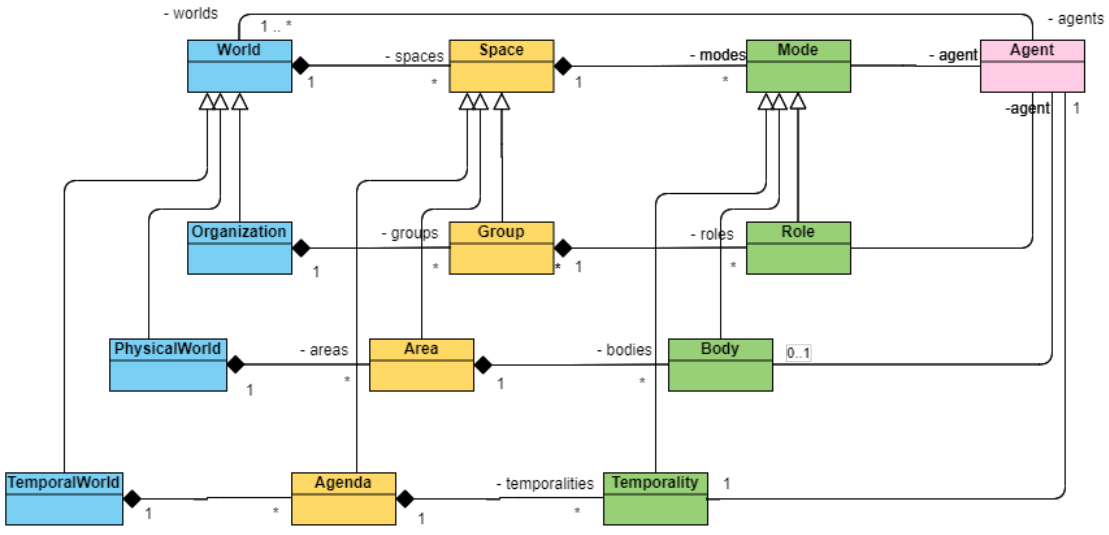


FIGURE 3.1 – Un méta-modèle UML du modèle AGRET

La relation d'agrégation qui lie l'espace au mode est remplacée par la même relation qui lie la zone au corps, le groupe au rôle et l'agenda à la temporalité. Dans la même logique, la relation d'agrégation reliant le monde à l'espace est remplacée par une agrégation qui lie l'organisation au groupe, le monde physique à la zone et le monde temporel à l'agenda. Ces différentes notions sont explicitées dans les sous-sections suivantes.

3.1.1 Le Monde (World)

Un monde propose les primitives requises pour qu'un agent puisse entrer dans un espace et en obtenir le mode. Ce mode est le seul moyen par lequel un agent peut agir dans un espace. Un agent

peut exister dans plusieurs mondes à la fois. Comme l'illustre la figure 3.2, les mondes sont utilisés comme points de départ pour que les agents entrent dans des groupes et des zones. Ils permettent également aux agents d'accéder à un ou plusieurs agendas.

Quand un agent est créé, il doit s'inscrire dans un monde. Par exemple, un agent social, c'est-à-dire un agent qui joue un rôle dans des groupes, doit s'inscrire en premier lieu à une organisation. Supposons que l'agent est inscrit à une organisation o, ensuite, pour entrer dans un groupe, il peut utiliser la primitive :

$$Role\ r = o.demanderRole(NomGroupe, TypeRole, NomRole, a)\ [32] \tag{3.1}$$

qui lui donne la possibilité de faire une requête pour entrer dans un groupe afin de jouer le rôle du type $TypeRole$ avec l'autorisation a. Si cela est possible, l'agent obtiendra un rôle à travers lequel il aura la possibilité d'agir au sein d'un groupe.

De même, un agent situé physiquement (possédant un corps) dans une zone doit s'inscrire en premier lieu dans un monde physique. Supposons que l'agent soit inscrit dans un monde physique p, ensuite, pour entrer dans une zone, il doit utiliser la primitive :

$$Corps\ c = p.demanderCorps(NomZone, TypeCorps, Localisation, a)\ [32] \tag{3.2}$$

qui lui donne la possibilité de faire une requête pour posséder un corps lui permettant d'avoir une existence physique dans une zone avec l'autorisation a. Si cela est possible, l'agent obtiendra un corps à travers lequel il aura la possibilité d'agir au sein de cet espace physique.

Dans la même logique, un agent situé dans la dimension temps devra s'inscrire dans un monde temporel. Supposons que l'agent est inscrit dans un monde temporel mt, ensuite pour avoir accès à un agenda, il doit utiliser la primitive :

$$\begin{aligned} Temporalite\ tmp = mt.demanderTemporalite(&NomAgenda,\\ &TypeTemporalite,\\ &LocalisationTemporelle,\\ &a) \end{aligned} \tag{3.3}$$

qui lui donne la possibilité de faire une requête pour posséder une temporalité lui permettant d'avoir une existence temporelle au sein d'un agenda avec l'autorisation a. $LocalisationTemporelle$ permet de spécifier l'ensemble des paramètres d, f, p, v qui définissent la position de la temporalité dans l'environnement temporel : début, fin, période, variabilité. Nous détaillons cela dans la section 3.2 suivante. Pour faire le parallèle avec la localisation spatiale de l'agent à un instant t, ces paramètres déterminent la position de l'agent dans l'environnement temporel à un slot temporel t. Nous expliquons cela dans la sous-section 3.3.3.

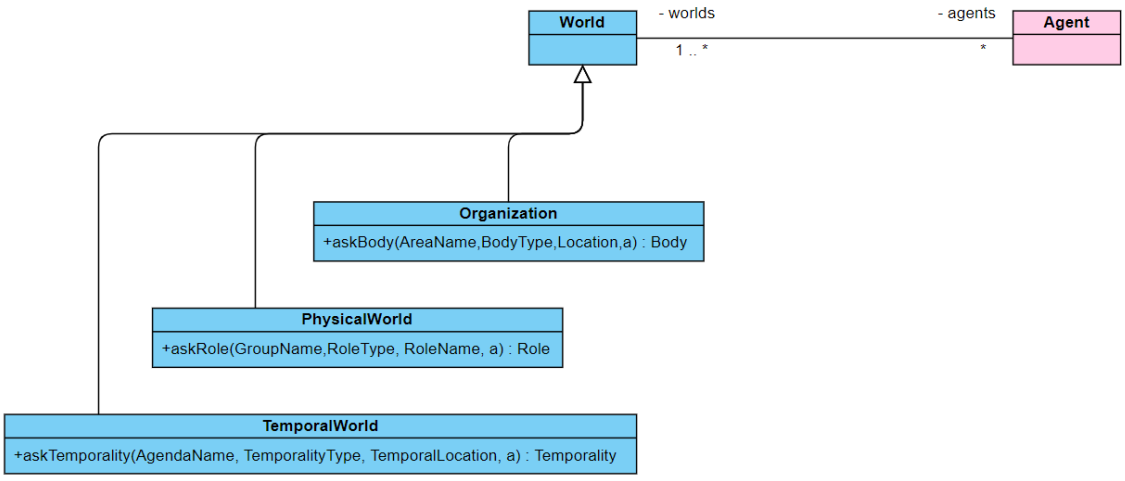


FIGURE 3.2 – Un méta-modèle UML du modèle AGRET : Le monde

3.1.2 L'espace (Space)

De la même manière que l'organisation peut être composée de différents groupes, le monde physique de différentes zones, le monde temporel est composé d'agendas. Il peut exister plusieurs types d'agendas relatifs à différentes thématiques. En fonction de la thématique et des objectifs du modèle, ces agendas peuvent être personnels ou partagés. Si nous faisons le parallèle avec ce qui se fait dans le monde réel, chaque individu possède un agenda personnel dans lequel il note ses activités personnelles. Dans le cadre de l'organisation d'une activité collective comme une réunion par exemple, un moyen de trouver un créneau horaire qui convient à un nombre maximum de participants est d'utiliser un agenda partagé ou collaboratif comme Evento, Doodle, etc. Il existe également des agendas qui sont accessibles par des groupes de personnes ayant les autorisations nécessaires. C'est le cas des agendas du personnel enseignant de l'université par exemple.

3.1.3 Le Mode (Mode)

Une fois l'une des primitives précédentes utilisée, toutes les compétences associées au mode sont disponibles à l'agent. Dans le cas de l'organisation, l'agent aura naturellement la possibilité d'envoyer des messages aux agents au sein de ce groupe en utilisant une primitive de ce genre :

$$r.sendMessage(RoleName, Message) \text{ [32]} \tag{3.4}$$

qui exprime une requête pour le rôle lui permettant d'envoyer un message aux agents ayant le rôle $RoleName$ dans un groupe.

Dans le cas d'un monde physique, en fonction des capacités du corps, l'agent peut se déplacer, saisir, etc. avec des commandes telles que :

$$c.bouge(x, y) \text{ [32]} \tag{3.5}$$

où x et y sont des coordonnées géographiques. Cette requête permet de déplacer le corps de l'agent vers le point de coordonnées (x, y).

De même, dans le cas d'un monde temporel, l'agent pourra prévoir une activité future en utilisant la primitive suivante :

$$tmp.creer(d, f, p, v) \tag{3.6}$$

Cette requête consiste à demander la création d'une nouvelle localisation temporelle de l'agent au niveau de l'environnement temporel.

L'environnement temporel reprend les principes de base du modèle à temporalité, une des approches d'ordonnancement du temps dans les simulations multi-agent que nous avons introduite dans la section 2.2.2.2 du chapitre 2 précédent. Ces principes servent dans la mécanique et la structuration de l'environnement temporel. Afin de ne pas porter confusion, nous tenons à préciser que la réutilisation des principes de bases du modèle à temporalité dans l'environnement temporel est indépendante de l'approche utilisée pour l'ordonnancement de la simulation. Nous préconisions, l'utilisation d'une approche de type modèle à temporalité comme approche d'ordonnancement du temps. Cependant, l'environnement temporel tel que nous le définissons est assez générique pour fonctionner aussi bien avec d'autres types d'approches comme l'approche à pas de temps constant, l'approche événementielle, ou l'approche hybride.

3.2 Le modèle à temporalité [71] pour structurer l'environnement temporel

Chaque environnement est structuré différemment selon son type. Par exemple, l'environnement spatial peut être structuré à partir de SIG, d'espace continu, de grille. L'environnement social peut être structuré à partir de réseaux. Dans notre cas, nous proposons une structuration de l'environnement temporel sur la base du modèle à temporalité.

Le modèle à temporalité est un type particulier d'approche d'ordonnancement du temps dans les simulations multi-agent. C'est une approche dite expressive, car elle permet à l'agent de décrire lui-même sa propre dynamique temporelle. Cela se fait par définition d'une structure de donnée appelée *temporalité*. Un agent définit une ou plusieurs temporalités en fonction de la complexité et de la diversité des activités qu'il souhaite entreprendre. Son comportement résulte de l'exécution de l'ensemble de ces activités.

Definition 3.2.1. Temporalité au niveau de l'ordonnanceur. Au niveau de l'ordonnanceur, une temporalité est la description d'une série d'occurrences temporelles. Cette description spécifie les points de l'axe du temps, pour lesquels l'agent souhaite que l'ordonnanceur de la simulation déclenche l'exécution de son comportement.

Une temporalité t est définie formellement par :

$$t = \{id, d, f, p, v\} \tag{3.7}$$

— id : l'identifiant de la temporalité. Ce paramètre permet à l'agent d'identifier le contexte dans lequel il est activé ;
— $[d, f]$: l'intervalle de temps durant lequel la temporalité est applicable ;
— p : la période qui définit la série d'occurrences (p = 0 si l'action est ponctuelle) ;
— v : la variabilité, c'est-à-dire un nombre réel définissant la précision en dessous de laquelle l'occurrence temporelle reste valide.

Une temporalité provoquera le déclenchement de l'exécution du code du comportement de l'agent à toutes les dates x vérifiant $x = d + p * k$, avec k un entier tel que $0 \leq k \leq n$, et n le plus grand entier vérifiant $(d + p * n) = f$. On nomme cet ensemble de dates : dates de déclenchement de la temporalité.

3.2.1 Fonctionnement et gestion interne

Les temporalités sont établies à la phase d'initialisation de la simulation suite à l'invocation d'une méthode particulière sur chacun des agents. Par la suite, ces derniers peuvent à tout moment redéfinir ou créer de nouvelles temporalités. Cela leur permet d'ajuster leur dynamique comportementale en fonction de la situation.

Le traitement interne des temporalités par l'ordonnanceur fait intervenir deux structures de donnée comme le montre la figure 2.5 du chapitre 2 précédent :

Definition 3.2.2. Slot temporel. Élément d'ancrage correspondant à un point de l'axe du temps sur lequel le simulateur va être amené à déclencher l'exécution du code de comportement d'un agent.

Definition 3.2.3. Tempo. Structure regroupant l'ensemble des temporalités qui à un instant donné sont situées sur un même slot temporel et qui ont la même valeur de période. C'est cette valeur de période qui caractérise le tempo en traduisant le fait que les temporalités qu'il comporte ont toutes le même "rythme".

L'ordonnanceur fait progresser le temps simulé en l'emmenant successivement sur chacune des positions signalées par un slot temporel. À chaque slot temporel, tous les tempos sont traités. Cela produit l'exécution de tous les comportements associés aux temporalités qu'ils comportent.

3.2.2 Réutilisation du modèle dans le cadre de l'environnement temporel

Nous reprenons ces mêmes principes de base dans notre modèle d'environnement temporel (temporalité, slot, tempo). Ils nous servent à structurer les informations temporelles contenues dans ce milieu d'interaction. Ils permettent également la mise en place de la mécanique environnementale.

Dans l'approche originale du modèle à temporalité proposé par Payet et al. [71], aucun support n'a été prévu pour le stockage des informations passées. L'ordonnanceur calcule les prochaines dates de déclenchement des temporalités futures avant chaque avancement du temps. Il supprime également les temporalités passées, car elles n'interviennent plus dans le cycle d'activation de la simulation. De plus, il n'existe pas de mécanisme permettant la perception d'informations sur la dynamique d'activation temporelle des agents. En effet, ces fonctionnalités ne sont pas nécessaires compte tenu de l'objectif pour lequel l'approche a été originellement conçue : l'ordonnancement de la simulation.

Cependant, le stockage et la perception des informations temporelles sont indispensables dans l'usage que nous en faisons. Nous travaillons alors sur la refonte de l'ancien modèle, en l'élargissant avec deux aspects. Nous mettons en place un support permettant aux agents, non seulement de partager des informations sur leur dynamique temporelle (comme c'est déjà le cas dans le modèle à temporalité), mais également de stocker et de percevoir des informations positionnées sur le passé, sur le présent et sur le futur planifié par les agents.

Definition 3.2.4. Temporalité dans l'environnement temporel. Dans l'environnement temporel, la notion de temporalité est un peu plus étendue. Nous la définissons comme étant la **représentation de l'agent dans l'environnement temporel**, au même titre que le corps de l'agent dans l'environnement spatial ou le rôle dans l'environnement social.

Le corps possède une localisation spatiale dans l'environnement physique. Son déplacement se traduit par la modification de sa localisation spatiale. De la même manière, une temporalité possède une localisation temporelle dans l'environnement temporel. Son déplacement se traduit par la modification de sa localisation temporelle. C'est cette notion de localisation temporelle dans l'environnement temporel qui est équivalente à la notion de temporalité dans le modèle à temporalité (au niveau de l'ordonnanceur). Cette localisation temporelle est définie par $t = \{id, d, f, p, v\}$ où id, d, f, p, v sont les paramètres obligatoires de la temporalité qui permettent d'établir l'axe temporel des agents du système. Son déplacement dans l'environnement temporel se traduit alors par la modification de sa localisation temporelle, plus précisément par la modification des différents paramètres id, d, f, p, v. Nous y revenons plus en détail dans la sous-section 3.3.3. d, f, p, v sont des paramètres objectifs. C'est-à-dire que leur valeur se calcule à partir de données réelles, sans prendre en compte du ressenti ou des préjugés de l'agent. D'autres paramètres optionnels peuvent être rajoutés en fonction des besoins et du but du modélisateur. Ces paramètres peuvent être objectifs ou subjectifs. Par exemple : le coût, la criticité, la priorité ou la pondération sont des paramètres subjectifs. Elles correspondent à l'évaluation de l'importance que l'agent accorde à l'occurrence temporelle, à un ressenti de l'agent. Un exemple de paramètre optionnel objectif est la longueur prévisionnelle de la file d'attente. Nous verrons un exemple d'utilisation de ces paramètres optionnels dans notre deuxième contribution que nous présentons dans le chapitre 4 suivant.

Au niveau de l'ordonnanceur, dans le modèle à temporalité, les créations et les modifications de temporalités sont immédiatement traitées par l'ordonnanceur. Pour cela, il actualise les structures de représentation lui servant à déterminer la façon dont le temps simulé va s'écouler et les activations à réaliser au cours de sa progression. Ainsi, comme dans la plupart des approches classiques d'ordonnancement du temps dans les simulations agents, l'activation des agents par l'ordonnanceur se fait par lien direct entre ce dernier et l'agent à activer. Dans l'approche que nous proposons, nous cassons ce lien direct en interfaçant l'environnement temporel entre l'agent et l'ordonnanceur (cf figure 3.3). Le processus de modification ou de création de temporalité se fait alors comme la suivante : à un slot de temps s :

1. L'ordonnanceur traite les demandes d'activation d'agents pour le slot courant s. Il réveille les agents qui se sont exprimés vouloir s'activer à s.
2. Les agents réveillés enclenchent leur cycle perception-décision-action/influence. Ils perçoivent leur contexte d'activation (phase de perception). Notamment au niveau de l'environnement temporel, ils perçoivent les informations relatives à l'action qu'ils ont prévu d'effectuer à ce slot s. Ils sont libres de choisir par eux même d'agir ou non (phase de décision). Ils définissent, modifient ou suppriment une ou plusieurs localisations temporelles au niveau de l'environnement temporel (phase d'influence). Ces définitions ou ces modifications de localisations temporelles ne sont pas traitées immédiatement par l'environnement temporel.
3. L'environnement temporel attend la fin de l'activation de tous les agents du système pour traiter l'ensemble des définitions et des modifications de localisations temporelles. Il s'occupe de leur stockage et de l'actualisation des structures de représentation de la dynamique temporelle des agents (rattachement à un slot, regroupement en tempos, etc.). Nous y revenons plus en détail dans 3.3.
4. L'ordonnanceur est mis à jour en fonction des données contenues dans l'environnement temporel. Il actualise les structures de représentation lui servant à déterminer la façon dont le temps simulé va s'écouler et les activations à réaliser au cours de sa progression.

Tout ce processus se fait sur le même slot de temps.

Dans l'approche que nous proposons, la localisation temporelle est traitée et stockée au niveau de l'environnement temporel afin que les agents puissent non seulement les créer ou les modifier, mais également les percevoir. En complément à cela, nous laissons à l'ordonnanceur de la simulation le soin de s'occuper de la gestion de l'écoulement du temps et de l'activation des agents.

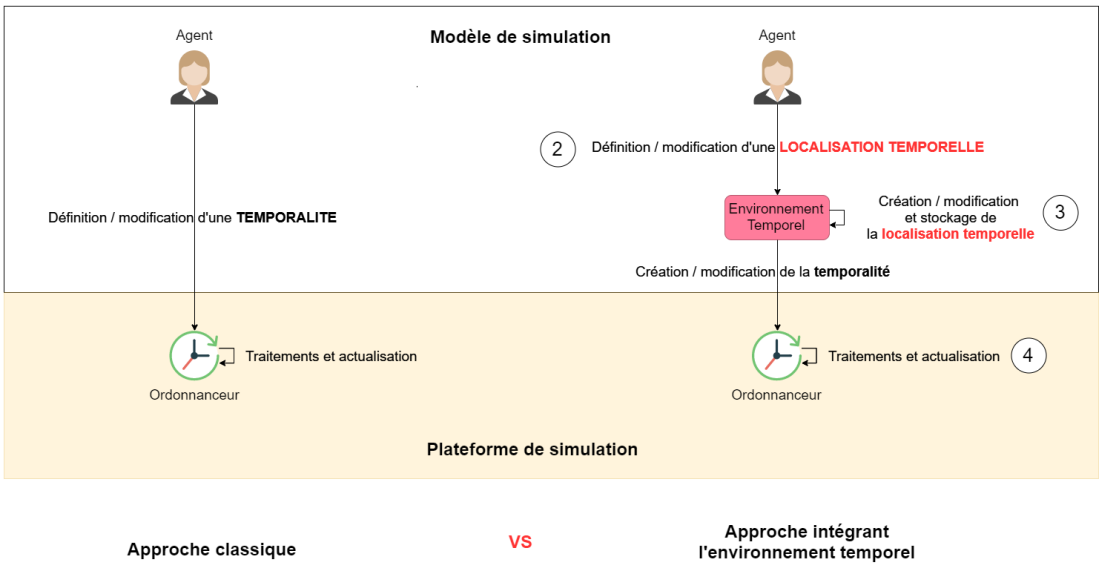


FIGURE 3.3 – Relations entre agents, environnement temporel et ordonnanceur.

3.3 Intégration du modèle influence-réaction pour la simulation (IRM4S)

Ferber et al. [33] critiquent AGRE et toutes les familles de modèles de type AGR (dont AGRET) du fait qu'ils ne fournissent pas de théorie de l'action qui prendrait en compte les actions concurrentes. Afin d'y remédier, une des solutions proposées consiste en l'utilisation du modèle influence/réaction [31] [33] pour gérer l'interaction entre l'agent et l'environnement.

3.3.1 IRM4S comme modèle d'interaction

Nous avons présenté dans la sous-section précédente le modèle sur lequel se base la structure et la mécanique de l'environnement temporel. Nous présentons maintenant le modèle d'interaction faisant le lien entre les agents et l'environnement temporel. Une description détaillée des différents modèles d'interaction a été faite dans la section 2.2 du chapitre 2 de cette thèse. Cette sous-section résume ce qui a été vu avant, afin d'introduire son utilisation dans le cadre de notre contribution.

Le cycle comportemental d'un agent a se résume en trois phases : une phase de perception, une phase de délibération et une phase d'action. Ce cycle comportemental peut se représenter par une fonction $Behaviour_a$.

3.3.1.1 Approche classique action/réaction

Dans une approche classique, le cycle comportemental d'un agent aboutit à la création d'un nouvel état global de l'environnement :

$$Behaviour_a : \Sigma \mapsto \Sigma \tag{3.8}$$

Cette fonction correspond à l'application successive de trois (3) fonctions :
- la fonction de perception qui calcule un percept à partir de l'état du système ;
- la fonction de mémorisation qui calcule le nouvel état interne de l'agent à partir du percept généré précédemment ;
- la fonction de décision qui modifie le monde suivant l'action de l'agent.

Cette approche pose des difficultés techniques et conceptuelles. Non seulement elle ne permet pas de modéliser ou de traiter facilement la simultanéité, mais elle viole également la contrainte d'intégrité environnementale et la propriété d'autonomie des agents du système. Nous constatons également un couplage fort entre agent, action et environnement. Ce couplage rend le modèle de simulation extrêmement sensible à la manière dont il est implémenté. Afin de nous affranchir de ces limites, nous nous tournons vers une approche de type influence/réaction.

3.3.1.2 Modèle Influence/Réaction

Contrairement au modèle action/réaction qui aboutit à la modification directe de l'état de l'environnement, dans le modèle Influence/Réaction, le cycle comportemental d'un agent aboutit à la production d'influences [31] :

$$Behaviour_a : \Gamma \mapsto \Gamma \tag{3.9}$$

Ce modèle introduit la notion d'état dynamique de l'environnement, noté $\delta \in \Delta$. Il s'agit d'une paire notée $< \sigma, \gamma >$ où $\sigma \in \Sigma$ est l'état de l'environnement (les variables environnementales) et où $\gamma \in \Gamma$ décrit l'ensemble des influences.

Ainsi, grâce aux deux notions : influence et réaction, l'idée majeure de cette approche est de faire une distinction claire entre la dynamique du niveau agent et la dynamique du niveau multi-agent.

Un changement s'opère alors au niveau des trois fonctions qui composent *behaviour* $Behaviour_a$:
- La fonction perception ne transforme plus l'état de l'environnement, mais les influences en percepts.
- La fonction mémorisation reste inchangée.
- La fonction décision ne modifie plus directement le monde, mais indique quelle opération exécuter (influence).

Pour articuler le niveau agent et le niveau multi-agent, la dynamique globale est décomposée en deux fonctions :
- La fonction exécution qui produit les influences, au niveau agent.
- La fonction réaction qui intègre la réaction du monde aux influences, au niveau multi-agent.

Dans l'approche que nous proposons, nous faisons une claire distinction entre ces deux phases : phase d'influence et phase de réaction. Dans la sous-section 3.2.2 précédente, la phase d'influence correspond à la dernière phase de l'étape numéro 2 du processus de modification ou de création de la localisation temporelle. La phase de réaction correspond à l'étape numéro 3.

Ferber et al. [32] proposent une "simplification" et une clarification du modèle influence/réaction dans le cadre de la simulation multi-agent. Le modèle s'appelle IRM4S. Il s'agit d'une approche adaptée dans un contexte de simulation multi-agent.

Comme dans toute approche de type influence/réaction, dans IRM4S, le cycle comportemental de l'agent aboutit à la production d'influences. Cependant, dans le modèle influence/réaction, les agents perçoivent ce qui les influence, mais ne sont pas influencés par l'état de l'environnement. Contrairement à cela, IRM4S prend en compte la perception locale et subjective de son environnement par un agent. Cela se traduit par la présence de l'état de l'ensemble des variables environnementales Σ, en entrée à la fonction $Behaviour_a$

Au niveau agent, les trois (3) fonctions composant la fonction behaviour deviennent alors :
- $Percept_a : \Sigma \times \Gamma \mapsto P_a$. Dans le modèle classique d'influence/réaction, cette fonction prend en entrée uniquement des influences. Contrairement à cela, dans IRM4S, cette fonction prend en entrée, en plus des influences, l'ensemble des variables environnementales, c'est-à-dire l'état de l'environnement.
- Les fonctions de mémorisation et de décision quant à elles restent inchangées.

Cette perception de l'état de l'environnement est importante dans l'approche que nous proposons. Elle permet aux agents d'extraire un ensemble d'informations spatiales, sociales et temporelles à partir des supports d'interaction qui dans notre cas sont les environnements. Ces informations servent par la suite dans le cadre du raisonnement anticipatif. Ce raisonnement est notre deuxième proposition que nous décrivons dans le chapitre 4 suivant.

Comme mentionné précédemment, un des intérêts majeurs du principe influence/réaction et de IRM4S est de bien distinguer ce qui est propre aux agents de ce qui se passe dans l'environnement. L'environnement possède alors une dynamique endogène et produit lui aussi des influences. L'évolution endogène de l'environnement découle aussi directement de l'état dynamique du système et est définie par $Natural_w : \Sigma \times \Gamma \mapsto \Gamma$.

Au niveau multi-agent, la dynamique globale est également décomposée en deux fonctions :
- $Influence : \Sigma \times \Gamma \mapsto \Gamma'$ définit globalement les influences produites au niveau micro (agents et environnement). $\gamma'(t) \in \Gamma'$

— *Reaction* : $\Sigma \times \Gamma' \mapsto \Delta$ qui intègre la réaction du monde aux influences au niveau multi-agent. où Δ est l'état dynamique de l'environnement défini précédemment par la paire $< \sigma, \gamma >$

Ce modèle a été particulièrement conçu pour répondre au problème de la simultanéité. Il a également l'avantage de faire le lien entre les dynamiques des niveaux micro (agent) et macro (système multi-agent). De plus, il sépare, de manière claire tout ce qui relève de l'agent de tout ce qui relève de l'environnement.

Pour rappel, son principe se base sur l'idée selon laquelle un agent ne peut pas directement changer l'état du monde, mais peut tout simplement influencer ses dynamiques. En d'autres termes, un agent ne peut que décider de l'action suivante à faire. C'est à son environnement de déterminer les conséquences de cette dernière. L'environnement dispose alors de ses propres modalités de fonctionnement qui s'imposent à l'agent. Par exemple, un agent veut envoyer un message et effectue l'opération qui consiste à l'envoyer. Son environnement se charge de le transmettre et de le délivrer à son destinataire. De la même manière, un agent décide de bouger, mais c'est son environnement (son corps et son environnement extérieur) qui exécute le déplacement. En suivant la même logique au niveau de l'environnement temporel du modèle AGRET : un agent peut définir une localisation temporelle. Cependant, c'est son environnement temporel qui se charge de la créer ou pas, en fonction des autres influences produites par les autres agents, en fonction des lois de l'environnement temporel, etc. Ainsi, l'environnement réagit aux influences produites par les agents.

Cette séparation stricte de tout ce qui relève de l'agent et tout ce qui relève de l'environnement se traduit par la séparation du cycle d'activité classique d'un simulateur utilisant un type d'ordonnancement à temps discret en deux phases :

— Une phase d'activation de l'environnement durant laquelle le simulateur doit mettre à jour l'horloge virtuelle, initialiser le prochain cycle, etc. ;
— Une phase d'activation des agents durant laquelle l'agent applique le processus itératif : perception, mémorisation, décision.

Dans cette section 3.1 où nous avons présenté les principes de bases du modèle AGRET, nous nous sommes principalement concentrés sur la phase d'activation des agents. Dans la sous-section 3.3.2 suivante, nous nous focaliserons plus sur la phase d'activation de l'environnement.

3.3.2 Un environnement temporel non contraint par le temps

Dans la plupart des simulateurs, la phase d'activation de l'environnement se déroule en début du cycle d'activité de la simulation. Cela est dû au fait que la plupart des environnements a besoin de calculer son nouvel état en début d'un cycle (instant t), avant l'activation des agents. Ces calculs doivent s'établir sur l'intervalle de temps qui s'est écoulé depuis la dernière activation, c'est-à-dire dans l'intervalle $]t - dt, t]$. Ces environnements ont des états qui évoluent dans le temps en fonction des lois inertielles qui leur sont propres, et des influences qu'exercent les agents. Ce sont donc, en quelque sorte, des **environnements contraints par le temps**.

Dans notre contexte, la mise en place d'un nouveau type d'environnement qui est l'environnement temporel vient chambouler le fonctionnement de ce cycle classique d'ordonnancement de la simulation. Par définition, l'état de l'environnement temporel détermine la dynamique d'écoulement du temps. Par conséquent, son état évolue hors du temps. En effet, l'état dynamique de l'environnement temporel reste soumis à des lois inertielles (la mécanique des slots temporels) et aux influences exercées par les agents sous forme de "localisations temporelles". Cependant, cela se fait indépendamment de l'intervalle de temps écoulé depuis les derniers calculs. Seule la connaissance du dernier slot temporel courant est nécessaire pour servir de point de référence aux nouvelles localisations temporelles exprimées par les agents, et aux déplacements des slots temporels périodiques. Ainsi, **l'environnement temporel n'est pas un environnement contraint par le temps**. De manière concrète, le calcul du prochain slot d'activation temporel $(t + dt)$ est fonction de l'ensemble des influences exercées par les agents durant le slot temporel courant (t) ainsi que des lois de l'environnement temporel. Le traitement des calculs au niveau de l'environnement temporel doit alors se faire obligatoirement avant, $t + dt$ car il permet de déterminer $t + dt$. Plus précisément, ce traitement doit se faire à la fin du cycle courant, car il faut attendre la fin du cycle d'activation de tous les agents et le calcul des réactions de l'environnement.

Ce raisonnement nous amène donc aux conclusions suivantes :

— les environnements contraints par le temps doivent être activés en début de cycle pour leur permettre de calculer leur nouvel état sur l'intervalle de temps écoulé $]t - dt; t]$.

— les environnements non contraints par le temps doivent être activés en fin de cycle pour leur permettre de calculer leur nouvel état en fonction du dernier slot temporel courant t.

Ainsi, aux deux phases du cycle d'activation classique de la simulation s'ajoute alors une troisième phase comme le montre la figure 3.4. Les trois nouvelles phases du cycle d'activation de la simulation sont :

1. la phase d'activation des environnements contraints par le temps sur l'intervalle de temps $]t(i-1), t]$;
2. la phase d'activation des agents sur le slot temporel courant t ;
3. la phase d'activation des environnements non contraints par le temps sur la considération que t vient d'être passé $]t, t(i+1)]$;

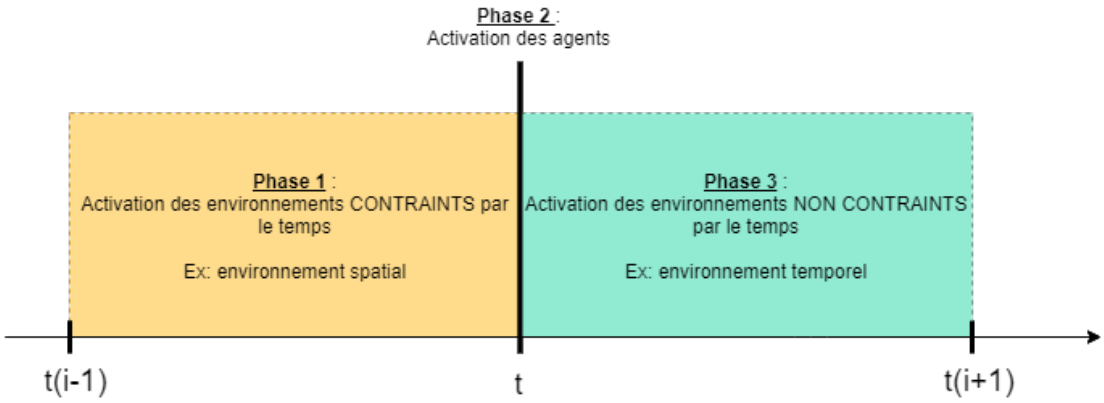


FIGURE 3.4 – Nouveau cycle d'activation de la simulation

3.3.3 Application du modèle IRM4S à l'environnement temporel

Afin d'illustrer le fonctionnement complet du cycle d'activité du simulateur, nous prenons l'exemple d'une application dans le cadre d'une simulation basée sur IRM4S (influence réaction pour la simulation). Comme expliqué dans 3.3.1, IRM4S fait une distinction claire et nette entre le temps correspondant à la gestion du niveau micro (phase d'influence) et le temps correspondant à la gestion du niveau macro (phase de réaction).

3.3.3.1 Au niveau agent (phase d'influence)

Les mécaniques de l'environnement temporel tel que nous le concevons se basent sur les mécaniques de l'approche d'ordonnancement de type modèle à temporalité. Ce choix se justifie par les avantages que le modèle peut apporter en termes de performance et d'expressivité des agents. De plus, son fonctionnement est en accord avec les principes du modèle d'interaction de type influence/réaction.

Nous apportons un nouveau dynamisme afin que ce concept qui à l'origine a été pensé pour l'ordonnancement de la simulation puisse s'adapter à un concept d'environnement. Dans ce cadre, nous redéfinissons certains termes et concepts de base. À partir de maintenant, tous les termes et références qui seront utilisés dans le cadre de l'environnement temporel se baseront sur les définitions qui suivent. Nous tenons également à bien préciser, afin de ne pas porter confusion, qu'il est important de bien faire la distinction entre environnement temporel (qui relève de l'environnement) et modèle à temporalité (qui relève de l'ordonnanceur).

Dans le modèle à temporalité, une temporalité $t = [id, d, f, p, v]$ est **statique** dans le temps. Cela veut dire que la valeur de ses paramètres ne varie pas en fonction du temps. Dans un environnement temporel, cette temporalité devient **dynamique**. Plus précisément, sa localisation temporelle, qui correspond à la valeur de l'ensemble de ses paramètres, peut varier dans le temps. Cela traduit le *déplacement* de l'agent dans l'environnement temporel.

Definition 3.3.1. Temporalité dans l'environnement temporel. La temporalité est le mode d'existence de l'agent, son apparence dans l'environnement temporel, au même titre que le rôle pour l'organisation et le corps pour l'espace.

La temporalité intègre alors des capteurs et actionneurs permettant aux agents de percevoir et d'influencer l'environnement temporel. Dans le contexte de l'interaction de l'agent avec son environnement temporel, cette influence consiste en une demande de création et de stockage ou de modification d'une *localisation temporelle* l.

Definition 3.3.2. Localisation temporelle. Une localisation temporelle

$$l = \{id, d, f, p, v, ...\} \tag{3.10}$$

est le t-uple qui traduit un besoin d'activation de l'agent à un slot de temps t. Ce sont les paramètres obligatoires. À ces paramètres peuvent se rajouter d'autres paramètres optionnels, définis en fonction du modèle et des objectifs de la simulation. Par exemple : dans notre cas, nous rajoutons un paramètre que nous appelons *satisfaction*, noté s. La valeur de ce paramètre est vide pour une localisation temporelle située dans le présent ou le futur. Par contre, elle prend une valeur comprise entre l'intervalle $[0, 1]$ pour une localisation temporelle passée. Cette valeur correspond à une note que l'agent attribue à la localisation temporelle passée après avoir évalué l'exécution du comportement relatif à cette localisation temporelle. Il s'agit d'un des rares paramètres d'une localisation temporelle passée qui peut être modifié par l'agent. Une note qui se rapproche de 1 veut dire que l'agent estime que l'action s'est bien passée. Au contraire, plus la note se rapproche de 0, plus l'agent considère que l'action ne s'est pas passée comme il l'aurait souhaité. Par exemple : Supposons une localisation temporelle qui correspond à un rechargement sur une borne. Cependant, l'agent n'a pas pu se recharger parce que la borne n'était finalement pas libre. Il pourrait donner une note qui se rapproche de 0 à cette localisation temporelle. La valeur de ce paramètre nous sert entre autres dans le cadre du raisonnement anticipatif que nous expliquons plus en détail dans 4.2.3.4.

Une localisation temporelle est rattachée à un slot temporel situé sur l'axe temporel de l'environnement temporel et peut être regroupée en tempos. Les notions de slots temporels et de tempos restent les mêmes que dans le cadre du modèle à temporalité.

Definition 3.3.3. Fenêtre temporelle ou horizon temporel de perception. Pour chaque agent ou chaque type d'agent, nous définissons une fenêtre temporelle ou horizon temporel de perception noté $\Delta_p = [\delta_{pmin}, \delta_{pmax}]$. L'horizon temporel permet aux agents de ne percevoir que des localisations temporelles rattachées aux slots temporels compris dans l'intervalle $[t - \delta_{pmin}, t - \delta_{pmax}]$, où t est un slot temporel. Cela traduit une propriété des agents : celle d'avoir une perception limitée de son environnement. Cette fenêtre temporelle de perception est l'équivalent temporel du champ de vision dans l'environnement spatial.

La perception de l'agent de son environnement temporel est contrainte par cette *fenêtre temporelle de perception* Δ_p et par un ensemble de règles qui définissent les conditions d'accès aux informations contenues dans l'environnement temporel. Cet ensemble de règles peut comprendre des lois de l'environnement, des règles d'accès aux informations en lien avec l'environnement social (système de réseau social) ou des paramètres définis par le concepteur, permettant de contrôler l'accès à certaines localisations temporelles. La perception devient alors

$$Perception : \Sigma \times \Gamma \times \Delta_p \mapsto P \tag{3.11}$$

où Σ est l'état de l'environnement, Γ est un ensemble d'influences, Δ_p est la fenêtre temporelle de perception et P est un ensemble de percepts.

3.3.3.2 Au niveau multi-agent (phase de réaction)

Traitement des localisations temporelles. Le traitement des localisations temporelles est le même que dans le cas de l'approche de type modèle à temporalité. Le monde temporel intègre un **axe temporel**. La localisation temporelle définie par un agent est rattachée à cet axe, au niveau d'un **slot temporel** t. t est ponctuel et traduit l'instant sur lequel la temporalité de l'agent est située dans le temps. Il est important de ne pas confondre la situation de la temporalité dans

le temps avec la localisation de la temporalité dans l'environnement temporel. La situation de la temporalité dans le temps est relative à l'horloge globale de la simulation, est gérée par l'ordonnanceur et correspond à un slot temporel t. La localisation de la temporalité dans l'environnement temporel, quant à elle, correspond à $l = \{id, d, f, p, v\}$ au niveau de l'environnement temporel. Comme dans le modèle à temporalité, les localisations temporelles de même période peuvent être regroupées dans un tempo.

Stockage des localisations temporelles. Le mécanisme utilisé au niveau de l'environnement temporel diffère également de celui utilisé dans le cadre du modèle à temporalité au niveau du stockage des localisations temporelles. En effet, dans le modèle à temporalité, aucun support n'est prévu pour le stockage des activations temporelles (temporalités au niveau de l'ordonnanceur) passées. Ces dernières sont supprimées après traitement par l'ordonnanceur. Ce fonctionnement est tout à fait cohérent dans le cadre d'une utilisation dans un contexte d'ordonnancement de la simulation pour les raisons suivantes :

— Cela permet d'améliorer les performances en libérant de la mémoire.
— Il n'est pas nécessaire, voire inutile, de stocker les activations temporelles passées dans le sens où du point de vue de l'ordonnanceur, ces dernières n'ont aucune incidence sur les activations temporelles futures. Au contraire, stocker ces informations ne ferait qu'alourdir le système.

Contrairement à cela, l'environnement temporel se doit de stocker les localisations temporelles : passées, présentes et futures. Cependant, la pertinence de ces informations varie en fonction du temps, du modèle et du type d'information. Une information passée peut avoir une durée de vie au-delà de laquelle elle n'est plus pertinente. De même, au-delà d'une certaine limite dans le temps, une information future n'est pas forcément intéressante. Afin de gérer cela, nous mettons en place une *fenêtre temporelle de stockage* dont les valeurs des bornes sont définies arbitrairement par l'utilisateur ou le concepteur du modèle.

Definition 3.3.4. Fenêtre temporelle ou horizon temporel de stockage. Une fenêtre temporelle de stockage $\Delta_s = [\delta_{smin}, \delta_{smax}]$ définit une règle au niveau de l'environnement temporel. Cette règle limite la portée du stockage des localisations temporelles rattachées à des slots temporels compris entre $t - \delta_{smin}$ et $t - \delta_{smax}$, t étant le slot temporel courant. Elle permet de définir une dynamique de l'environnement temporel qui au niveau spatial équivaut par exemple aux règles d'évaporation des phéromones dans le temps.

Ainsi, si à t un agent décide de créer ou de modifier une localisation temporelle, la réaction de l'environnement temporel aboutira à la création, la modification ou non de cette localisation temporelle, en fonction des lois de l'environnement et de la fenêtre temporelle de stockage. Par exemple : si un agent décide de créer une localisation temporelle relative à une activation dans le passé, la localisation temporelle ne sera pas créée. En effet, les lois de l'environnement temporel interdisent la création d'une localisation temporelle passée ou la modification de **certains** paramètres d'une localisation temporelle passée. Il en est de même pour la création ou la modification de localisations temporelles qui se situent au-delà de la borne supérieure de la fenêtre temporelle de stockage dans le futur.

L'agent n'a également pas la possibilité de modifier les paramètres obligatoires (id, d, f, p, v) relatifs à une activation temporelle passée qui se traduit par une localisation temporelle dans le passé au niveau de l'environnement temporel. Cela revient en effet à modifier le cours du temps dans le passé, ce qui est interdit. Cependant, comme évoqué dans 3.3.2, il est tout à fait possible d'y renseigner des paramètres subjectifs. Ce sont des paramètres qui n'ont pas d'incidence sur l'activation temporelle ni le cours du temps. Exemple : les systèmes de notation ou de retour d'expériences par rapport à l'exécution de l'action qu'un agent a effectuée et qui est relative à la localisation temporelle passée. De manière concrète, un agent ne pourra pas modifier la date de début d'une localisation temporelle passée relative à l'activation de son comportement "faire les courses" qu'il a déjà effectué. Par contre, il pourra attribuer une note à cette localisation temporelle pour renseigner si l'activité "faire les courses" s'est bien passée ou non. La notation se fait obligatoirement après l'exécution du comportement (présent) sur une localisation temporelle passée. Cette fonctionnalité nous est d'une utilité particulière dans le cadre de notre deuxième contribution qui est la mise en place d'un raisonnement anticipatif. Nous y reviendrons plus en détail dans le chapitre 4.

3.3.3.3 État de l'environnement temporel

À un instant t (horloge de la simulation), l'environnement temporel est composé :

— d'un axe temporel ponctué par des slots temporels positionnés sur le passé, sur le présent, sur le futur et compris entre Δ_s.

— d'un ensemble de localisations temporelles rattachées aux slots temporels.

Soit un agent a tel que $L_a(t)$ est l'ensemble des localisations temporelles définies par l'agent a à un instant t (slot) :

$$L_a(t) = \bigcup_{x=1}^{n} l_x \tag{3.12}$$

tel que l_x est rattaché à un slot temporel compris dans la fenêtre de stockage $\Delta_s(t)$, n le nombre de localisations temporelles.

Ainsi, *l'état (statique) de l'environnement temporel* à un instant t est défini par :

$$\sigma(t) = \bigcup_{y=1}^{m} L_{a_y}(t) \tag{3.13}$$

où m est le nombre d'agents du système. $L_{a_y}(t)$ est l'ensemble des localisations temporelles définies par l'agent a_y à t.

La figure 3.5 illustre l'état $\sigma(t)$ de l'environnement temporel à chaque instant du temps simulé. Nous distinguons notamment :

— l'axe de temps de la simulation qui correspond à l'horloge globale de la simulation et qui est gérée par l'ordonnanceur,

— l'axe de temps prévisionnel qui est l'axe de temps stocké au niveau de l'environnement temporel et qui traduit la dynamique prévisionnelle d'activation des agents. Cet axe de temps prévisionnel contient des localisations temporelles passées, présentes et futures.

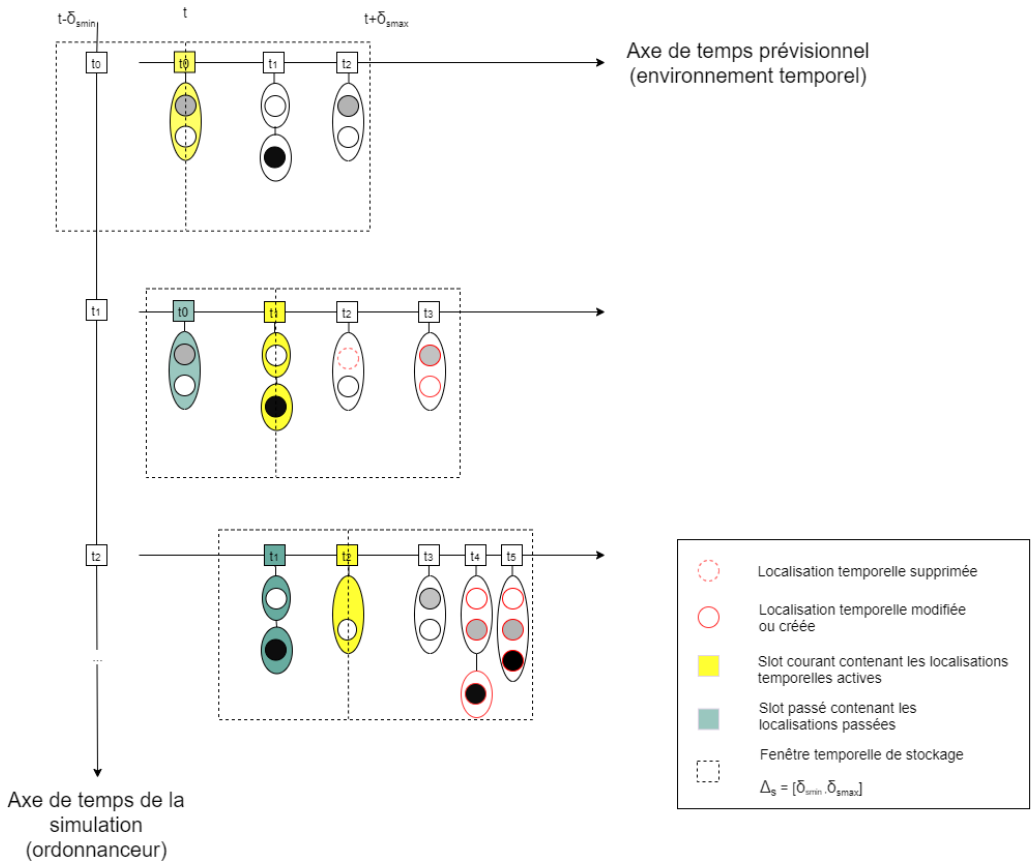


FIGURE 3.5 – Représentation de l'état de l'environnement temporel à chaque instant de l'axe de temps simulé.

3.3.3.4 Exemple

Nous illustrons l'application des concepts précédents à l'aide d'un exemple décrit par le scénario suivant :

1. Soient deux agents a_1 et a_2. Ces agents ont déjà chacun défini leur dynamique d'activation temporelle L_{a_1} et L_{a_2} telle que à $t = 0$, $L_{a_1}(0) = \{l_1, l_2, ..., l_m\}$ et $L_{a_2}(0) = \{l'_1, l'_2, ..., l'_n\}$
2. Phase d'influence : l'agent a_1 décide de créer la localisation temporelle l_{m+1} comprise dans la fenêtre temporelle de stockage Δ_s et l'agent a_2 décide de modifier la valeur v de la variabilité de sa règle temporelle l'_2 dont la date de déclenchement est antérieure à l'instant actuel. Il veut également définir une localisation temporelle future l'_{n+1} située en dehors de la fenêtre temporelle de stockage.
3. Phase de réaction

Ce scénario peut être représenté de la manière suivante :

1. $< \sigma(0) = L_{a_1}(0), T_{a_2}(0), \gamma(0) = \{\} >$
2. $\gamma'(0) = L_{a_1}.creer(l_{m+1}), L_{a_2}.modifierVariabilite(l'_2, v)$
3. $< \sigma(1) = L'_{a_1}(0), L'_{a_2}(0), \gamma(1) = \{\} >$ où $L'_{a_1}(0) = \{l_1, l_2, ..., l_m, l_{m+1}\}$ et $L'_{a_2}(0) = L'_{a_2}(0)$ n'a pas pu être modifié, car les lois de l'environnement ne le permettent pas. En effet, il est impossible de modifier la variabilité d'une temporalité dont la date de déclenchement est antérieure à l'instant actuel. Il est également impossible de créer une localisation temporelle rattachée à un slot situé en dehors de la fenêtre temporelle de stockage.

3.3.3.5 Règles d'accessibilité ou règles de visibilité

Actuellement, la perception de l'agent est contrainte :

— Au niveau de l'agent par :
  — la fenêtre de perception Δ_p qui définit un champ de vision temporel de l'agent ;
  — un paramètre a (autorisation) qui s'ajoute aux cinq paramètres id, d, f, p, v d'une localisation temporelle et qui permet à l'agent de gérer la règle d'accessibilité de chaque localisation temporelle qu'il définit. Ce même paramètre se retrouve également au niveau de l'environnement spatial et l'environnement social comme nous l'avons décrit dans 3.1.1.
— Au niveau de l'environnement, par la fenêtre de stockage Δ_s qui limite la durée du stockage des informations au niveau de l'environnement temporel.

Nous tenons à noter que par respect de la propriété d'autonomie d'un agent, une localisation temporelle est modifiable uniquement par son propriétaire. Ainsi, lorsque nous parlons d'accessibilité :

— pour l'agent ayant défini la localisation temporelle, il s'agit d'un accès en lecture/écriture,
— pour les autres agents, il s'agit d'un accès en lecture seule (visibilité).

Dans ce cadre, il est tout à fait envisageable de définir des règles d'accessibilité et de visibilité plus poussées. Un exemple est la mise en place d'un système de réseau social, en reliant l'environnement temporel à l'environnement social par exemple. Cela n'est pas l'objet de cette thèse, mais reste tout de même une piste intéressante sur laquelle nous travaillons et que nous continuerons à explorer dans les perspectives de nos travaux de recherches.

3.4 Modèle de référence pour un environnement temporel

Le modèle de référence pour l'environnement temporel schématisé par la figure 3.6 est repris de celui proposé par Weyns et al. [96]. Ce modèle contient les mêmes blocs que le modèle original. Cependant, il diffère de ce dernier par le contenu de chaque bloc. En effet, dans le modèle de référence que nous proposons, le contenu de chaque module est spécifique à l'environnement temporel.

3.4.1 Module état

Le module état est de type espace de stockage. Il permet de représenter l'état actuel de l'environnement. Il expose un certain nombre d'interfaces qui permettent à d'autres modules de lire et de modifier l'état de l'environnement. Dans le cas de l'environnement temporel, ce module contient l'axe temporel ponctué par les slots auxquels sont rattachées les localisations temporelles que nous avons décrites dans 3.3.3.3.

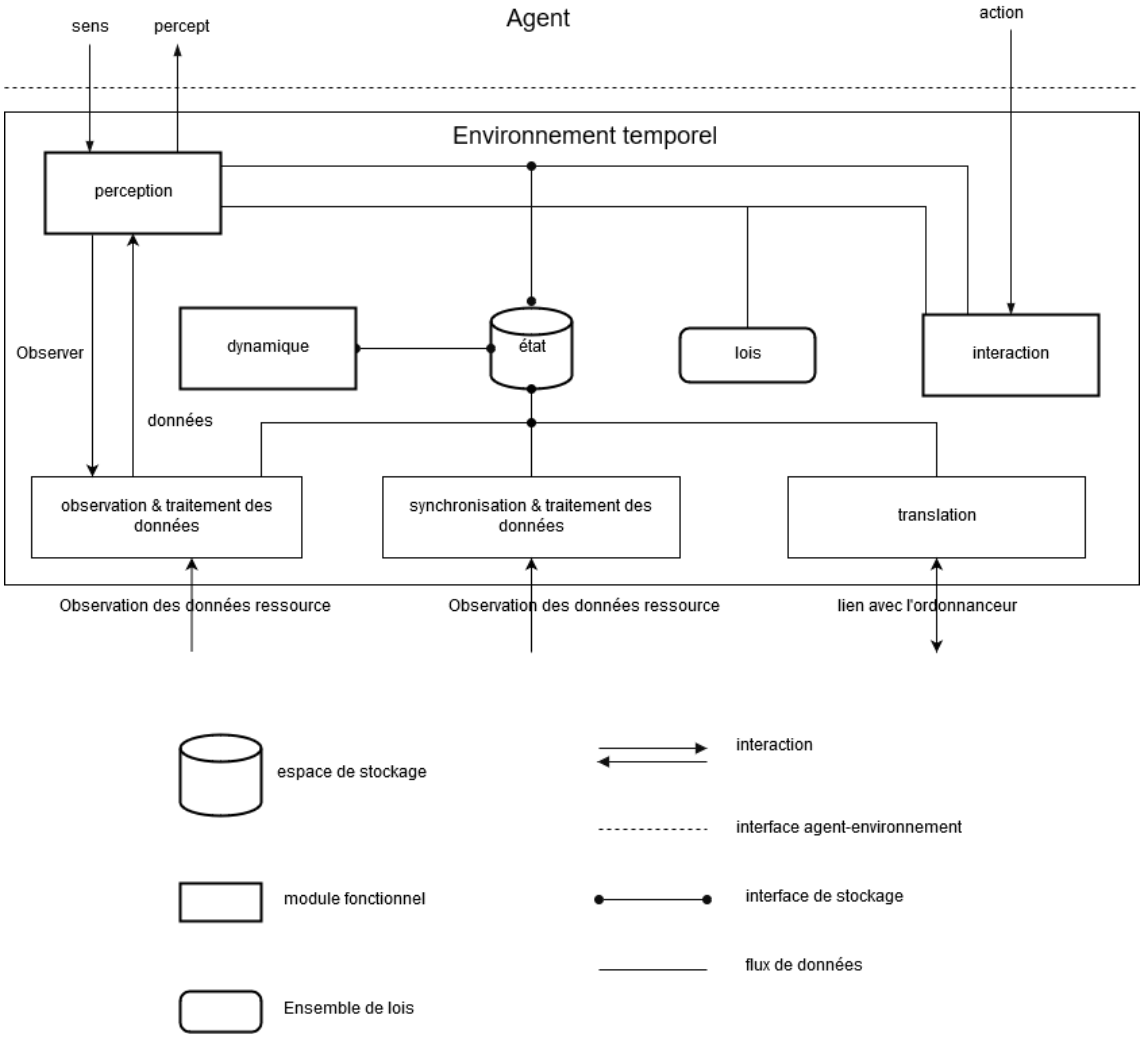


FIGURE 3.6 – Modèle conceptuel de référence d'un environnement temporel repris du modèle de Weyns et al. [96]

3.4.2 Module Lois

Cette composante permet de prendre en compte les contraintes spécifiques à l'application qui sont relatives aux interactions des agents. Quelques exemples de lois qui peuvent être modélisées :

- La définition de la fenêtre temporelle de stockage Δ_s.
- La définition d'une granularité temporelle (pas de temps minimum) dt_{min} comme celle définie dans le modèle à temporalité. Cette granularité est utilisée dans le calcul du rattachement d'une localisation temporelle à un slot temporel. Elle permet également d'affecter un pas de temps par défaut afin de rythmer l'activation des agents qui ne définissent aucune localisation temporelle.
- La loi interdisant la modification des paramètres id, d, f, p, v d'une règle temporelle passée.

3.4.3 Module perception

Ce module fournit les fonctionnalités permettant aux agents de percevoir leur environnement. Le module de perception, déclenché lorsqu'un agent souhaite récolter des informations au niveau de son environnement, génère un ensemble de percepts en fonction de l'état courant de l'environnement d'application, des influences et des données accessibles depuis le contexte de déploiement. La perception de l'agent est soumise aux lois de perception. Ce module est contraint par le module lois.

3.4.4 Module dynamique

Ce module gère les processus propres à l'environnement qui en font une entité active du système. Dans notre cas, il gère les mécaniques de l'environnement temporel permettant le stockage des localisations temporelles, leur rattachement à des slots, leur regroupement en tempos, etc.

Comme dans le modèle à temporalité, quand une localisation temporelle est créée, ou modifiée, sa prochaine date de déclenchement est recalculée. La valeur obtenue permet de positionner la localisation temporelle au niveau d'un slot temporel. Quitte à en créer un s'il n'en existe pas et s'il n'y a pas de contradiction avec le pas de temps minimum spécifié par l'expérimentateur dans le module lois. Une fois le slot temporel de destination trouvé, on insère la localisation temporelle dans le tempo de période compatible ou en en créant un si nécessaire. Notons que pour chaque insertion, le module dynamique exploite le paramètre de variabilité des localisations temporelles pour optimiser l'organisation des slots temporels/tempos. Par exemple : si aucun slot temporel ne se trouve sur la date calculée pour l'insertion de la localisation temporelle, mais qu'il existe, dans un intervalle inférieur à sa variabilité, une date portant un slot temporel, alors la nouvelle localisation temporelle sera insérée dans ce slot déjà existant.

3.4.5 Module interaction

Le module d'interaction gère les actions des agents sur l'environnement. Ce module doit intégrer le modèle d'interaction qui doit être exécuté. Dans notre cas, il s'agit du modèle IRM4S [32].

3.4.6 Module observation et traitement des données

Ce module définit les fonctionnalités permettant d'observer le contexte de déploiement. Les données récupérées par l'observation des éléments du contexte de déploiement sont envoyées au module de perception après un éventuel traitement. Dans le cadre de l'environnement temporel, ce module permet par exemple l'observation des corrections d'agenda lors de la remise en cause des décisions par l'agent.

3.4.7 Module synchronisation et traitement des données

Ce module maintient une représentation du contexte de déploiement dans le module état. Il fournit également des fonctions additionnelles pour adapter les capteurs au problème. Dans notre utilisation de l'environnement temporel, ce module est vide.

3.4.8 Module translation

Ce module fait le lien et gère la synchronisation entre l'environnement temporel et l'ordonnanceur.

3.5 Conclusion

3.5.1 Synthèse

Notre première contribution que nous avons décrite dans ce chapitre est un modèle de représentation du temps. L'objectif consiste en une prise en compte de la dimension temporelle comme un milieu d'interaction, au même titre que l'environnement spatial et l'environnement social. Nous définissons alors un nouveau type d'environnement qui est l'environnement temporel. Cet environnement intègre :

- un support et une mécanique de gestion des informations temporelles basés sur l'approche de type modèle à temporalité ;
- un modèle d'interaction de type influence/réaction (IRM4S).

L'introduction de cet environnement temporel apporte une nouvelle vision concernant la gestion des différents environnements par l'ordonnanceur de la simulation, vis-à-vis de la dimension temps. Elle induit un changement au niveau du cycle d'activation classique des simulations multi-agent.

En effet, la mise en place de l'environnement temporel nous fait prendre conscience de l'existence d'au moins deux types d'environnements dans les simulations multi-agent :

— les environnements contraints par le temps ;
— les environnements non contraints par le temps.

Par conséquent, si avant, le cycle d'activité classique d'un simulateur utilisant un type d'ordonnancement à temps discret était rythmé par deux phases : une phase d'activation de l'environnement et une phase d'activation des agents. Désormais, l'introduction de la notion d'environnement temporel nous amène à diviser la phase d'activation de l'environnement en deux. Désormais, les trois (3) phases du cycle d'activation de la simulation sont :

1. La phase d'activation des environnements contraints par le temps sur l'intervalle de temps $]t(i-1), t]$;
2. La phase d'activation des agents sur le slot temporel courant t ;
3. La phase d'activation des environnements non contraints par le temps sur la considération que t vient d'être passé $]t, t(i+1)]$;

L'intégration de cet environnement temporel dans le système repose sur le modèle générique d'organisation AGR et sa variante AGRE. Nous l'appelons AGRET ou Agent-Groupe-Rôle-Environnement-Temps, car il vient compléter l'aspect situé dans le temps à l'aspect situé socialement et spatialement de AGRE. Ainsi, dans AGRET, un agent dispose d'une temporalité dans son environnement temporel, tout comme il dispose d'un corps dans l'environnement spatial ou d'un rôle dans l'environnement social. Cette temporalité intègre un ensemble de capteurs et d'actionneurs lui permettant d'interagir avec l'environnement temporel selon un modèle d'interaction de type Influence/Réaction (IRM4S). Au niveau agent, cette interaction se présente sous forme de définition, de modification ou de perception de localisations temporelles. Une localisation temporelle traduit un besoin d'activation de l'agent à un instant t. La perception de l'agent est contrainte par un horizon temporel de perception, défini arbitrairement par le modélisateur. Cet horizon peut être propre à chaque agent ou à chaque type d'agent. Il peut être également commun à tous les agents ou à tous les types d'agents. Le stockage de ces localisations temporelles au niveau de l'environnement temporel est contraint par un horizon temporel de stockage. Cet horizon est également défini arbitrairement par le concepteur du modèle.

Le traitement des localisations temporelles quant à lui ainsi que les mécaniques permettant la dynamique propre à l'environnement temporel sont inspirés des mécaniques du modèle à temporalité.

La gestion de la dimension temporelle se trouve donc divisée en deux :

— Le stockage et le traitement des localisations temporelles se font au niveau de l'environnement temporel, au niveau du modèle de simulation. Plus particulièrement, le stockage et la perception des localisations temporelles passées, présentes et futures sont de nouvelles fonctionnalités qui n'existent pas dans le cadre de l'approche de type modèle à temporalité, mais que nous avons introduites dans notre proposition ;
— L'ordonnancement de la simulation se fait au niveau de l'ordonnanceur, au niveau de la plateforme de simulation.

Ces deux entités : environnement temporel et ordonnanceur sont complémentaires. Cependant, le fonctionnement de l'environnement temporel est indépendant de l'approche d'ordonnancement du temps utilisé au niveau de l'ordonnanceur de la simulation. Par conséquent, il est tout à fait envisageable de mettre en place un environnement temporel dans un modèle de simulation utilisant une autre approche d'ordonnancement.

Les principes de bases du modèle AGRET, inspirés de ceux de AGRE se résument alors comme suit :

Principe 3.5.1. Un monde multi-agent est constitué d'agents (individus) qui peuvent percevoir et agir dans des espaces et manifester leur existence à travers leurs modes.

Une organisation est un type de monde dans lequel les espaces sont des groupes et dans lequel les agents perçoivent et agissent à travers leurs rôles.

Un monde physique est un type de monde dans lequel les espaces sont des zones ou aires et dans lesquelles les agents perçoivent et agissent à travers leurs corps.

Un monde temporel est un type de monde dans lequel les espaces sont des agendas et dans lequel les agents perçoivent et agissent à travers leurs temporalités.

Principe 3.5.2. Un agent peut appartenir simultanément à un monde social, physique et temporel. Le nombre de rôles qu'il peut jouer n'est pas restreint. Contrairement à cela, le nombre de corps

qu'il peut posséder est contraint par la condition selon laquelle un agent peut agir dans l'espace à travers un seul corps. Il en est de même pour sa représentation temporelle, un agent peut agir dans l'environnement temporel à travers une seule temporalité.

Principe 3.5.3. Un agent peut posséder différents modes de différents types. La contrainte concernant le nombre de modes qu'un agent peut posséder dépend du monde dans lequel il est inscrit.

Un agent peut jouer plusieurs rôles dans un groupe et peut faire partie de plusieurs groupes.

Un agent peut posséder un seul corps dans un monde physique. Ce principe exprime le fait qu'un agent ne peut exister dans deux places différentes en même temps. Il en est de même pour sa temporalité.

Notons $es : m.op(a1, .., an)$ l'action d'un agent avec le mode m exécutant l'opérateur op avec les arguments $a1, .., an$ dans l'espace es. Notons alors les assertions suivantes :

— $mode(x, m, s)$: l'agent x a un mode m dans l'espace s

— $type(m, M)$: le mode m est de type M

— $op(o(a1, ..an), M)$: l'opérateur $o(a1, ..an)$ est défini dans le mode type M.

Principe 3.5.4. Le mode est le moyen par lequel un agent agit dans un espace. Un agent peut agir dans un espace si un de ses modes dans l'espace lui donne la possibilité de le faire. Ainsi, un agent x peut effectuer une action act dans l'espace es, s'il existe un mode m de x dans es tel que le type de m permet act.

$$s : m.o(a1, .., an) \Rightarrow \exists x : Agent, mode(x, m, s) \wedge type(x, M) \wedge op(o(a1, ..an), M) \tag{3.14}$$

3.5.2 Limites et perspectives

Le modèle AGRET permet la prise en compte au même titre des trois dimensions : espace physique, temps et organisation dans les simulations multi-agent. Ces trois dimensions correspondent aux trois (3) dimensions que doit intégrer simultanément un système territorial [83]. Dans notre cas il s'agit de la ville intelligente et plus précisément de la mobilité intelligente. Nous nous intéressons particulièrement à la gestion de ressources partagées dans l'espace physique et le temps. L'originalité de notre proposition repose sur la considération du temps comme un nouveau milieu d'interaction : un environnement. Plus particulièrement, dans le domaine de la mobilité, le support d'interaction mis en place, basé sur le modèle AGRET, sert par exemple pour l'échange d'informations dans le cadre de l'optimisation de la gestion des places de parking, de la gestion des rechargements de véhicules avec des bornes publiques, ou de la gestion des véhicules libres-services (vélos, trotinettes, etc.). Ces échanges d'informations spatiales, sociales et temporelles sont nécessaires à la mise en place de la dimension participative/collective dans le système de ville. Cette dimension est indispensable et constitue la base même du concept de ville intelligente : permettre aux citoyens de participer à la construction, la planification et la gestion de la ville et des services numériques qui s'y rapportent. Dans ce chapitre, cette participation se fait sous forme d'échange d'informations. Nous allons voir dans le chapitre 4 suivant comment ces informations échangées sont utilisées par les agents, à travers un mécanisme de raisonnement anticipatif, pour optimiser le système au niveau individuel et collectif.

Le modèle d'environnement temporel que nous avons présenté est un modèle de base. Plusieurs aspects peuvent encore être améliorés et plusieurs perspectives sont intéressantes à explorer.

3.5.2.1 Des règles d'accessibilité basées sur une approche de type réseau social

Dans notre approche, nous avons choisi de nous concentrer davantage sur les similitudes et les relations qui peuvent exister entre le temps et l'espace physique. Nous constatons que la troisième dimension, la dimension sociale, est traitée de manière assez marginale. En effet, des études ont déjà été menées au cours de la thèse concernant l'intérêt des approches comme celles utilisées dans les réseaux sociaux (Facebook, LinkedIn) dans l'optimisation des simulations multi-agent. Nous présentons ces débuts de travaux dans la section 7.2.2 du chapitre 7.

De manière générale, la dimension sociale, déjà existante actuellement dans AGRET offre un moyen de partitionner un SMA. Chaque partition constitue un contexte d'interaction pour les agents. Une utilisation consiste à mettre en place des règles de visibilité des informations temporelles qui se basent sur les groupes sociaux et les rôles.

Pour faire l'équivalence au monde réel, si nous prenons l'exemple de Facebook : un agent possédant un compte Facebook y partage un certain nombre d'informations. Il peut définir des règles de confidentialité différentes en fonction de l'espace où il effectue le partage et des règles d'accessibilité.

— Sur son mur ou sur sa page, il peut partager une information publiquement, de manière privée (à lui uniquement), à ses amis ou à certaines personnes uniquement. Il peut également restreindre l'accès de manière à ce que l'information ne soit pas visible par certaines personnes ;
— Dans un groupe, son contrôle est plus restreint, car il est contraint par des règles imposées par le groupe et par les administrateurs ou modérateurs du groupe. Il en est de même pour une publication sur le mur ou sur la page d'une tierce personne. Il doit se soumettre aux différentes règles que le propriétaire du mur ou de la page a définies au préalable.

Cet exemple laisse transparaître des fonctionnalités qui sont déjà en place dans notre système comme la notion de groupe, la notion d'organisation, et la notion de règles d'accessibilité, qui peuvent différer en fonction du type d'agent, du lien entre les agents, de l'appartenance à un groupe, etc.

La piste des réseaux sociaux présente un réel potentiel, car il s'agit d'outils qui sont fortement utilisés pour les échanges d'informations dans les villes. Dans cette optique, d'un point de vue conceptuel, l'architecture déjà en place, au niveau de l'environnement social dans AGRET (monde, espace et mode) constitue une base solide qui devrait permettre de faire évoluer facilement notre approche sans restructuration majeure de l'architecture. D'un point de vue pratique, la structuration de la dimension sociale dans notre plateforme de simulation SimSKUAD et dans notre modèle de simulation SkuadCityModel laisse déjà transparaître quelques notions liées aux réseaux sociaux : avatar, participant, boîte mail, etc. Nous en reparlons dans le chapitre 5.

Pour des raisons d'organisation et de temps, nous avons choisi de nous concentrer principalement sur la représentation du temps et le raisonnement temporel. Néanmoins, l'intégration des approches liées aux réseaux sociaux figure alors parmi les perspectives intéressantes pour la suite de nos travaux.

3.5.2.2 Amélioration des horizons temporels

Il s'agit ici de l'amélioration de l'horizon temporel de perception au niveau de l'agent et de l'horizon temporel de stockage au niveau de l'environnement temporel. Comme nous l'avons mentionné dans nos descriptions, il est très important de choisir une valeur optimale de ces horizons en fonction des objectifs et des particularités de la simulation et des agents. Selon les besoins, il est également possible de définir des horizons temporels "ciblés". Un horizon temporel périodique en est un exemple. La valeur de l'horizon temporel de stockage peut également varier en fonction du type d'application. En effet, la pertinence et l'obsolescence des informations varient en fonction des objectifs. Par exemple : Une simulation de l'évolution du développement d'une ville aurait peut-être besoin de stocker des informations sur plusieurs années. Contrairement à cela, une simulation de l'occupation des routes pourrait n'avoir besoin de stocker que des informations sur quelques heures, quelques semaines ou quelques mois uniquement.

3.5.2.3 Les environnements non contraints par le temps

Nous avons introduit dans ce chapitre la notion d'environnement non contraint par le temps. Dans ce cadre, nous avons donné l'exemple de l'environnement temporel. Nous nous posons donc la question concernant la possibilité d'existence d'autres environnements dont l'état évoluerait hors du temps. Nous pensons par exemple aux méta-environnements d'observation. Ces méta-environnements d'observation traitent les flux d'informations générés par la simulation. À partir de cela, ils génèrent à leur tour des informations utiles qu'ils affichent à l'utilisateur par exemple. L'état de ces méta-environnements semble évoluer hors du cadre imposé par le temps. Des études portant sur ces types de méta-environnements ont été menées auparavant au sein de notre équipe de simulation [21] [74]. Dans ce contexte, la piste des environnements non contraints par le temps pourrait constituer une suite logique de nos travaux.

Chapitre 4

Raisonnement anticipatif dans les SMA : Apports du modèle AGRET

Nous avons présenté, dans le chapitre 3 précédent, un support d'interaction qui permet l'échange d'informations spatiales, temporelles, et sociales. Cette approche intègre une nouvelle considération du temps sous forme de milieu d'interaction que nous avons appelé environnement temporel. Dans ce cadre, nous avons proposé une extension du modèle classique d'organisation AGR et de sa variante AGRE. Nous avons nommé cette extension AGRET. Ce chapitre 4 vient compléter cette première proposition par un raisonnement temporel : un raisonnement anticipatif. Il s'agit d'une possibilité qui est permise par la visibilité sur la dimension future du temps qu'apporte notre représentation de la dynamique temporelle sous forme d'environnement. En effet, ce raisonnement anticipatif permet à l'agent d'exploiter les informations passées, présentes et futures planifiées par les agents eux-mêmes. L'objectif est d'optimiser l'emploi du temps, c'est-à-dire le faire converger vers un emploi du temps plus pertinent, par ajustement successif.

Différents problèmes sont associés au raisonnement temporel. Des exemples sont la planification, l'ordonnancement de tâches et l'anticipation. Nous pensons que l'anticipation est un sujet pertinent dans notre contexte pour les raisons suivantes :

— Dans notre cadre applicatif, comme dans la plupart des systèmes de mobilité dans les villes intelligentes, les agents agissent selon un emploi du temps qu'ils définissent eux-mêmes. Cet emploi du temps peut être amené à être révisé à n'importe quel moment de la simulation. Il est donc indispensable d'inclure des capacités d'anticipation dans le processus décisionnel de l'agent. Cela devra lui permettre de prendre en compte des prédictions lors de sa planification ;

— Leur capacité d'anticipation devra permettre aux agents d'optimiser leur comportement en n'effectuant ainsi que les actions véritablement nécessaires [29]. En effet, elle permettrait de réduire l'ensemble des cas possibles et répondrait à la problématique d'explosion combinatoire des cas possibles. Nous avons évoqué cette problématique dans le chapitre 2 de ce manuscrit. Il s'agit d'une problématique commune aux simulations multi-agent pour les villes intelligentes.

— Notre première contribution consiste en la considération du temps comme un environnement et met en valeur le côté plutôt réactif des agents. Un modèle d'agent réactif présente plusieurs avantages, notamment en matière de simplicité. Cependant, afin de modéliser, de façon plus réaliste, l'humain que nous simulons, nous contrebalançons ce côté réactif des agents par un côté cognitif. En effet, l'anticipation occupe une place majeure parmi un éventail de processus généralement considérés comme cognitifs et dans lesquels la maîtrise humaine est indéniable. Par conséquent, exhiber des comportements anticipatifs rendrait l'agent plus crédible [81].

Au final, nos contributions relèvent toutes les deux de la dimension temporelle. En effet, comme mentionné dans 2.2.2.2, l'étude du temps revêt deux aspects : la représentation du temps et le raisonnement temporel. Notre première contribution se situe au niveau de la représentation du temps. Nous avons proposé une représentation du temps sous forme d'environnement, au même titre que l'environnement spatial ou l'environnement social. Dans ce chapitre, nous présentons un raisonnement temporel qui en résulte. Plus particulièrement, nous montrons les avantages que cela apporte

en matière de raisonnement anticipatif. Pour ce faire, nous exploitons, dans notre approche, le flux de perception de l'ensemble des environnements dans AGRET et plus particulièrement, la perception des informations sur la dynamique temporelle des agents. Cette perception est permise par ce nouvel environnement de nature temporel. Ainsi, en plus des informations passées et présentes qui sont déjà prises en compte dans la plupart des approches classiques d'anticipation, l'environnement temporel offre une nouvelle visibilité sur la dimension future du temps (futur planifié par les agents), que nos agents prennent aussi en compte dans leur raisonnement anticipatif. Cette prise en compte de la dimension future du temps dans le raisonnement anticipatif constitue la principale originalité de notre proposition.

Nous présentons, dans 4.1, le modèle sur lequel nous basons notre proposition. Nous détaillons, par la suite, notre deuxième contribution dans la section 4.2 . Nous y présentons le fonctionnement des différents modèles qui composent nos agents. Plus particulièrement, nous montrons le fonctionnement de notre modèle de perception (4.2.2), notre modèle prédictif (4.2.3) et de notre modèle de décision (4.2.4). Enfin, comme à chaque fin de chapitre, nous terminons par une synthèse et par l'énoncé des limites donnant suite à des perspectives pour nos travaux futurs (4.3).

4.1 Modèle de base

Definition 4.1.1. Anticipation. Il existe plusieurs définitions de l'anticipation. Nous en citons quelques-unes dans le chapitre 2. La définition qui se rapproche le plus de la nôtre est celle donnée par Butz et al. [13]. Ces auteurs définissent l'anticipation comme un processus, ou un comportement, qui ne dépend pas uniquement du passé et du présent, mais aussi des prédictions, des attentes, ou des croyances concernant le futur. Une prédiction est une représentation d'un événement futur particulier [72]. Dans notre cas, comme dans la réalité, certains événements futurs peuvent être prévus et définis à l'avance par les agents. Par exemple : de manière générale, une personne a déjà plus ou moins connaissance de ses projets. Si elle travaille, elle a déjà plus ou moins connaissance d'au moins une partie de son emploi du temps de la semaine. Elle peut déjà prévoir, à peu près, à quelle heure elle est censée être sur son lieu de travail et à quelle heure elle devrait quitter son travail. Dans notre approche, chaque agent partage ses projets, sous forme de localisations temporelles, au niveau de l'environnement temporel.

Definition 4.1.2. Localisation temporelle. Nous rappelons qu'une localisation temporelle

$$l = \{id, d, f, p, v, ...\} \tag{4.1}$$

que nous avons défini initialement dans 3.3.2 correspond à la position de l'agent dans l'environnement temporel. Cette position est décrite par un ensemble des paramètres dont id, d, f, p, v sont les paramètres obligatoires qui au niveau de l'ordonnanceur traduisent un besoin d'activation de l'agent à un slot de temps t (cf définition dans 3.2). À ces paramètres peuvent se rajouter d'autres paramètres optionnels, définis en fonction du modèle et des objectifs de la simulation.

Chaque localisation temporelle traduit une volonté ou un souhait de l'agent d'être activé à un instant t pour effectuer une action. La localisation temporelle, au niveau de l'environnement temporel, est l'équivalent de la localisation spatiale au niveau de l'environnement spatial. L'ensemble des localisations temporelles permet, en quelque sorte, de déduire l'**emploi du temps** de l'agent. Dans le raisonnement anticipatif que nous mettons en place au sein des agents du système, nous exploitons les informations que l'agent récolte sous forme de percepts au niveau des différents environnements du système afin de lui permettre de remettre en question son emploi du temps. Notre objectif principal est de faire converger ce dernier, par ajustements successifs, vers un emploi du temps plus pertinent. Le raisonnement anticipatif consiste donc à :

- évaluer ou réévaluer la possibilité d'exécution d'une activité ;
- choisir l'activité la plus pertinente à exécuter ;
- enrichir et apporter plus de précision à l'emploi du temps de l'agent.

L'étude des classes d'anticipations et de quelques exemples de travaux ([22] [25] [81]) portant sur l'intégration d'une capacité d'anticipation au sein des agents fait ressortir quelques similarités en termes d'approche. De manière générale, l'architecture d'un agent doté d'une capacité d'anticipation intègre au minimum deux modèles : un modèle prédictif et un modèle de décision.

4.1.1 Modèle prédictif

Certaines approches d'anticipation dans les SMA[25] [81] reposent sur la capacité des agents à réaliser des prédictions. Grâce à ces prédictions, l'agent peut s'attendre à la réalisation d'événements futurs, et agir au préalable. Ces prédictions peuvent être internes ou externes.

Une prédiction interne correspond à une prévision de l'évolution d'un état interne de l'agent tandis qu'une prédiction externe concerne un futur état du monde. Selon Reynaud, une prédiction ne peut se construire qu'à partir d'un ensemble de modèles appelé modèle prédictif. Ce modèle prédictif intègre obligatoirement un modèle de l'environnement et un modèle des actions. L'ensemble est appelé modèle du monde [22]. Ce modèle fournit, en sortie, des prédictions d'un état ou d'un événement futur. Ces prédictions sont primordiales, car elles permettent d'éviter des comportements aberrants. Par exemple : un automobiliste qui refuse de recharger son véhicule électrique tant que la batterie n'est pas encore totalement à plat (ne pas prendre en compte la consommation électrique nécessaire pour aller faire recharger son véhicule). En tant qu'humains, nous effectuons ces prédictions de manière très naturelle, sans en avoir réellement conscience. Cependant, pour qu'un agent puisse anticiper, même des événements paraissant évidents, il est indispensable que ces événements soient mentionnés dans le modèle de l'environnement [81]. Dans les approches classiques, ce modèle est capable d'exprimer, par exemple :

— des probabilités d'apparition d'événements conditionnées par l'heure de la simulation ou par le déclenchement d'autres événements ;
— des faits généraux sur le monde (lieux, horaires d'ouverture, conditions météo, etc.) ;
— Des approximations de temps de trajet.

Dans certaines approches comme celle de Doniec et al. [25] ou de Reynaud [81], le fonctionnement d'un tel modèle consiste à simuler en accéléré la représentation de l'environnement au sein même de l'agent et ainsi fournir des prédictions.

L'approche que nous proposons est différente de cela dans le sens où nous construisons différemment nos prédictions externes, c'est-à-dire l'état futur du monde. Dans notre modèle de l'environnement, nous exploitons la perception de la dimension future du temps qui résulte des informations que les agents partagent sous forme de localisations temporelles au niveau de l'environnement temporel. Une partie de l'état prévisionnel du monde est donc obtenu par perception de l'axe temporel contenu dans l'environnement temporel. Comme expliqué dans le chapitre 3, cet axe est alimenté et mis à jour par les agents eux-mêmes. Cela vient enrichir ou alléger les microsimulations effectuées par les agents en interne. Dans certains cas, notre approche nous permet de nous affranchir des approches classiques qui réalisent, au sein même de chaque agent, des microsimulations pour générer des prédictions. Cette prise en compte de la dimension future planifiée, en plus de la dimension passée et la dimension présente, dans le raisonnement anticipatif est la principale originalité de notre approche.

Dans notre proposition, la perception de cet état prévisionnel de l'environnement temporel intervient sur plusieurs niveaux dans le mécanisme de raisonnement anticipatif de l'agent :

— En début de cycle, l'agent perçoit son contexte d'activation spatial, social et temporel.

— En milieu de cycle, avant l'arbitrage des décisions : Les informations perçues par l'agent au niveau des différents environnements du modèle AGRET (temporel, spatial et social) jouent plusieurs rôles :
  — situer les actions dans les environnements. Exemple : aller se divertir à 12 h au parc localisé à (x, y) ;
  — vérifier que les préconditions nécessaires à l'exécution de chaque action sont remplies. Exemple : Est-ce que le véhicule électrique dispose d'une charge de batterie suffisante pour effectuer le trajet domicile-travail ? La vérification de ces préconditions permettent non seulement l'enchaînement des actions, mais également de dresser une liste de déclinaisons possible de cet enchaînement d'actions. En fin de cycle, l'agent devra choisir la déclinaison qui lui semble la plus avantageuse en termes de coûts ;
  — renseigner un ensemble d'indicateurs ou de coûts nécessaires à l'arbitrage des décisions. Exemple : Aller se recharger sur une borne b_1 se trouvant à une distance d_1 pourrait engendrer une dépense d'énergie e_1. La longueur estimée de la file d'attente (queue) au niveau de b_1 est q_1. Dans cet exemple, les coûts sont calculés à partir de e_1, d_1 et q_1.

4.1.2 Modèle de décision

Dans un modèle d'agent classique intégrant un raisonnement anticipatif, le modèle de décision propose un comportement ou une action à partir des prédictions générées par le modèle prédictif. Plusieurs approches peuvent être utilisées à ce niveau. Doniec et al. [25] utilisent par exemple un modèle réactif tandis que Reynaud [81] part sur un modèle plus complexe, basé sur la composition de comportement et la comparaison des instanciations de plans. Un plan est un enchaînement d'actions que l'agent prévoie d'effectuer.

Une approche réactive comme celle utilisée par Doniec et al. se compose d'une couche prédictive et d'une couche réactive. La couche prédictive utilise un modèle de l'environnement ainsi qu'un anticipateur qui permettent de simuler plus rapidement l'environnement et générer des prédictions (couche prédictive). Lorsque l'anticipateur détecte un état indésirable, la couche d'anticipation agit sur la couche de décision (couche réactive) pour modifier le comportement de l'agent.

Contrairement à cette démarche réactive, Reynaud utilise le modèle décisionnel véritablement utilisé comme simulateur de lui-même. En d'autres termes, lorsque l'agent réalise des prédictions, il se sert du modèle décisionnel véritablement utilisé, et l'applique à une situation virtuelle anticipée. La qualité des prédictions est donc excellente, au prix d'un coût en ressources de calcul élevé, puisque les calculs ne sont pas simplifiés.

D'un point de vue conceptuel, notre démarche se rapproche plus de celle de Reynaud dans le sens où nous effectuons une comparaison des instanciations de plan. Cependant, d'un point de vue technique, nous utilisons un module décisionnel moins complexe. Notre module décisionnel exploite les informations collectées à travers les perceptions spatiale, temporelle et sociale traduites sous forme de coûts par notre modèle prédictif pour l'arbitrage des décisions. Son fonctionnement consiste en une comparaison des instanciations de plans. Pour cela il applique une pondération sur chaque coût correspondant. Le choix repose sur la minimisation des coûts pondérés. Les valeurs des pondérations sont arbitraires. Elles sont définies par le concepteur du modèle ou par l'utilisateur en fonction des objectifs de la simulation. Une fois l'instance de plan de coût pondéré minimale choisit, chaque action le composant est envoyée aux actionneurs pour être transformée en influences. Les actions qui doivent être immédiatement exécutées sont envoyées aux actionneurs correspondants. Les actions qui doivent être exécutées plus tard quant à elles sont envoyées aux actionneurs de l'environnement temporel afin qu'elles puissent être transformées en localisations temporelles. De cette manière, l'agent obtient un emploi du temps plus riche et plus précis. Dans ce cadre, différents paramètres facultatifs peuvent être rajoutés aux localisations temporelles. Ces paramètres sont par exemple des références vers les autres environnements du système comme l'identifiant de la borne de recharge pour une action "se recharger", l'identifiant d'un espace de loisir, etc. Nous expliquons cela plus en détail dans la section 4.2.

4.1.3 Modèle d'agent de base

Afin de clarifier les concepts abordés dans la suite de ce chapitre ainsi que le formalisme, voici un résumé du vocabulaire utilisé.

Definition 4.1.3. Action élémentaire. Une action élémentaire correspond à un opérateur relatif à une activité. Une action élémentaire est abstraite. Par exemple : se recharger, aller travailler, etc.

Definition 4.1.4. Action concrète. Une action concrète est l'instanciation d'une action élémentaire dans l'environnement. En d'autres termes, il s'agit de la prise en compte, en pratique, de la réalité environnementale concrète dans une action abstraite. Une action concrète est située. Par exemple : Soient l'action élémentaire *ope* ="se recharger" et la borne b qui est située dans l'environnement spatial et qui est disponible à t. L'action concrète *opc* associée à *ope* et b est *opc* = "se recharger sur la borne b à t".

Definition 4.1.5. Plan. Nous appelons plan, noté OPc, un enchaînement d'actions concrètes.

$$OPc = \{opc_1, opc_2, ..., opc_n\}$$

Dans toutes nos notations, nous utilisons des caractères en **MAJUSCULES** pour désigner un **ENSEMBLE** et des caractères en **minuscules** pour désigner un **élément**. Par exemple le plan OPc est un ensemble d'actions concrètes tandis que *opc* est une action concrète.

Nous nous basons sur un modèle comportemental d'un agent réactif comme le montre la figure 4.1. Le cycle comportemental classique de l'agent consiste en une phase de perception suivie d'une

phase de réaction. Ce choix illustre ce que nous avons évoqué en début de ce chapitre. En effet, l'anticipation est considérée comme une capacité cognitive. Elle vient en complément au côté réactif des agents. L'objectif est d'exhiber des comportements anticipatifs afin d'être plus fidèle à l'humain qui est simulé. Néanmoins, l'approche reste tout de même applicable dans le cas d'un agent cognitif de base.

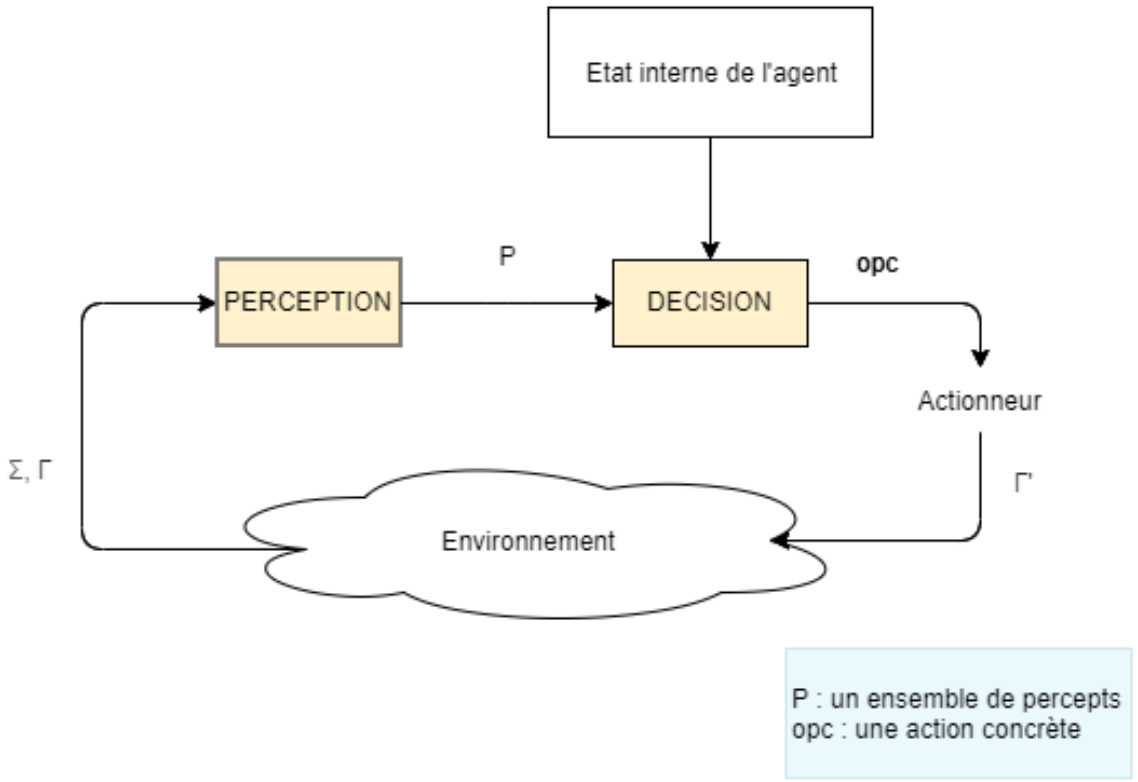


FIGURE 4.1 – Cycle comportemental classique d'un agent.

Comme nous l'avons précisé et expliqué dans la section 3.3.1 dans le chapitre 3, le modèle d'interaction sur lequel nous nous basons est le modèle IRM4S. Pour rappel, ce modèle est une amélioration du modèle influence/réaction. Elle prend en compte, dans son formalisme des caractéristiques propres à la simulation. Dans IRM4S, le cycle comportemental de l'agent aboutit à la production d'influences.

$$Behaviour_a : \Sigma \times \Gamma \mapsto \Gamma \tag{4.2}$$

La première grande différence par rapport au modèle influence/réaction classique est la suivante : dans le modèle influence/réaction, Ferber and Müller modélisent la perception locale et subjective de l'environnement par une influence $\gamma \in \Gamma$. La fonction *Percept* se révèle très contraignante et entraîne une certaine confusion. Elle suppose notamment qu'un agent ne puisse pas percevoir des variables appartenant à Σ [24]. Cette perception des variables environnementales a été rajoutée dans IRM4S.

Notre modèle d'agent de base comporte alors deux modules :

1. Un module perception qui comme son nom l'indique gère la perception et le traitement du contexte. Il fournit à l'agent les fonctionnalités lui permettant de collecter les états externes, appelés également états des environnements. Comme dans IRM4S, son fonctionnement consiste en l'application de la fonction

   $$Perception : \Sigma \times \Gamma \mapsto P \tag{4.3}$$

   où Σ est l'état de l'environnement, Γ est un ensemble d'influences externes (provenant de l'environnement), P est un ensemble de percepts.
2. Le module décision qui détermine l'action concrète à réaliser en fonction de l'ensemble des percepts qu'il reçoit en entrée et de l'état interne de l'agent.

   $$Decision : P \times S \mapsto opc \tag{4.4}$$

où S est l'état interne de l'agent et *opc* est une action concrète.

4.2 Proposition d'un raisonnement anticipatif basé sur AGRET

4.2.1 Exemple

Dans la suite de ce chapitre, nous illustrerons chaque concept sur la base de l'exemple suivant.

Soit un système composé d'un ensemble de n agents conducteurs de véhicules électriques noté $A = \{a_1, a_2, ..., a_n\}$ et d'un ensemble de m agents gestionnaires de bornes de recharge électriques noté $B = \{b_1, b_2, ..., b_m\}$. Pour simplifier la compréhension, nous appelons les agents conducteurs de véhicules électriques "automobilistes" et les agents gestionnaires de bornes de recharge électrique "bornes". Un automobiliste définit, en début de simulation, un ensemble de localisations temporelles relatives à son emploi du temps. L'ensemble de ses localisations temporelles indique les instants auxquels l'agent souhaite effectuer des actions. En fonction de la richesse des informations contenues dans les localisations temporelles, l'agent peut facilement identifier les actions élémentaires ou les actions concrètes correspondantes. Nous définissons également une valeur par défaut de l'horizon temporel de perception Δ_{pt}, propre à chaque type d'agent. Dans notre exemple, nous choisissons de mettre en oeuvre le mécanisme d'anticipation uniquement au niveau des automobilistes. Lorsqu'un automobiliste planifie d'aller se recharger à une borne, il en informe cette dernière. En parallèle à cela, il crée la localisation temporelle correspondante. Lorsque la borne reçoit cette notification en provenance de l'automobiliste, elle crée ou met à jour la localisation temporelle correspondante.

Pour gérer l'accessibilité aux informations contenues dans l'environnement temporel, nous définissons deux règles :

- **Publique** : la localisation temporelle est perceptible par tous les agents du système ;
- **Privée** : seul l'agent à l'origine de la création de la localisation temporelle peut percevoir cette dernière.

Dans notre scénario, les localisations temporelles définies par les automobilistes sont de visibilité privée. Contrairement à cela, les localisations temporelles définies par les bornes sont de visibilité publique.

Dans notre exemple, deux types de localisations temporelles différentes sont alors créées :

- Une localisation temporelle appartenant à l'automobiliste qui indique qu'il prévoit d'aller se recharger à la borne à un instant. Cette localisation temporelle est propre à l'automobiliste. Elle est de visibilité privée ;
- Une localisation temporelle appartenant à la borne lui indiquant qu'elle a prévu de recharger un véhicule à cet instant. Cette localisation temporelle est de visibilité publique. C'est-à-dire que tous les autres agents du système peuvent la percevoir.

Le partage de ces localisations temporelles au niveau de l'environnement temporel permet également à l'ordonnanceur d'avoir accès aux informations nécessaires pour l'activation des agents du système. Les temporalités correspondantes sont alors créées au niveau de l'ordonnanceur. Ces temporalités indiquent à l'ordonnanceur que la borne et l'automobiliste souhaitent être respectivement activés aux instants correspondant aux dates de déclenchement respectives de chaque localisation temporelle. Deux situations peuvent avoir lieu au niveau de la borne :

- Si aucun agent n'a encore signalé à la borne une intention de venir se recharger à l'instant t : une nouvelle localisation temporelle sera créée. Cette localisation temporelle comporte un paramètre facultatif q (queue en anglais) qui stocke la valeur de la longueur prévisionnelle de la file d'attente. Ce paramètre est mis à jour lorsqu'une intention de se recharger au même instant est signalée à la borne. Il est également mis à jour lorsqu'un automobiliste annule son intention de se recharger et le signale à la borne.

- Dans le cas où un ou plusieurs agents a déjà signalé à la borne son intention de venir se recharger à l'instant t, la localisation temporelle correspondante existe déjà. La borne procède alors à une mise à jour de celle-ci en incrémentant la valeur du paramètre q. Cette mise à jour de la valeur la longueur prévisionnelle de la file d'attente correspondante à la localisation temporelle, au niveau de l'environnement temporel, n'a aucune incidence sur la temporalité correspondante au niveau de l'ordonnanceur. En effet, les seuls changements pouvant avoir des répercussions au niveau de l'ordonnancement de la simulation sont ceux qui affectent la dynamique d'activation temporelle des agents. Il s'agit notamment de la

suppression d'une localisation temporelle ou de la modification d'un ou de plusieurs des paramètres obligatoires d, f, p, v.
Au niveau de l'environnement temporel, en fonction de la capacité de la borne, la mise à jour de la valeur de la file d'attente q peut affecter la longueur des files d'attente au niveau des localisations temporelles rattachées aux slots temporels suivants. Par exemple : Supposons qu'à un slot t_0, une borne ait défini une localisation temporelle l_1 dont la valeur de $q = 3$. La capacité de la borne lui permet de recharger uniquement un véhicule toutes les 30 minutes. À $t_1 = t_0 + 30$ minutes, la borne dispose d'une localisation temporelle l_2 dont $q = 2$ même si aucun véhicule n'a explicitement exprimé vouloir se recharger à t_1. Cette valeur correspond aux deux véhicules qui sont en attente de rechargement depuis t_0. Comme à t_0, la borne ne pourra recharger qu'un seul véhicule, les deux autres véhicules de la file d'attente s'ajouteront donc à la liste des véhicules en attente de rechargement à t_1, et ainsi de suite. Cela fait partie de la dynamique propre à l'environnement temporel. Par conséquent, toutes ces mises à jour sont gérées au niveau de ce dernier.

Nous tenons à noter que dans ce chapitre, notre objectif est d'introduire essentiellement un raisonnement anticipatif basé sur AGRET d'un point de vue conceptuel. Les détails techniques et pratiques complémentaires sont abordés dans le prochain chapitre (chapitre 5).

4.2.2 Perception

La figure 4.2 schématise le cycle comportemental d'un agent intégrant le raisonnement anticipatif que nous proposons. Cette figure résulte de l'ajout de notre raisonnement anticipatif et de la perception de l'environnement temporel au modèle d'agent schématisé sur la figure 4.1.

Une description détaillée du fonctionnement du module perception est faite dans le chapitre 3. Les explications apportées à ce niveau concernent spécifiquement l'utilisation de ce module dans un contexte d'anticipation. Elles viennent compléter les détails déjà donnés auparavant.

Le modèle AGRET tel que nous le définissons dans 3.1 considère les trois dimensions : temps, espace physique et organisation comme des milieux d'interactions. Par conséquent, la perception de l'agent concerne non seulement la dimension spatiale, mais également la dimension temporelle et la dimension sociale. Dans chacun de ces environnements, cette perception est contrainte par un horizon de perception. Au niveau de l'environnement spatial, il s'agit du champ de vision. Au niveau de l'environnement temporel, nous parlons d'horizon temporel de perception. Au niveau de l'environnement social, dans la littérature SMA on parle notamment de réseaux d'accointances [20]. Par exemple : une liste d'amis, ou un cercle de connaissance dans les réseaux sociaux. La perception de l'environnement spatial ou social est classique. Par contre, la perception de l'environnement temporel est une nouveauté que nous introduisons dans cette thèse. Les percepts issus de l'environnement temporel interviennent au niveau de plusieurs étapes du cycle d'activation classique de l'agent. Durant la phase de perception du contexte d'activation, les percepts temporels permettent à l'agent de prendre connaissance de l'ensemble des activités qu'il prévoit d'effectuer à l'instant présent. En fonction de l'étendue de son horizon temporel de perception, il peut également percevoir, l'ensemble des activités qu'il projette d'effectuer dans le futur ou qu'il avait prévu d'effectuer dans le passé. En fonction des règles d'accessibilité, l'agent peut également percevoir des informations concernant la dynamique temporelle des autres agents, que ces derniers partagent au niveau de l'environnement temporel. Ces informations temporelles captées par l'agent découlent des localisations temporelles que chaque agent du système a définies et partagées au niveau de l'environnement temporel. Ces localisations temporelles sont rattachées à des slots temporels. Un slot temporel indique un point de l'axe temporel correspondant à un moment d'activation. L'ensemble est stocké dans l'environnement temporel. Dans la réalité, ce fonctionnement est similaire à la consultation d'un agenda privé ou partagé, d'un emploi du temps, ou de différentes sources d'informations partagées comme les réseaux sociaux, avant d'effectuer une action ou avant de planifier une future action.

Le fonctionnement de ce module perception se décrit formellement comme suit :

$$Perception : \underbrace{\Sigma_t \times \Delta_{pt}}_{temps} \times \underbrace{\Sigma_e \times \Delta_{pe}}_{espace} \times \underbrace{\Sigma_s \times \Delta_{ps}}_{organisation} \times \Gamma \mapsto P \tag{4.5}$$

où

— Σ est l'état de l'ensemble des environnements tel que

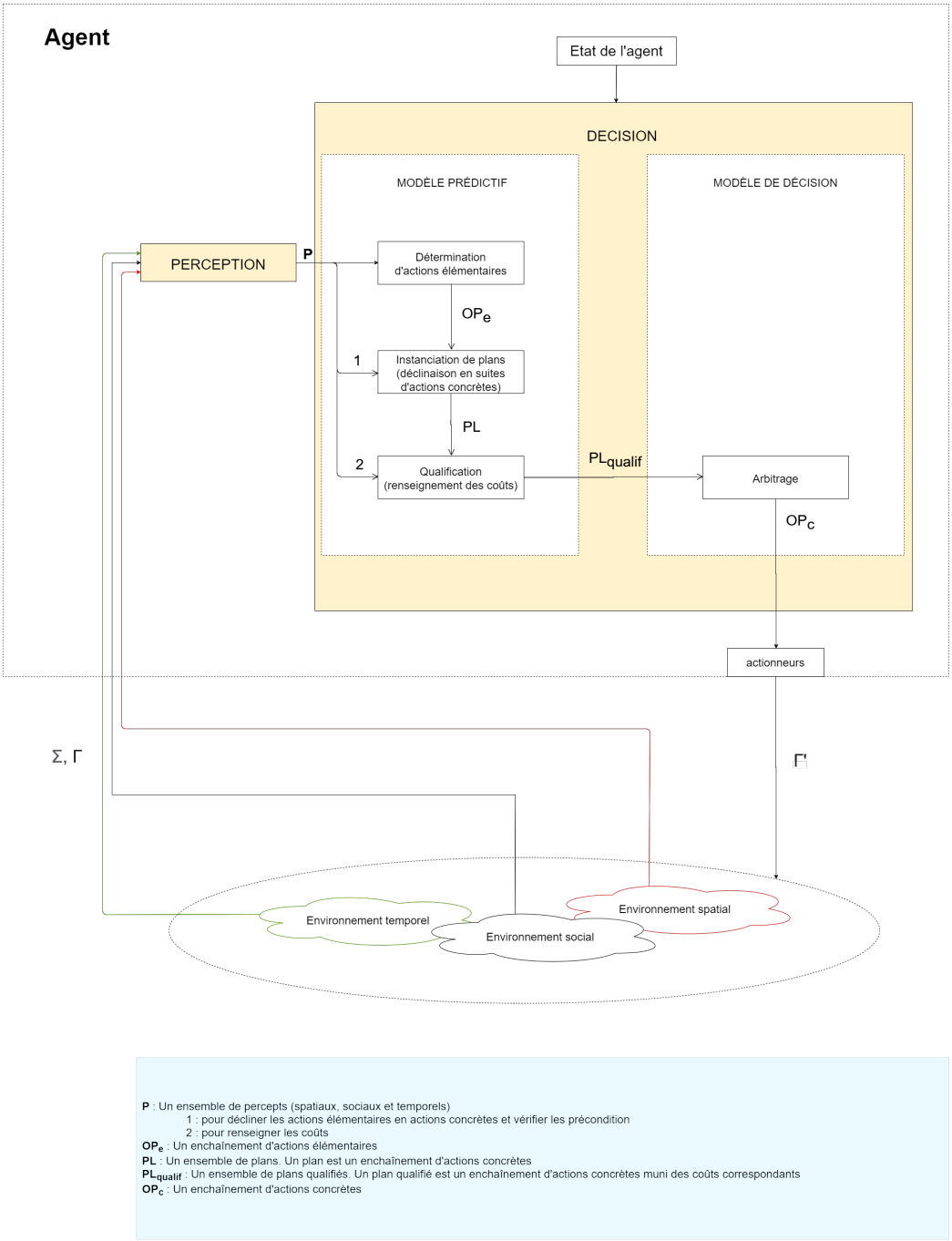


FIGURE 4.2 – Schéma de l'architecture agent intégrant une approche d'anticipation basée sur AGRET

- Σ_t est l'état de l'environnement temporel
- Σ_e est l'état de l'environnement spatial (espace)
- Σ_s est l'état de l'environnement social

— Γ est un ensemble d'influences externes
— Δ_p est un horizon de perception tel que

- Δ_{pt} est l'horizon temporel de perception
- Δ_{pe} est le champ de vision
- Δ_{ps} est l'horizon spatial de perception

— P est un ensemble de percepts spatiaux, sociaux et temporels :

$$P = P_e \times P_s \times P_t \tag{4.6}$$

Nous proposons une amélioration de la formule correspondante à la perception dans IRM4S. Nous distinguons clairement les horizons de perception : temporel (Δ_{pt}), spatial (Δ_{pe}) et social (Δ_{ps}) en les rajoutant explicitement dans la description formelle du fonctionnement du module perception. Ces horizons contraignent la perception de l'agent. Les percepts générés en sortie sont alors un ensemble de trois types de percepts différents, mais complémentaires : les percepts temporels (P_t), les percepts spatiaux (P_e) et les percepts sociaux (P_s).

Réglage de l'horizon temporel de perception par l'agent. L'agent peut être amené à jouer avec le réglage de son horizon temporel de perception. Il peut donc choisir d'élargir son champ de perception pour percevoir un ensemble plus grand d'informations temporelles. Cet élargissement lui permet d'enrichir l'ensemble des informations à sa disposition pour l'aider dans l'élaboration de son modèle prédictif et dans sa prise de décision. Dans le cas contraire, il peut réduire son champ de perception pour se concentrer uniquement sur quelques informations spécifiques qu'il juge suffisantes pour cela. Au niveau de l'environnement temporel, le réglage de l'horizon de perception entraîne une influence interne. Cette influence interne a pour vocation de modifier l'état interne de l'agent. Ce type d'influence est différent des influences externes qui ont pour objectifs de modifier l'environnement. Nous parlerons des influences externes dans 4.2.4.

Selon les besoins, il est également possible de définir des horizons temporels *ciblés* comme un horizon temporel périodique. Un exemple d'utilisation consiste en la perception du taux d'occupation d'une borne électrique tous les jours à 9 h pour anticiper les coûts correspondant à l'action de vouloir recharger son véhicule sur cette borne dans le futur à la même heure.

4.2.3 Modèle prédictif

Comme le montre la figure 4.2, notre modèle prédictif et notre modèle de décision sont intégrés au niveau du module décision de l'agent. La perception de l'ensemble du contexte d'activation spatial, social et temporel intervient en entrée à notre modèle prédictif, au niveau des trois (3) blocs qui le composent : la détermination d'une série d'actions élémentaires, la déclinaison de cette série d'actions élémentaires en actions concrètes (déclinaisons de plans), la qualification des actions concrètes par des coûts. Le modèle de décision quant à lui est composé d'un seul bloc appelé arbitrage. Nous allons détailler, dans cette sous-section, le fonctionnement des trois blocs composant notre modèle prédictif. Cependant, avant d'entrer dans ces détails nous décrivons dans la partie 4.2.3.1 suivante, le modèle d'action utilisé.

4.2.3.1 Modèle d'action

Plusieurs formalismes peuvent être utilisés pour modéliser l'action dans les SMA [29] : transformation d'un état global, réponse à des influences, processus informatique, modification locale, déplacement physique, commande, etc. Dans notre approche, nous définissons l'action comme la production d'une influence. Elle est considérée comme un opérateur *op* dont l'exécution produit une nouvelle influence. Ainsi, à partir d'un état σ_1 de Σ, l'exécution de l'action *op* produit une influence γ_1 de Γ. Par exemple, si à l'état σ_1, l'agent A se trouve à l'endroit $Lieu1$ et qu'on applique l'action $op = aller(A, Lieu1, Lieu2)$, l'influence γ_1 qui en résulte est le **souhait** de l'agent de se déplacer au lieu $Lieu2$:

$$\gamma_2 = Exec(aller(A, Lieu1, Lieu2), \sigma_1, \gamma_1) \tag{4.7}$$

$$Op \times \Sigma \times \Gamma \mapsto \Gamma \tag{4.8}$$

Nos actions ont les caractéristiques classiques suivantes :

Des préconditions. Certaines actions ne peuvent être réalisées directement. Elles ont besoin qu'un certain nombre de préconditions soient remplies avant de pouvoir être réalisées. Dans la pratique, ces conditions doivent être vérifiées avant que l'on ne puisse appliquer l'opérateur. Ces préconditions peuvent :

- porter sur des objets ou sur la valeur des variables de l'inventaire de l'agent. Cet inventaire peut être une liste des possessions matérielles ou autre. Ces objets ou valeurs peuvent être retirés lors de l'action. Par exemple : avoir en sa possession un véhicule et avoir un niveau de charge de batterie suffisant pour pouvoir effectuer l'action "se déplacer". Une fois exécutée, cette action diminue la valeur de la charge de la batterie ;
- être uniquement des prérequis. Par exemple : l'action "travailler" nécessite que l'agent soit déjà sur son lieu de travail ;
- porter sur la réalisation au préalable d'un ensemble d'actions. Par exemple : l'action "travailler" nécessite l'exécution de l'action "se déplacer à son lieu de travail" (si l'agent n'y est pas).

Les préconditions permettent le chaînage des actions en vue de réaliser un but. Si un agent veut obtenir les effets d'une action, mais que les préconditions de cette action ne sont pas remplies, il va chercher un ensemble d'autres actions à effectuer au préalable, et ainsi de suite.

Des influences. A priori, toutes les actions ont une influence sur l'environnement appelée influence externe, ou sur l'agent qui l'exécute appelée influence interne.

- Les influences externes concernent les actions des agents qui peuvent avoir comme objectif de modifier l'environnement. Par exemple : la demande de création d'une nouvelle localisation temporelle dans l'environnement temporelle, ou la tentative de déplacement d'un objet dans l'environnement spatial.
- Les influences internes quant à elles sont celles qui affectent directement l'agent via les variables internes ou son inventaire. En effet, beaucoup de préconditions des actions reposent sur les variables d'inventaire. Souvent un agent est contraint de modifier ses actions en priorisant d'abord celle lui permettant d'obtenir une ressource d'inventaire nécessaire à la réalisation de l'action qui l'intéresse vraiment. Par exemple : Un agent a_1 souhaite aller travailler. Cependant, il n'a pas assez d'énergie pour effectuer le trajet domicile-travail. Il va donc d'abord devoir se recharger sur une borne de recharge b_1 qui n'est pas forcément située spatialement à proximité de lui. Il va devoir réaliser l'action "se déplacer vers b_1" avant de pouvoir "se brancher sur b_1". Cela aura 2 effets sur son inventaire :
  - la diminution de son niveau de batterie pendant le déplacement pour qu'il puisse se brancher à la borne ;
  - Il pourra, par la suite, réellement se recharger et cela permettra l'augmentation de son niveau de batterie jusqu'à obtenir la quantité de charges suffisante pour aller travailler.

Des coûts Nous partons du principe selon lequel les actions peuvent avoir des importances et rôles différents selon les simulations. Nous laissons donc la possibilité au modélisateur de rajouter des critères de coût sur les actions. Les coûts peuvent dépendre de la simulation ainsi que des objectifs de l'agent ou du modélisateur. Certains coûts sont obligatoires, d'autres sont optionnels. Dans notre modèle, ces critères de coût peuvent également être objectifs ou subjectifs. Les coûts objectifs sont factuels. Ils sont calculés directement à partir de données perçues au niveau des environnements comme le coût temporel, le coût énergétique ou la longueur prévisionnelle de la file d'attente. Les coûts subjectifs quant à eux concernent plus un ressenti de l'agent par rapport à la réalisation de l'action. Un exemple de coût subjectif est la criticité, c'est-à-dire le degré d'importance que l'agent accorde à une action. Pour aider à l'arbitrage, ces critères de coût peuvent être sujets à des pondérations. Ces pondérations sont prédéfinies par le modélisateur en fonction de la simulation et du type d'agent. Nous en reparlons plus en détail dans 4.2.4.

Des actionneurs. En fonction de la simulation, le modélisateur dote les agents d'un ensemble d'actionneurs. Chaque action qu'un agent peut effectuer utilise un certain nombre de ces actionneurs. Le nombre d'actions simultanées qu'un agent peut accomplir est donc limité par le nombre d'actionneurs dont il dispose. Évidemment, en fonction de la simulation et du résultat désiré, les

actionneurs pertinents ne sont pas les mêmes. Dans certains cas, le modélisateur peut avoir besoin d'augmenter le nombre d'actionneurs, alors que dans d'autres, un seul actionneur peut suffire. Exemple : un agent possédant un actionneur "créateur de localisation temporelle" et un actionneur "modificateur de localisation temporelle" peut définir à la fois une nouvelle localisation temporelle dans l'environnement temporel et en modifier une autre.

4.2.3.2 Détermination d'actions élémentaires

Ce module permet à l'agent de générer un enchaînement d'actions élémentaires à réaliser en tenant compte de l'ensemble des percepts qu'il reçoit du module perception et de son état interne. Pour ce faire, il fait correspondre une action élémentaire à une localisation temporelle, selon les informations contenues dans cette dernière et autres percepts.

Nous illustrons ce fonctionnement à travers l'exemple que nous avons défini précédemment dans 4.2.1. Nous nous focalisons uniquement sur la perception de l'automobiliste de son environnement temporel qui est contrainte par son horizon temporel $\Delta_{pt} = [5, 4]$. Il perçoit donc les localisations temporelles comprises dans l'intervalle $[t-5, t+4]$. S'il est actuellement 8 h, cet horizon limite sa perception temporelle à la perception des localisations temporelles rattachées aux slots temporels situés entre 3 h et 12 h. L'agent est actuellement à son domicile. L'ensemble de percepts temporels que l'automobiliste peut récolter est alors le suivant : $\{l_1 = \{"travail1", "8", "12", "24", "0.1"\}, l_2 = \{"loisir1", "12", "14", "", 0.2"\}\}$
où l_1 et l_2 sont des localisations temporelles de la forme $l = \{id, d, f, p, v\}$ comme définies auparavant dans le chapitre 3.

L'ensemble des actions élémentaires correspondantes est :
$OPe = \{ope_1 = "aller\ travailler", ope_2 = "aller\ se\ divertir"\}$

De manière formelle, le fonctionnement du module de détermination d'actions élémentaires consiste en l'application de la fonction :

$$P \mapsto OPe \tag{4.9}$$

où P est l'ensemble des percepts spatiaux, sociaux et temporels. Ope est un enchaînement d'actions élémentaires.

4.2.3.3 Instanciation de plans.

Pour commencer, nous rappelons que comme défini précédemment dans 4.1.3, nous distinguons deux types d'actions :

— l'action élémentaire qui correspond à l'application de l'opérateur *ope* ;
— l'action concrète qui correspond à l'application de l'opérateur *opc*.

Une action élémentaire est abstraite. Une action concrète est l'instanciation d'une action élémentaire dans les environnements, c'est-à-dire une version située de l'action élémentaire. Par exemple : "se recharger" est une action élémentaire tandis que "se recharger sur une borne b_1 de coordonnée (x, y) à l'instant t_1" est une action concrète. Un enchaînement d'actions concrètes OPc est appelé plan (cf définition 4.1.5).

Comme mentionné dans 4.1.1, l'approche d'anticipation que nous proposons diffère de la plupart des approches classiques d'anticipation que nous retrouvons dans la littérature pour les raisons suivantes :

— nous exploitons la perception des informations temporelles permises par l'environnement temporel dans la mise en oeuvre de notre modèle du monde ;
— nous exploitons des informations passées et présentes, mais également les informations sur le futur planifié, au niveau du raisonnement de l'agent.

L'instanciation de plans consiste en l'exécution de deux processus : la situation d'une action élémentaire au niveau des environnements (déclinaison en actions concrètes), la vérification de l'applicabilité d'une action concrète (vérification des préconditions).

1. Situation au niveau de l'environnement ou déclinaison en actions concrètes. À la réception de l'ensemble des actions élémentaires OPe, et en fonction de l'ensemble des percepts spatiaux, sociaux et temporels, ce module situe l'ensemble des actions élémentaires dans l'environnement. Ces actions élémentaires deviennent alors des actions concrètes.

Exemple : Notre ensemble d'actions concrètes est :
$\{opc_1 =$ "*aller travailler au lieu* (x, y) 8 *h*",
$opc_2 =$ "*se divertir au lieu* (x_1, y_1) 12 *h*" **ou**
$opc'_2 =$ "*se divertir au lieu* (x_2, y_2) 12 *h*"$\}$

Dans cette démarche de déclinaison des actions élémentaires en actions concrètes, nous considérons deux cas :

— Une seule option s'offre à l'agent : dans ce cas une seule action concrète correspond à l'action élémentaire. Exemple : L'agent n'a qu'un seul lieu de travail et travaille selon un horaire fixe ;

— Plusieurs options s'offrent à l'agent : dans ce cas, l'action élémentaire est déclinée en plusieurs actions concrètes. Nous aurons donc, un ensemble d'actions concrètes correspondant. Exemple : L'agent a le choix entre plusieurs lieux de loisirs avec des horaires différents. Cela entraîne la création de plusieurs déclinaisons de plans parmi lesquels le modèle de décision devra faire son choix : $\{OPc_1, OPc_2, ..., OPc_n\}$.

2. Vérification de l'applicabilité de chaque action concrète. Cette étape consiste à vérifier que toutes les préconditions nécessaires à l'exécution de chaque action concrète sont remplies.

Exemple : pour l'action "aller travailler à l'endroit de coordonnées (x, y) à 8 h", ce module permet à l'automobiliste de vérifier s'il dispose d'un niveau de batterie suffisant pour effectuer le trajet. Les percepts spatiaux lui permettent de déterminer la distance du trajet entre sa localisation actuelle et son lieu de travail. À partir de cette information, il estime l'énergie nécessaire pour effectuer le trajet. Comme mentionné dans 4.2.3.1, les préconditions permettent le chaînage des actions en vue de réaliser un but.

Deux cas de figure peuvent se présenter :

— **Dans le cas où l'action concrète est applicable**, c'est-à-dire que les préconditions sont vérifiées, il passe directement à la vérification des conditions préalables relatives à l'action suivante. Cette étape se répète jusqu'à ce que toutes les actions soient vérifiées. À la fin des itérations, le module renvoie directement l'ensemble de plans en sortie. Chaque plan est un enchaînement d'actions concrètes.

— **Dans le cas où les préconditions d'une action ne sont pas remplies**, le module de détermination d'actions concrètes cherche une autre action à réaliser au préalable. Cette dernière est choisie de manière à permettre à l'agent de satisfaire aux préconditions de l'action qu'il devait effectuer à l'origine. Il réitère le processus jusqu'à ce qu'il trouve une action ou un ensemble d'actions applicables.

Exemple :

1. Le module reçoit les percepts suivants : il est 8 h, l'automobiliste est situé à son domicile et il a 30% de batterie. La première action concrète à évaluer est "aller au travail (lieu de travail déjà connu) à 8 h".
2. La précondition requise pour pouvoir exécuter cette action est "avoir assez de charges de batterie pour effectuer le trajet domicile-travail".
3. Il calcule le niveau de batterie nécessaire. Supposons qu'il constate que son niveau actuel de batterie n'est pas suffisant (la précondition n'est pas satisfaite). Il va choisir d'exécuter au préalable l'action "aller se recharger". Cette dernière va lui permettre d'augmenter son niveau de batterie pour qu'il puisse par la suite exécuter l'action "aller au travail".

Une fois cette étape terminée, nous aurons un ensemble de plans notés PL. Chaque plan OPc composant PL résulte des diverses déclinaisons d'actions concrètes candidates. De manière formelle, le fonctionnement de ce module est traduit par la fonction suivante :

$$OPe \times P \times S \mapsto PL \tag{4.10}$$

où S est l'état interne de l'agent, OPe est un enchaînement d'actions élémentaires $P = P_e \times P_t \times P_s$ est l'ensemble des percepts (spatiaux, temporels et sociaux) et PL l'ensemble des déclinaisons de plans d'actions concrètes candidates.

PL est de la forme :

$$PL = \{OPc_1, OPc_2, ..., OPc_n\} \tag{4.11}$$

où

$$OPc_1 = \{opc_1, opc_{c2}, OPc_3, ..., opc_m\},$$
$$...,$$
$$OPc_n = \{opc_1^n, opc_2^n, ...opc_m^n, opc_{m+1}\}$$

où $\{opc, opc', ..., opc^n\}$ est l'ensemble des n déclinaisons d'une action élémentaire. OPc est un plan et opc est une action concrète.

Il est tout à fait possible que chaque déclinaison de plan n'ait pas le même nombre d'actions concrètes. Par exemple : Pour une action élémentaire : "se divertir". L'agent peut avoir le choix entre plusieurs lieux de loisir. Ce qui entraîne plusieurs déclinaisons d'actions concrètes. En fonction de l'énergie nécessaire pour effectuer le trajet vers chaque lieu de loisir, il est possible qu'une déclinaison nécessite l'exécution d'une action au préalable. Par exemple : se recharger si le niveau d'énergie disponible ne permet pas d'effectuer le trajet.

À ce niveau, le processus d'anticipation permet d'augmenter ou, au contraire, de limiter le nombre d'actions concrètes candidates via un flux de perception optimisé grâce à un réglage de l'horizon de perception.

Réglage des horizons de perception. Plus les horizons de perceptions (spatial, temporel et social) sont larges, plus la probabilité d'avoir un nombre important de déclinaisons d'actions concrètes est élevée. Par exemple : Plus le champ de vision est large, plus le nombre de zones de loisir à proximité (inclus dans le champ de vision) peut être important. En particulier, au niveau de l'environnement temporel, un horizon temporel large signifie que le raisonnement anticipatif permet à l'agent de remettre en question des activités éloignées dans le futur. Cependant, le nombre d'itérations au niveau de la vérification d'applicabilité est proportionnel au nombre d'actions concrètes. Ces itérations sont coûteuses en temps et en puissance de calcul. Pour des raisons de performance, il est alors important de limiter le nombre de déclinaisons en définissant un nombre maximum d'itérations. Ce nombre prend une valeur arbitraire définie par le concepteur du modèle ou par l'utilisateur. Dans le cas contraire, plus les horizons de perception sont étroits, plus la probabilité d'avoir des déclinaisons d'actions concrètes est réduite.

Par ailleurs, le nombre d'actions concrètes n'est pas seulement proportionnel à l'étendue des horizons de perceptions (temporelle, spatiale et sociale), il varie aussi en fonction de la densité des informations perceptibles au niveau des environnements. Dans le cas de l'environnement temporel, les règles d'accessibilité influent énormément sur le volume d'information que l'agent peut percevoir. Moins les règles d'accessibilité sont strictes, plus l'agent peut percevoir de localisations temporelles. Un volume plus grand d'informations temporelles peut être alors pris en compte dans le processus de décision. Cela augmente le besoin en ressources computationnelles nécessaires au traitement de ces informations. La pertinence des règles appliquées est fonction des objectifs de la simulation et des modélisateurs. Il revient donc au modélisateur de trouver le bon équilibre dans la définition des règles d'accessibilité.

Nous tenons tout de même à noter que la perception de l'environnement temporel est limitée naturellement par des lois de l'environnement comme l'horizon de stockage par exemple. Une fois encore, l'étendue de cet horizon temporel de stockage varie en fonction des objectifs du modélisateur et du modèle de simulation. Ainsi, même si la valeur de l'horizon temporel de perception permet à l'agent de voir au-delà des limites définies par l'horizon de stockage, cela n'aura aucune incidence sur la simulation. En effet, l'horizon de stockage limite les informations contenues au niveau de l'environnement. L'agent ne pourra tout simplement pas percevoir ce qui n'existe pas dans l'environnement temporel.

Pour résumer, le réglage de l'horizon temporel de perception est complexe. Une portée trop petite revient à ne rien anticiper. Contrairement à cela, une portée trop grande est coûteuse en termes de calcul pour peu de bénéfice. En effet, il se peut que l'ensemble des informations accessibles à l'instant actuel ne soit pas suffisant pour qu'une remise en cause d'une décision lointaine dans le futur soit pertinente. De même, du point de vue de la visibilité, toutes les informations accessibles par l'agent ne lui sont pas forcément utiles. Au contraire, il risque de se faire polluer d'informations

non pertinentes qui alourdissent les traitements. C'est pourquoi il est très important de choisir une valeur optimale de cet horizon de perception et de définir des règles de visibilité en fonction des objectifs et des particularités de la simulation et des agents.

Conditions d'arrêt La question de la condition d'arrêt du processus de vérification de l'applicabilité des actions concrètes est également primordiale. En effet, comme nous l'avons mentionné précédemment, les itérations sont coûteuses en temps et en ressources computationnelles. De plus, dans des cas extrêmes où aucune précondition ne peut être satisfaite, il se peut que l'algorithme tombe dans une boucle infinie. Notre proposition consiste alors à définir un nombre maximum d'itérations et/ou une durée d'attente maximum (time out). La valeur de chacun de ces critères d'arrêt est définie arbitrairement par le concepteur.

4.2.3.4 Qualification des actions concrètes

Cette étape consiste à renseigner l'ensemble des coûts correspondant à chaque déclinaison d'action concrète.

Definition 4.2.1. Action concrète qualifiée. Une action concrète candidate, combinée avec ses coûts devient une action concrète qualifiée. Le chaînage d'actions concrètes et d'actions concrètes qualifiées est appelé plan d'actions concrètes qualifiées OPc_q. L'ensemble des différentes déclinaisons de plans d'actions concrètes qualifiées est noté PL_{qualif}.

Il est important de noter qu'il est inutile de qualifier toutes les actions concrètes. Les seules actions concrètes qualifiées sont :

— celles qui présentent plusieurs déclinaisons ;
— celles qui ont été rajoutées comme actions à réaliser au préalable. En d'autres termes, celles qui n'étaient pas prévues dans le emploi du temps initial, mais qui ont été introduites suite à l'étape de la vérification de l'applicabilité.

La principale raison est que les coûts servent au module arbitrage dans le choix de l'instanciation de plan la plus pertinente. Dans ce cadre, se concentrer uniquement sur les coûts des actions concrètes propres à chaque déclinaison de plan suffit pour différencier chaque plan. Par exemple : Soient les plans OPc_1 =“aller travailler à (x,y) à t0”,“se recharger à b1 à t1”,“aller se divertir à (x1,y1) à t2” et OPc_2 =“aller travailler à (x,y) à t0”,“aller se divertir à (x2,y2) à t2”. OPc_1 et OPc_2 sont des déclinaisons de plans correspondants à OPe =“aller travailler”,“se divertir”. Le coût correspondant aux actions concrètes “se recharger” (action à réaliser au préalable) et “se divertir” (action ayant plusieurs déclinaisons) suffisent pour départager les deux déclinaisons de plans. Renseigner les coûts de l'action “aller travailler” qui est commune aux deux déclinaisons n'est pas forcément utile.

Pour renseigner ces coûts, le module se sert des percepts. Exemple : les percepts de l'environnement spatial lui permettent de renseigner la quantité d'énergie nécessaire pour effectuer l'action. Au niveau de l'environnement temporel, il peut récolter la longueur prévisionnelle de la file d'attente au niveau d'une borne. Enfin, au niveau de l'environnement social, il a le nombre prévisionnel de personnes comptant se rendre au même endroit.

Coûts. Les percepts permettent également à l'agent d'enrichir et de qualifier une action concrète par des coûts. Ces coûts sont nécessaires à l'arbitrage de la décision. Prenons l'exemple de trois coûts :

1. le coût associé la longueur prévisionnelle de la file d'attente q' ; (q est le paramètre contenant la longueur de la file d'attente au niveau de la localisation temporelle, q' est le coût associé)
2. le coût énergétique e ;
3. la criticité cr.

Pour faciliter les traitements au niveau du modèle de décision, nous faisons revenir les valeurs de chacun des coûts à une valeur comprise entre l'intervalle $[0, 1]$.

Calcul de q', coût associé à la longueur de la file d'attente. La longueur prévisionnelle de la file d'attente associée à une action concrète opc_1=“se recharger à b_1 à t” est obtenue à partir de la valeur du paramètre facultatif q de la localisation temporelle de b_1 au slot t. Pour que l'agent puisse récupérer cette valeur, t doit être compris dans son horizon temporel de perception. Afin

d'enrichir cette information, l'agent peut la croiser ou la compléter avec d'autres informations contenues dans d'autres environnements. Par exemple : Au niveau de l'environnement social, il peut effectuer une requête qui consiste à récolter le nombre actuel de conducteurs qui sont en train de signaler à la borne b_1 leur intention de s'y recharger à t. En d'autres termes, en plus des agents déjà comptés dans la valeur de q, combien d'autres agents conducteurs de véhicule seraient actuellement intéressés pour se recharger sur b_1 à t.

Le calcul de cette valeur de coût s'effectue en prenant comme référence la plus grande valeur de la longueur de la file d'attente. Par exemple : Si nous avons trois déclinaisons de la même action concrète et que les valeurs des longueurs prévisionnelles de la file d'attente sont respectivement 8, 10 et 0. Alors la valeur de référence est 10. Pour avoir les coûts correspondants, nous divisons chacune de ces valeurs par 10. Les valeurs de q' pour chaque déclinaison d'action concrète sont donc respectivement $q' = 0.8$ c'est-à-dire (8/10), $q' = 1$ c'est-à-dire (10/10) et $q' = 0$ c'est-à-dire (0/10).

Calcul de e, coût énergétique. La valeur de l'énergie nécessaire peut par exemple être calculée en fonction de la moyenne de la consommation énergétique du véhicule et de la distance du trajet entre l'emplacement actuel du véhicule et celui de la borne. Les informations nécessaires pour ce calcul sont perceptibles au niveau de l'environnement spatial, en fonction du champ de vision de l'agent.

Comme pour le calcul de la valeur de q', nous prenons comme référence la valeur maximale de la quantité d'énergie nécessaire. Par exemple si les quantités d'énergie nécessaires sont respectivement 5000, 6000 et 10000, alors les valeurs des coûts énergétiques sont $e = 0.5$, $e = 0.6$ et $e = 1$.

La longueur prévisionnelle de la file d'attente ainsi que la quantité d'énergie requise sont des coûts objectifs. Cela veut dire qu'ils se calculent sur la base de données factuelles réelles. La criticité quant à elle est un coût subjectif.

Calcul de c, coût associé à la criticité. La criticité indique le degré d'importance que l'agent accorde à l'action concrète correspondante. Plus la valeur de la criticité se rapproche de 0, moins l'agent accorde de l'importance à l'action associée. Contrairement à cela, plus la valeur de la criticité se rapproche de 1, plus l'agent considère l'action comme importante.

Une façon de calculer la criticité consiste à se baser sur un paramètre porté par les localisations temporelles passées et que nous avons évoquées dans 3.3.2. Ce paramètre est la *satisfaction*, noté s. Sa valeur correspond à une note que l'agent attribue à une localisation temporelle passée après avoir évalué son exécution. Ainsi, en fonction de la valeur de la satisfaction relative à une action similaire dans le passé, de l'état interne actuel de l'agent, et des différentes informations présentes et futures que l'agent peut percevoir au niveau de l'environnement temporel, l'agent pourra renseigner la criticité d'une action concrète. Par exemple, pour l'action concrète opc_1 ="se recharger à la borne b_1 à t". Plusieurs localisations temporelles passées concernent des actions similaires, c'est-à-dire "se recharger à la borne b_1 à un instant quelconque dans le passé". Pour toutes ces localisations temporelles, la valeur du paramètre satisfaction est proche de 0, c'est-à-dire que les actions se sont mal passées. Dans ce cas, si l'agent dispose encore de suffisamment de charges de batterie pour effectuer sa prochaine action, l'agent peut alors estimer que l'action concrète opc_1 aura une criticité faible. Cela signifie que l'agent accorde une moindre importance à cette action concrète.

Une action concrète qualifiée est de la forme $\{opc, CO\}$ où CO est un ensemble de coûts. Exemples de déclinaisons d'actions concrètes qualifiées :

— $\{opc_q$ ="se recharger sur la borne b_1 à t_1",$\underbrace{\{q' = 0.8, e = 0.5, c = 0.1\}}_{ensemble\ de\ coûts\ CO}\}$
— $\{opc'_q$ ="se recharger sur la borne b_1 à t_2",$\{q' = 1, e = 0.6, c = 0.1\}\}$
— $\{opc''_q$ ="se recharger sur la borne b_2 à t_1",$\{q' = 0, e = 1, c = 0.7\}\}$

Le fonctionnement du module qualification correspond alors à l'application d'une fonction $Qualification$ qui prend en entrée un ensemble de plans d'actions concrètes candidates PL, un ensemble de percepts (spatiaux, temporels et sociaux) P, l'état interne S de l'agent. Cette fonction génère en sortie un ensemble de plans d'actions concrètes qualifiées PL_{qualif}.

$$Qualification : PL \times P \times S \mapsto PL_{qualif} \tag{4.12}$$

Réglages et conditions d'arrêt. En fonction de la richesse des informations contenues dans l'environnement temporel, plus l'horizon temporel est large, plus les coûts pourraient être précis. Cependant, cela se fait au risque d'alourdir le traitement des informations et/ou d'être submergé d'informations qui ne sont pas forcément indispensables à la prise de décision. Contrairement à cela, plus l'agent réduit son champ de perception, moins les coûts sont précis. Cependant, cela permet d'alléger les traitements et/ou de se concentrer sur les informations spécifiques qui peuvent suffire pour faire un choix pertinent. Comme pour l'étape précédente, la question de la condition d'arrêt du processus est donc primordiale. En effet, se focaliser sur un champ de perception trop petit peut revenir à ne rien anticiper. Dans le cas contraire, trop essayer d'élargir son champ de perception peut revenir à consommer énormément de ressources pour un résultat potentiellement inutile. Différentes possibilités sont alors envisageables pour gérer cette condition d'arrêt. La plus naturelle est l'utilisation d'une simple durée. Cela consiste à fixer les valeurs maximales et minimales de l'horizon temporel au-delà desquelles on considère que les informations ne sont plus assez pertinentes pour être prises en compte. Cette durée peut-être déterminée de plusieurs manières. Elle peut être fixée arbitrairement par le modélisateur, en fonction des ressources disponibles et du nombre d'agents simulés. Elle peut également être déterminée suite à des tests concernant la qualité des anticipations, en fonction de la durée anticipée. Lors du choix de sa valeur, il est important de prendre en compte le fait que chaque itération de cette boucle peut être coûteuse, à la fois en ressources et en temps. Une autre condition d'arrêt naturelle consiste aussi à fixer un nombre maximum d'itérations.

Par défaut, la limite concernant l'élargissement de l'horizon de perception est imposée par l'environnement temporel lui-même via l'horizon temporel de stockage. Cependant, il est également possible pour le modélisateur de définir lui-même les valeurs maximales de cet horizon temporel de perception. Bien évidemment, les conditions d'accès définies au niveau de l'environnement temporel (cf 3.3.3.1) viennent compléter la fonction de l'horizon temporel de perception. Cela permet de respecter le principe selon lequel un agent doit avoir une perception limitée de son environnement.

Abonnement et notification Si la question de l'horizon temporel de perception est intéressante, celle de la fréquence de déclenchement du processus anticipatif doit également être posée. Ici encore, une fréquence trop faible risque de ne pas faire détecter un certain nombre de comportements intéressants pour l'agent et/ou pour l'ensemble du système. Contrairement à cela, une fréquence trop élevée sera contre-performante. Cela revient à remettre en question plusieurs fois le même comportement.

Dans ce cadre, il existe plusieurs façons de déclencher l'activation des agents pour que celui-ci puisse déclencher son processus de perception qui induit le processus de raisonnement anticipatif. Nous en citons deux :

- La programmation de son activation par l'agent lui-même en définissant une localisation temporelle au niveau de l'environnement temporel ;
- L'activation de l'agent suite à la réception d'une notification due à un changement au niveau de l'environnement.

La première méthode relève d'un fonctionnement classique de l'ordonnanceur de la simulation que nous avons déjà bien détaillée dans les parties précédentes. Dans cette partie, nous nous focalisons sur la deuxième méthode qui est beaucoup moins commune en termes d'utilisation. Dans la littérature, certains travaux comme ceux de Badeig [6] ou Reynaud [81] prennent en compte cet aspect de l'environnement. Dans sa thèse, Badeig met en place un environnement *actif* appelé Eass (Environment as Actif Support for Simulation). Ce modèle intègre un processus de sélection d'actions des agents via des filtres afin de prendre en charge la gestion du processus d'activation. Reynaud quant à lui propose une utilisation de la sémantique de l'environnement dans le processus de décision. Pour cela, il procède de trois manières différentes. Celle qui nous intéresse est celle que l'on appelle *proactive*. Dans ce cadre, la sémantique de l'environnement peut proposer d'elle-même des informations aux agents. Pour ce faire, il faut que ces agents s'inscrivent comme intéressés par un certain type de service. Quand un service correspondant est détecté dans son environnement proche, une notification lui est envoyée.

Dans l'approche d'anticipation que nous proposons, l'agent peut être notifié des événements pouvant avoir des conséquences sur l'exécution d'une action qu'il a planifiée dans le futur (sous forme de localisation temporelle). Il peut alors être amené à remettre en question son souhait d'exécution de cette action. Ces événements concernent la création, la suppression, ou la modification d'une ou d'un ensemble de locations temporelles par d'autres agents. Cette fonctionnalité peut être

mise en place et activée par défaut par le concepteur du système. Elle peut également être activée volontairement par les agents, sous forme d'abonnement à un slot particulier de l'environnement temporel ou à un objet particulier de l'environnement spatial comme une borne de recharge par exemple. Dans notre modèle de simulation, lorsqu'un automobiliste réserve une borne, la borne définit une localisation temporelle qui traduit sa volonté de recharger le véhicule à un instant t. Dans ce cadre, elle inscrit automatiquement l'automobiliste en tant qu'abonnée aux mises à jour concernant les modifications de cette localisation temporelle. Ainsi, l'automobiliste est tenu au courant, par exemple, d'un changement au niveau de la longueur prévisionnelle de la file d'attente. Il peut par la suite remettre en question son action de se recharger sur cette borne à cet instant. En fonction de ses objectifs, si l'agent suppose que la file d'attente est trop longue, il peut relancer son processus de raisonnement afin de rechercher une autre borne de recharge ou un autre créneau qui lui convienne mieux.

Notre système de notification exploite la dimension sociale du modèle AGRET dans le sens où les notifications se font par communication par le biais de l'environnement social. Son fonctionnement requiert une combinaison et un croisement des informations de la dimension temporelle, la dimension sociale et la dimension spatiale. Un exemple d'utilisation très simple de ce système consiste à ajouter un autre paramètre optionnel qui stocke l'identifiant de tous les abonnés à la borne. Ainsi, la borne pourra envoyer une notification sous forme de message par le biais de l'environnement social aux automobilistes concernés en utilisant leur identifiant. Nous en reparlons plus en détail dans le chapitre 5 suivant. Une utilisation plus élaborée du système d'abonnement et de notification consiste à définir un groupe d'abonnés au niveau de l'environnement social. L'abonnement à un objet ou à une localisation temporelle consiste alors en une inscription au groupe d'abonnés correspondant. Une fois inscrit dans ce groupe, l'agent aura accès à un certain nombre d'informations relatives à l'objet ou à la localisation temporelle. Il sera également notifié des mises à jour correspondantes. Nous en reparlons dans 4.3.2.2.

Un autre fonctionnement de ce processus de notification consiste à alerter tous les agents à proximité lorsqu'une borne de recharge est disponible. La notion de proximité ici peut être spatiale, sociale, ou temporelle. Un agent à proximité spatiale d'une borne est un agent qui se situe à une certaine distance physique de la borne. Un agent à proximité sociale est par exemple un agent qui fait partie d'un groupe particulier. Un agent à proximité temporel est un agent qui a défini une localisation temporelle similaire aux alentours du slot temporel courant. Il peut s'agir par exemple d'un agent qui a prévu de se recharger sur la borne dans l'heure qui suit. L'objectif est d'inciter un agent à venir se recharger dans l'immédiat de manière à mieux répartir l'occupation des bornes de recharge dans le temps et dans l'espace. Pour ce faire, la borne perçoit l'environnement pour trouver les véhicules à proximité. Elle utilise ensuite l'environnement social pour notifier les automobilistes.

La fréquence de la remise en question d'une décision future (localisation temporelle dans le futur) est une caractéristique importante du raisonnement anticipatif. En effet, une fréquence trop importante est coûteuse, et peut être contre-productive, par exemple si l'agent change d'avis trop souvent. À l'inverse, une fréquence trop faible limite la réactivité de l'agent et le rend trop rigide dans ses choix. Dans notre cas, cette fréquence est paramétrable par le concepteur du modèle.

4.2.4 Modèle de décision

Le modèle de décision correspond au fonctionnement du module d'arbitrage. Ce fonctionnement consiste en la sélection de l'instance de plan la plus pertinente parmi l'ensemble PL_{qualif} des plans d'actions concrètes candidates qualifiées générées par le modèle prédictif. Une action concrète candidate est un couple (opc, CO) où $CO = \{co_1, co_2, ..., co_n\}$ est un ensemble de coûts correspondant à *opc*. Dans notre modèle, l'algorithme de sélection de plans applique une pondération sur les différents coûts puis choisit le plan dont la somme des coûts pondérés de toutes les actions concrètes qui le compose est minimale. Ce plan est considéré comme étant l'instance de plan la plus pertinente.

Pondérations. Les pondérations p_n sont arbitraires, définies par le modélisateur, peuvent être propres à chaque agent ou à chaque type d'agents et à chaque coût co_n. Soient : l'action concrète opc dont l'ensemble des coûts est $\{co_1, co_2, ..., co_n\}$ et l'ensemble des pondérations correspondant est $Po = \{po_1, po_2, ..., po_n\}$. La somme des coûts pondérés est calculée de la manière suivante :

$$sp = \sum_{i=1}^{n} (co_i * po_i) \tag{4.13}$$

Exemple. Soient les pondérations $p_L = 0.5$, $p_e = 0.25$, $p_{cr} = 0.25$ correspondant à chacun des coûts q' (longueur de la file), e (coût énergétique), et cr (criticité). Les sommes des coûts pondérés correspondant à chaque action concrète de l'exemple donné dans 4.2.3.4 sont :

— $\{opc_{p1}$ ="se recharger sur la borne b_1 à $t+1$",$\{c = 0.8 * 0.5 + 0.5 * 0.25 + 0.1 * 0.25 = 0.55\}\}$
— $\{opc_{p2}$ ="se recharger sur la borne b_1 à $t+2$",$\{c = 1 * 0.5 + 0.6 * 0.25 + 0.1 * 0.25 = 0.67\}\}$
— $\{opc_{p3}$ ="se recharger sur la borne b_2 à $t+1$",$\{c = 0 * 0.5 + 1 * 0.25 + 0.7 * 0.25 = 0.425\}\}$

Nous avons donc un ensemble de couples d'actions concrètes pondérées $opc_p = (opc, sp)$. Le module arbitrage calcule la somme de tous les sp de tous les opc_p qui composent le plan. La somme est notée SP. Le plan dont la valeur de SP est minimale est considéré comme le plan le plus pertinent. En cas de litige, c'est-à-dire dans le cas où l'on a plusieurs plans dont la valeur de la somme des coûts pondérés est la même, le module d'arbitrage en sélectionnera aléatoirement un parmi ceux dont le total des coûts pondérés est le plus bas. Le module arbitrage correspond alors à l'application de la fonction :

$$Arbitrage : PL_{qualif} \times Po \mapsto OPc \tag{4.14}$$

Chaque action concrète contenue dans le plan OPc sélectionné est envoyée aux actionneurs correspondant pour être transformée en influences. Ces influences peuvent être internes, c'est-à-dire qu'elles ont une incidence sur l'état interne de l'agent, ou externes, c'est-à-dire qu'elles ont une conséquence sur l'état de l'environnement. Les actions qui doivent être exécutées immédiatement sont envoyées aux actionneurs correspondants. Les actions qui doivent être exécutées plus tard sont envoyées aux actionneurs de l'environnement temporel afin d'être transformées en localisations temporelles. De cette manière, l'agent obtient un emploi du temps plus riche et plus précis. Cela peut donner lieu à l'ajout de paramètres facultatifs au niveau des localisations temporelles futures pour le stockage des informations.

4.3 Conclusion

4.3.1 Synthèse

Notre deuxième contribution que nous avons décrite dans ce chapitre est un raisonnement temporel : un raisonnement anticipatif. Son fonctionnement repose sur le modèle AGRET qui est notre première contribution. Ce raisonnement anticipatif consiste en la remise en question de la totalité ou d'une partie de l'emploi du temps de l'agent afin de l'enrichir et de l'optimiser. Cette remise en question est permise par la visibilité sur la dimension future du temps apportée par la mise en place de l'environnement temporel. En effet, dans la plupart des approches classiques d'anticipation dans les SMA, le raisonnement anticipatif prend en compte uniquement les informations sur la dimension passée et la dimension présente du temps. Dans ces types d'approches, l'agent est contraint de deviner ce qui va se passer dans le futur, car il n'a aucune information sur l'emploi du temps des autres agents. La plupart du temps, la prédiction du futur résulte de microsimulations que les agents lancent en interne. Nous avons donc proposé une amélioration de ce raisonnement anticipatif par l'intégration de l'environnement temporel qui offre une nouvelle dimension d'expression et de partage entre les agents. La nature temporelle de cette nouvelle dimension permet de capter les projets individuels de chaque agent et de les diffuser sur le collectif. L'analyse prédictive réalisée par les agents peut alors opérer par perception dans ce nouvel environnement commun, plutôt que par des calculs de simulation reproduits par chacun d'eux.

Du point de vue de son architecture, le raisonnement anticipatif que nous proposons est composé de deux modèles :

1. **Un modèle prédictif** intégrant un modèle d'action et un modèle d'environnement. Ce modèle d'action est caractérisé par des préconditions, des influences, des coûts et une correspondance avec un ou plusieurs actionneurs. Contrairement aux modèles prédictifs classiques qui consistent à lancer uniquement des microsimulations à l'intérieur de l'agent, nous proposons un modèle qui exploite la perception de l'environnement temporel en complément ou en substitution aux microsimulations. Les percepts temporels récoltés correspondent à l'état passé, présent et futur (état prévisionnel) de l'environnement temporel. La perception des environnements spatial, social, et temporel est contrainte par les horizons de perception. Ces différents percepts rentrent en jeu sur deux niveaux :

   — au niveau du modèle des actions : la perception des environnements permet de situer les actions, de vérifier les préconditions requises pour leur exécution et de les enrichir par des coûts. Ce fonctionnement peut se retrouver dans d'autres approches classiques qui exploitent également la perception du contexte spatial et social dans son modèle prédictif. Dans notre cas, nous rajoutons la perception du contexte temporel. De plus, la perception de ce contexte temporel nous permet une prise en compte non seulement des informations passées et présentes, comme ce qui se fait généralement dans la plupart des approches d'anticipation classique, mais également une prise en compte des informations futures.

   — au niveau du modèle de l'environnement, et plus précisément au niveau du modèle de l'environnement temporel : La démarche classique permettant d'obtenir un modèle prédictif de l'environnement consiste à lancer des microsimulations en accéléré à l'intérieur de chaque agent. Contrairement à cela, dans notre approche, l'état prévisionnel de l'environnement temporel est directement obtenu par perception d'une partie de l'axe temporel contenu dans l'environnement temporel. Cet état prévisionnel est d'autant plus intéressant en ce qui concerne sa précision, car il permet de raisonner sur des informations réelles partagées, au lieu de se baser sur des hypothèses. C'est-à-dire que l'agent n'essaye plus de deviner ce que les autres prévoient de faire comme ce qui est le cas dans la plupart des approches. Dans l'approche que nous proposons, l'agent perçoit les projets individuels des autres agents qui les diffusent sur le collectif, à travers l'environnement temporel.

2. **Un modèle de décision** qui effectue une sélection de plans sur la base de critères de minimisation des coûts associés aux déclinaisons de plans générés par le modèle prédictif.

D'un point de vue plus général, notre raisonnement anticipatif se base sur une participation proactive des agents au niveau collectif du système. Cette participation se retrouve sur deux aspects :

— Au niveau du partage des informations temporelles, de manière individuelle, par les agents par le biais de l'environnement temporel. Ce partage se fait au bénéfice du collectif ;
— Au niveau de la prise en compte des informations partagées par le collectif au niveau du raisonnement individuel de chaque agent pour satisfaire à des objectifs individuels et collectifs.

Ce fonctionnement est en parfait accord avec un des principes de base de la ville intelligente que nous avons énoncés dans les chapitres 1 et 2 de ce manuscrit : permettre à chaque individu de s'engager en faveur des objectifs collectifs pour le développement urbain de manière proactive.

4.3.2 Limites et perspectives

Plusieurs améliorations sont envisageables au niveau de l'approche d'anticipation que nous proposons. Par exemple : l'amélioration des conditions d'arrêts des itérations, l'amélioration du système de pondération, le réglage de l'horizon temporel de perception ou encore l'amélioration de la fréquence d'anticipation que nous avons déjà explicités précédemment.

Les deux principales perspectives sur lesquelles nous aimerions nous concentrer dans la poursuite de nos travaux de recherche concernent l'exploitation des réseaux sociaux et la proposition d'un modèle formel de perception.

4.3.2.1 Un modèle formel de perception

L'exploitation de la perception des environnements, et en particulier de l'environnement temporel constitue une des originalités de l'approche que nous proposons. En effet, elle permet d'introduire la prise en compte d'une nouvelle dimension dans le raisonnement anticipatif : la dimension future du temps. Il s'agit d'une dimension qui n'a pas toujours été considérée dans la plupart des approches d'anticipation que nous rencontrons dans la littérature, car les agents n'avaient accès à aucune information sur ce que les autres agents prévoient de faire. Bien que nous avons déjà commencé à formaliser cette perception, nous pensons qu'il serait intéressant, dans la suite de nos travaux, d'élaborer un modèle formel de perception de l'environnement temporel.

4.3.2.2 L'exploitation des réseaux sociaux

Nous avons abordé dans 4.2.3.4 la problématique concernant la fréquence de déclenchement du raisonnement anticipatif. Parmi les solutions proposées, nous avons défini un système d'abonnement et de notification. Ce système fonctionne de deux façons :

— l'agent peut s'abonner à un objet de l'environnement et reçoit une notification lorsqu'un événement qui s'y rapporte a lieu ;
— un agent peut décider de notifier un ensemble d'agents à proximité spatiale, temporelle ou sociale de lui.

Nous pensons que cette fonctionnalité peut facilement s'implémenter en exploitant la dimension sociale qui est déjà présente dans le modèle AGRET. Par exemple, au lieu de stocker la liste des abonnés dans un paramètre de l'environnement temporel, nous pouvons créer un groupe social qui contient la liste des abonnés à une localisation temporelle. Ainsi, à chaque fois qu'un agent s'abonne à une localisation temporelle, il est rajouté automatiquement à ce groupe. Ce groupe permet alors aux agents d'être notifiés et d'accéder aux informations relatives à la localisation temporelle à laquelle ils sont abonnés.

Ce fonctionnement est très similaire à ce que nous pouvons rencontrer actuellement dans les réseaux sociaux où un individu peut s'abonner à une personne, une page ou une publication. Cela lui permet de recevoir les notifications correspondantes.

Par ailleurs, la notion de popularité que nous pouvons retrouver à travers les mentions "j'aime", les réactions, le nombre de vues, etc. dans les réseaux sociaux peut permettre de juger de l'importance d'une information vis-à-vis du système. Cette notion d'importance de l'information pourrait être exploitée au niveau de la pondération des coûts. Cependant, les réseaux sociaux peuvent contenir de fausses informations ou de fausses réactions. Il faudra donc être vigilant sur ce point.

Cette perspective rejoint alors celle que nous avons énoncée dans le chapitre 3 précédent. Nous pensons que la piste des réseaux sociaux est aussi intéressante à ce niveau. De plus, nous rappelons que du point de vue conceptuel et technique, l'architecture en place au niveau du modèle AGRET ainsi que l'architecture de la plateforme SimSKUAD et du modèle de simulation SkuadCityModel sur lesquels nous travaillons intègrent déjà les notions de base nécessaires à la mise en place de ce système. Nous considérons alors qu'il s'agit d'une suite logique des travaux que nous avons déjà menés jusque là.

Chapitre 5

Implémentations et mises en œuvre

Dans ce chapitre, nous abordons la mise en œuvre de nos contributions, décrites dans les chapitres 3 et 4 précédents. Nous présentons une implémentation de nos propositions sur la plateforme SimSKUAD et le modèle de simulation SkuadCityModel [78]. Nous commençons par une présentation de nos choix conceptuels et techniques. Dans 5.1.1, nous décrivons la plateforme de simulation SimSKUAD et les concepts de base qui y sont propres. Nous détaillons par la suite, dans 5.1.2, le modèle d'agent SimSKUAD. Nous abordons ensuite, dans 5.1.3, l'approche d'ordonnancement du temps, plus précisément l'implémentation de l'approche de type modèle à temporalité au niveau de la plateforme. Nous terminons cette section par une explication du fonctionnement et de l'implémentation des dimensions physiques et sociales dans SimSKUAD (cf. 5.1.4). Nous estimons que la compréhension de ces bases permet de mieux appréhender nos choix.

Une fois ces bases décrites, nous détaillons l'implémentation de l'environnement temporel dans SimSKUAD dans 5.2. Dans la section 5.3 suivante, nous montrons une utilisation de cet environnement temporel dans le cadre de la mise en œuvre d'un raisonnement anticipatif. Dans ce contexte, nous nous basons particulièrement sur une implémentation au niveau du modèle de simulation SkuadCityModel. SkuadCityModel est un modèle que nous développons au sein même de notre équipe. Bien évidemment, nous terminons le chapitre par une conclusion (5.4) où nous dressons un bilan et où nous énonçons les perspectives pour la suite de nos travaux.

5.1 Choix conceptuels et techniques

5.1.1 Présentation générale de la platefome de simulation SimSKUAD

5.1.1.1 SKUAD (Software for Ubiquitous Agent Development)

SimSKUAD est le *mode simulé* de l'outil Software Kit for Ubiquitous Agent Development (SKUAD). SKUAD est une plateforme libre, développée par notre équipe de recherche Système Collectif Adaptatif du LIM [3]. À l'origine, cette plateforme a été conçue spécialement pour des applications dans le domaine des objets connectés. Le système multi-agent dans SKUAD est particulier dans le sens où il se veut être ambiant et ubiquitaire. Le SMA est ambiant, car les agents SKUAD sont capables d'évoluer dans notre environnement physique, en temps réel. Il est également ubiquitaire puisque les agents évoluent dans cet environnement, indépendamment du substrat physique sur lequel ils s'exécutent. SKUAD est également multi plateforme. Elle peut être exécutée sur différents dispositifs matériels possédant un microprocesseur tels que les ordinateurs classiques ou les Raspberry Pi. Une version spécifique peut également s'exécuter sur les microcontrôleurs comme les puces Arduino ou ESP8266.

Comme le montre la figure 5.1, SKUAD a une architecture modulaire. Elle est composée de quatre (4) briques principales, dont les plus hautes dépendent des plus basses.

1. **AgentU** : Le module AgentU regroupe les interfaces et formalismes qui permettent la création d'agents. Par défaut, les agents peuvent interagir avec un environnement physique et/ou avec une dimension sociale. Ils interagissent à travers ces derniers. Nous en reparlons en détail dans la section 5.1.4.
2. **UDA** : Le module UDA gère la normalisation des dispositifs physiques et la communication entre eux et les agents. Pour cela, il utilise un processus d'introspection. Ces dispositifs

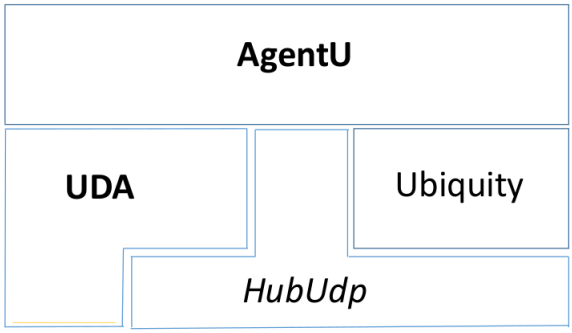


FIGURE 5.1 – Diagramme décrivant l'architecture de SKUAD.

physiques regroupent les capteurs et les actionneurs qui permettent l'interaction physique entre les agents et l'environnement physique.

Definition 5.1.1. Environnement physique. Nous désignons par environnement physique dans SKUAD tout type d'environnement permettant une interaction par câblage physique. Ce câblage physique permet à l'agent de disposer d'un ensemble de capteurs et d'actionneurs lui permettant d'interagir avec son environnement. L'environnement physique ne se limite donc pas à l'environnement spatial. Dans SimSKUAD, l'environnement temporel est aussi considéré comme un environnement physique dans le sens où les interactions entre les agents et l'environnement temporel se font par câblage physique. Nous y revenons plus en détail dans 5.2.

3. **Ubiquity** : Contrairement à UDA qui gère les interactions physiques, le module Ubiquity gère les interactions sociales entre les agents. Ces interactions prennent la forme de communications implicites ou explicites. C'est donc à ce niveau que se gère la dimension sociale dans le SMA.
4. **HubUDP** : HubUDP est la brique de plus bas niveau. Elle fait une abstraction de l'environnement physique réel, permettant les communications à travers le réseau. Il s'agit d'une librairie qui se veut zéro config. HubUDP permet également la création de réseaux superposés.

SKUAD est alors qualifié de plateforme *bimodale*. Cette bimodalité se traduit par la capacité des agents de SKUAD à s'exécuter dans un contexte réel ou dans un contexte simulé. Du point de vue pratique, cela signifie que l'environnement physique dans SKUAD peut être un environnement réel ou un environnement simulé.

5.1.1.2 SimSKUAD

Le mode simulé de SKUAD est appelé SimSKUAD. Dans SimSKUAD les agents opèrent dans une dimension virtuelle du temps. Les dispositifs matériels font partie de l'environnement spatial virtuel. Les agents prennent connaissance de cet environnement à travers des données reçues par les capteurs et par la suite, agissent grâce aux actionneurs. L'approche de gestion du temps par défaut est de type modèle à temporalité. Le modèle d'interaction quant à lui est de type influence-réaction. Cela justifie notre choix de cette plateforme de simulation pour l'implémentation de notre solution. Néanmoins, comme mentionné dans 2.3.2, les approches que nous proposons sont assez génériques pour pouvoir être implémentées sur des modèles et des plateformes utilisant d'autres types d'approches d'ordonnancement et d'autres modèles d'interactions. Cependant, pour des raisons de performance, nous préconisons l'utilisation d'une approche d'ordonnancement de type modèle à temporalité et d'un modèle d'interaction de type influence-réaction.

Dans notre contexte, l'utilisation de SKUAD (et donc de SimSKUAD) se fait sous la forme d'une librairie Java. Nous pouvons voir sur la figure 5.2, un diagramme de classe relatif au pilotage du modèle de simulation. Ce pilotage est géré par la classe **SimAU** qui implémente l'interface **SimPilot**. Cette interface contient les constantes relatives à l'état des processus et des éléments observables de la simulation. Elle contient également la signature des méthodes de lancement, de réinitialisation, de suspension et d'arrêt de la simulation.

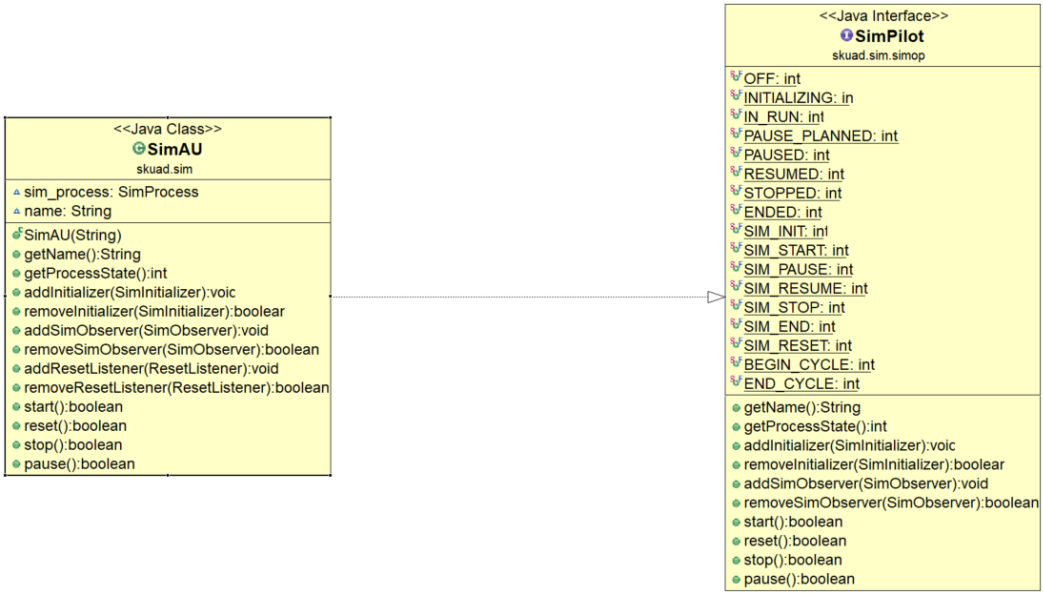


FIGURE 5.2 – Pilotage du modèle de simulation.

5.1.2 Modèle d'agent dans SimSKUAD

Un agent SimSKUAD est un objet Java qui implémente l'interface AgentU. La figure 5.3 montre un diagramme UML partiel, présentant les classes en lien avec la définition et la manipulation des agents dans SimSKUAD. Ce diagramme est partiel dans le sens où seuls quelques méthodes et attributs sélectionnés sont affichés. Ces attributs et méthodes sont ceux que nous considérons comme essentiels à la compréhension du fonctionnement du modèle.

1. **AgentU** Un agent SimSKUAD se définit à l'aide d'une classe qui implémente directement ou indirectement l'interface AgentU. AgentU définit toutes les méthodes dont le système SKUAD a besoin pour manipuler en interne les objets agents. L'interface AgentU comporte exactement 14 méthodes. Voici leur signature :

```
1  //invoqué au démarrage de l'agent
2  public void start(AgentUDescriptor aud, int ctx);
3  //invoqué lors de la mise à l'arrêt de l'agent
4  public void stop(AgentUDescriptor aud, int ctx);
5  //évènement subit par l'agent
6  public void event(AgentUDescriptor aud, int code, int ctx);
7
8  //Exécution du comportement de l'agent
9  public void behavior(AgentUDescriptor aud, Pulse pulse);
10
11 //évènement d'attachement de device (dimension physique)
12 public void notifyPlugDevice(AgentUDescriptor aud, String slot, Device device);
13 public void notifyUnplugDevice(AgentUDescriptor aud, String slot);
14 public void notifyChangeDevice(AgentUDescriptor aud, String slot, Device device);
15 //écoute des devices
16 public void stateDevice(AgentUDescriptor aud, String slot, Device dev,
17 boolean ready, boolean config_changed);
18 public void sensorChange(AgentUDescriptor aud, String slot, int sensor_id,
19 Value value);
20 //Réponse aux lectures des capteurs
21 public void sensorReading(AgentUDescriptor aud, String query_label, Value value);
22
23 //Avatars associés à l'agent (dimension sociale)
24 public void notifyPlugAvatar(AgentUDescriptor aud, String name, Avatar avatar,
```

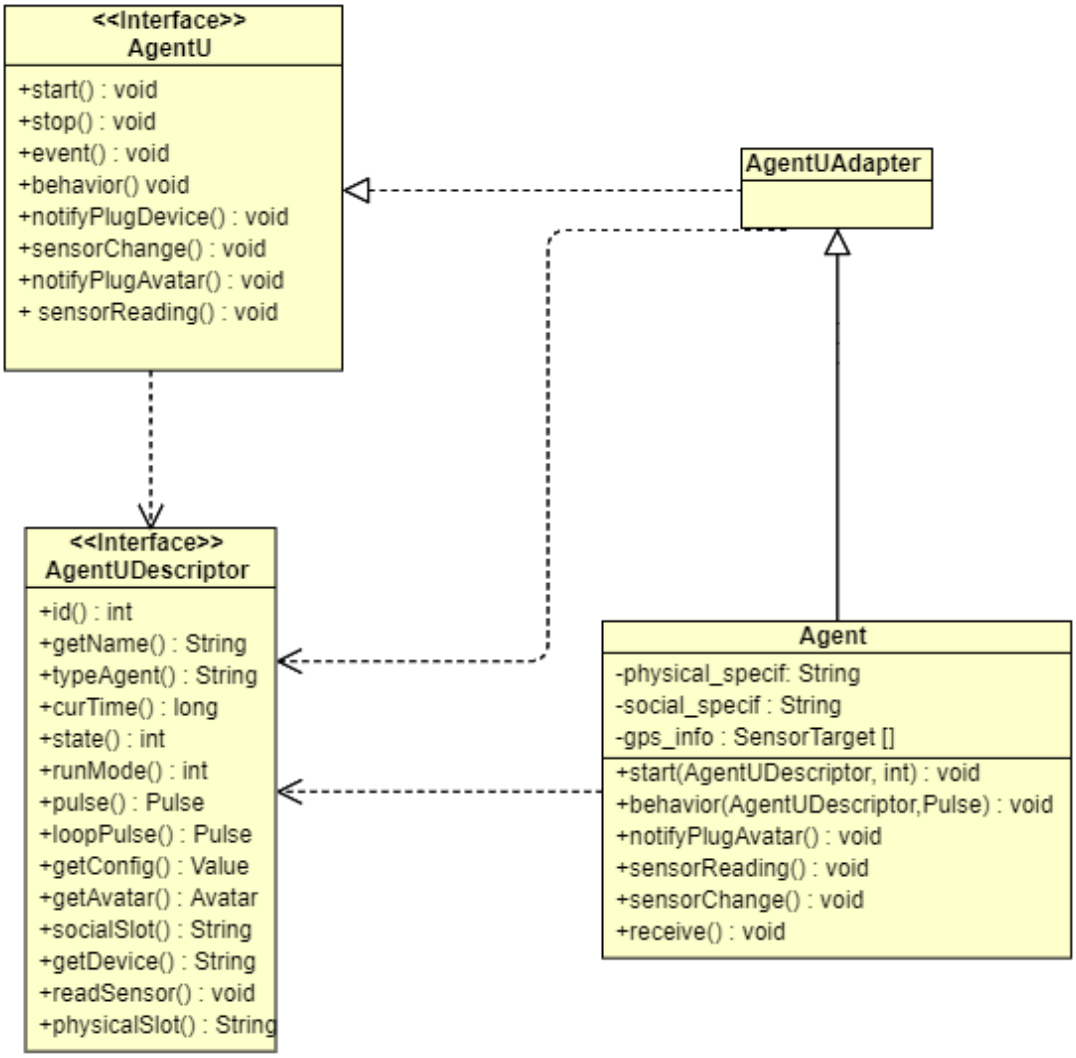


FIGURE 5.3 – Un agent SimSKUAD.

```
25  int contexte);
26  public void beforeUnplugAvatar(AgentUDescriptor aud, String name, Avatar avatar,
27  int contexte);
28  public void notifyUnplugAvatar(AgentUDescriptor aud, String name, int contexte);
29  public void receive(AgentUDescriptor aud, Avatar avatar, MailBox box,
30  Message mess);
```

Les lignes 1 à 6 regroupent les méthodes permettant la gestion du cycle de vie des agents. Comme dans le modèle à temporalité, la méthode *start*() est appelée au démarrage de la simulation afin que chaque agent puisse définir sa dynamique d'activation temporelle initiale par le biais de la définition de localisations temporelles. Dans le cas où l'agent n'en définit pas, un pas de temps par défaut lui est attribué pour rythmer son activation. Bien évidemment, cette dynamique peut être modifiée par l'agent lui-même à tout moment de la simulation.

La méthode *behavior*() dont la signature est affichée à la ligne 9 est celle qui gère l'exécution du comportement de l'agent. C'est la méthode qui est appelée pour enclencher le cycle d'activation de l'agent.

Les lignes 11 à 19 regroupent les méthodes relatives à la gestion des dispositifs matériels attachés à l'agent. Parmi eux, les lignes 11 à 14 gèrent les évènements d'attachement et de détachement du dispositif matériel. L'écoute se gère au niveau des méthodes de la ligne 16 et 18. La méthode *sensorReading*() (ligne 21) gère la réponse à la lecture des capteurs. Enfin,

les lignes 22 à 25 montrent les méthodes de gestion de la dimension sociale, c'est-à-dire à la gestion des avatars associés à l'agent.

2. **AgentUAdapter** Cette classe est un adaptateur pour l'interface AgentU. Cela voudrait dire qu'elle contient des implémentations concrètes des méthodes dont les signatures sont définies dans AgentU. Ainsi, dans la pratique, pour définir une classe agent SimSKUAD il suffit d'hériter de la classe AgentUAdapter.

3. **AgentUDescriptor** Pour chaque agent instancié, le système SKUAD crée spécifiquement un objet défini par l'interface AgentUDescriptor. Cette interface correspond au descripteur interne de l'agent. Ce descripteur procure à l'agent un ensemble de traitements qu'il peut déclencher. Il est transmis à l'agent en premier argument de chacune des méthodes de l'interface AgentU défini précédemment. De cette façon, nous nous assurons que ce descripteur est disponible dans chacun des espaces de code de l'agent.

   L'interface AgentUDescriptor contient globalement cinq (5) groupes de méthodes :

```
1  public int id(); //identifiant de l'agent
2  public String getName(); //nom affecté à l'agent
3  public String typeAgent(); //type de l'agent
4  public int state(); //état courant de l'agent
5  public int runMode(); //mode d'exécution de l'agent
6
7  public void log(String mess);
8
9  //pulse ponctuel
10 public Pulse pulse(String label, long delay, boolean fire_uptodate);
11 //pulse périodique
12 public Pulse loopPulse(String label, long delay, long period,
13                         boolean reset_on_resume);
14
15 public Avatar getAvatar(String slot_name);
16 public String socialSlot(Avatar a);
17 public boolean isSocialPlugged(String slot_name);
18 public Iterable socialsSpecif();
19
20 public Device getDevice(String slot_name);
21 public void readSensor(String query_label, SensorTarget[] targets);
22 public String physicalSlot(Device d);
23 public boolean isPhysicalPlugged(String slot_name);
24 public Iterable physicalsSpecif();
25 public boolean isReady(String slot);
26 public int trueType(String slot);
```

   Le premier groupe (ligne 1 à 5) rassemble les méthodes de consultation des informations relatives à l'agent. Le deuxième groupe comprend uniquement la méthode *log*() qui comme son nom l'indique gère les messages de log. Le troisième groupe (lignes 9 à 11) gère la rythmique de comportement de l'agent. Le quatrième groupe (lignes 15 à 18) gère la liaison des agents avec la dimension sociale (slots sociaux). Le dernier groupe (lignes 20 à 26) gère la liaison avec la dimension physique (slots physiques).

5.1.3 Ordonnancement du temps : Mise en œuvre du modèle à temporalité dans SimSKUAD

SimSKUAD utilise une approche d'ordonnancement de type modèle à temporalité. Le fonctionnement de cette approche est décrit théoriquement dans 2.2.2.2 et 3.2 des chapitres 2 et 3 de ce manuscrit.

La figure 5.4 montre un diagramme de classe partiel correspondant à l'implémentation de l'approche d'ordonnancement de type modèle à temporalité dans SimSKUAD.

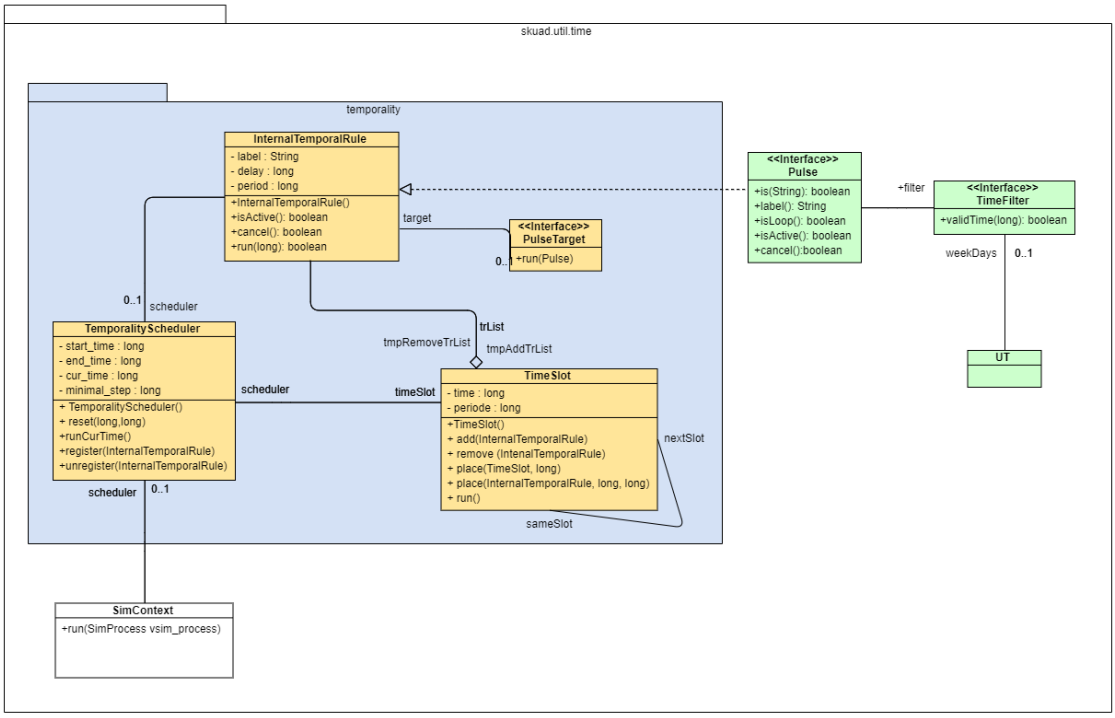


FIGURE 5.4 – Implémentation du modèle à temporalité dans SimSKUAD.

1. **InternalTemporalRule.** Cette classe gère les traitements des temporalités. Elle contient notamment la méthode *run()* qui permet l'activation d'une temporalité. Elle contient également les différentes méthodes permettant sa manipulation et la consultation de la valeur de chacun de ses paramètres.

   Pour ne pas porter confusion, nous tenons à noter que la notion de temporalité évoquée dans cette section est celle définie à l'origine dans l'approche de type modèle à temporalité dans 3.2. À titre de rappel, une temporalité est l'expression des besoins d'un agent en termes d'activation. Elle est définie par le tuple $t = \{id, d, f, p, v\}$. Il est important de ne pas confondre cette notion avec la notion de temporalité au niveau de l'environnement temporel. En effet, au niveau de l'environnement temporel, une temporalité est la représentation de l'agent au niveau de l'environnement temporel.

2. **TimeSlot.** Comme son nom l'indique, cette classe regroupe les opérations relatives à la gestion des slots temporels. Ces opérations peuvent être par exemple le rattachement ou le détachement d'une temporalité à un slot temporel, l'exécution de toutes les temporalités situées sur un même slot, le calcul du prochain slot de positionnement et le repositionnement d'une temporalité, etc. La gestion des slots temporels est implémentée sous forme de listes chaînées. Ces listes permettent de gérer l'écoulement du temps simulé et de déterminer les agents à activer à chaque avancement du temps. Cette forme d'implémentation présente différents avantages notamment en matière de performances. Cela a été démontré dans [71] et [78].

3. **TemporalityScheduler.** C'est la classe qui gère l'axe temporel de la simulation. Elle contient les méthodes qui permettent d'initialiser ou de réinitialiser l'ordonnanceur de la simulation, d'enregistrer ou de désenregistrer une temporalité, etc.

4. **Pulse** Cette classe contient le descripteur d'une unité d'activation temporelle (temporalité). Du point de vue d'un agent, la création d'une temporalité se fait par la création d'un objet de type Pulse. Un Pulse est un objet qui représente la configuration d'une rythmique comportementale d'un agent. Il peut être ponctuel ou périodique. Comme mentionné précédemment dans 5.1.2, les traitements réalisés par l'agent sont définis au niveau de la méthode de nom **behavior** de l'interface AgentU. Un pulse spécifie alors le, ou les moments sur lesquels la méthode behavior *behavior*() d'un agent doit être déclenchée. Le déclenchement

de cette méthode indique à l'agent qu'il est temps pour lui d'activer son cycle comportemental (perception - décision - influence). Lors du déclenchement de la méthode behavior, l'agent peut identifier le pulse qui a conduit à ce réveil. Cela lui permet de reconnaître le comportement qu'il a prévu d'exécuter lorsqu'il a créé ce pulse.

5. **TimeFilter et UT.** Ces classes contiennent les utilitaires nécessaires à la gestion du temps. Par exemple : les constantes et les méthodes permettant de récupérer le jour de la semaine, le temps restant pour atteindre un prochain horaire spécifié, etc.

6. **SimContext.** Cette classe gère le cycle d'exécution du simulateur expliqué dans 3.3.2. Ce cycle d'exécution consiste en :
   — Une phase d'activation de l'environnement durant lequel le simulateur doit traiter les conflits de simultanéité des actions du cycle précédent, mettre à jour l'horloge virtuelle, initialiser le prochain cycle, etc.
   — Une phase d'activation des agents durant laquelle l'agent applique le processus itératif : perception, mémorisation, décision.

   Le traitement de l'exécution de la simulation se fait par appel de la méthode $run(SimProcess\ vsim\_process)$

5.1.4 Modélisation de l'environnement dans SimSKUAD

SKUAD distingue deux dimensions de l'environnement : la dimension physique et la dimension sociale. Dans SimSKUAD, l'agent est représenté dans l'environnement physique par son **corps** (body). Le corps d'un agent est un objet de l'environnement physique. Si nous regardons la figure 5.6, la classe correspondante est la classe **BodyObject**.

Dans la dimension sociale, l'agent est représenté par son **participant**. Contrairement à la représentation de l'agent dans l'environnement physique qui nécessite la création d'une classe particulière, la création de la représentation de l'agent dans la dimension sociale se fait *automatiquement* une fois l'agent câblé à son environnement social. Dans l'absolu, les instances de la classe participant ne sont pas modifiables. Ainsi, pour permettre à l'agent propriétaire d'une de ces instances de modifier celle qui lui appartient, un objet d'écriture spécifique appelé **avatar** lui est fourni.

Les interactions entre l'agent et ces deux environnements sont spécifiées par des objets que nous appelons *slots*.

Definition 5.1.2. Slot dans SKUAD. Dans SKUAD, un slot permet le câblage entre un agent et un environnement. **Ce slot est différent du slot temporel dans le modèle à temporalité et de l'environnement temporel qui est un point de l'axe temporel**.

Dans SKUAD, un slot peut être de deux natures :
— physique : pour caractériser les capacités d'interactions physiques de l'agent ; en d'autres termes, sa faculté à opérer des actions ou des lectures sur des dispositifs matériels ;
— sociale : pour caractériser les capacités d'interactions sociales de l'agent. En d'autres termes, ces slots caractérisent sa faculté à communiquer explicitement (par message) ou implicitement (par observation), avec d'autres agents.

L'interface AgentUDescriptor décrite précédemment dans la section 3 contient les groupes de méthodes relatives à la gestion de ces slots. Le quatrième groupe de méthode (lignes 15 à 18) gère le câblage des agents avec la dimension sociale (slots sociaux). Le dernier groupe (lignes 20 à 26) gère le câblage avec la dimension physique (slots physiques).

La définition des slots quant à elle se fait au niveau de la classe de définition de l'agent (la classe qui hérite de AgentUAdapter ou d'une classe qui implémente l'interface AgentU). Le code correspondant est de la forme :

```
1 public final static String physical_specif = "..."; //slots physiques
2 public final static String social_specif = "..."; //slots sociaux
```

Nous détaillons dans la suite les particularités relatives à la dimension physique et la dimension sociale.

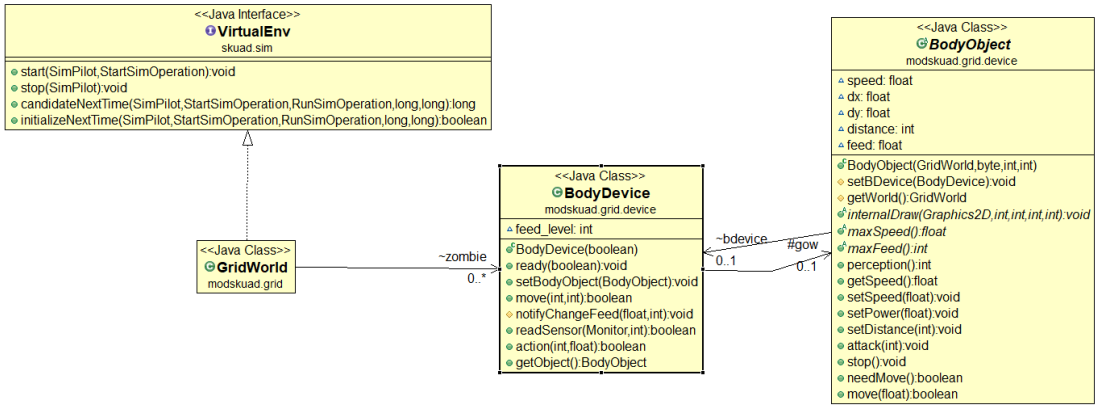


FIGURE 5.5 – Modélisation d'un environnement physique dans SimSKUAD

5.1.4.1 Environnement physique

Selon la figure 5.5, la définition d'un environnement physique dans SimSKUAD se fait par la création d'une classe qui implémente l'interface **VirtualEnv**. VirtualEnv contient la signature des méthodes permettant la gestion de l'environnement par l'ordonnanceur de la simulation.

Comme mentionné précédemment, un agent est représenté dans son environnement physique par son corps. Sur la figure 5.5 , la classe BodyObject est la classe correspondant à la définition du corps de l'agent. Ce corps est associé à des objets de l'environnement appelé *devices*.

Definition 5.1.3. device. Les devices sont des dispositifs matériels intégrant des capteurs et des actionneurs que les agents utilisent pour interagir avec un environnement physique.

Comme le montre la figure 5.6, un device se définit par une classe qui implémente l'interface **Device**. Cette interface contient la signature de toutes les méthodes de traitement permettant à l'agent d'interagir avec son environnement physique : les méthodes d'action, les méthodes de lecture des capteurs et les méthodes d'écoute.

Dans la pratique, la création d'un device peut se faire par héritage de la classe **BaseDevice** ou de la classe **MinimalDevice**. Ces deux classes implémentent l'interface Device. Elles ont l'avantage de déjà contenir les définitions des différentes méthodes de traitement d'un objet de type device.

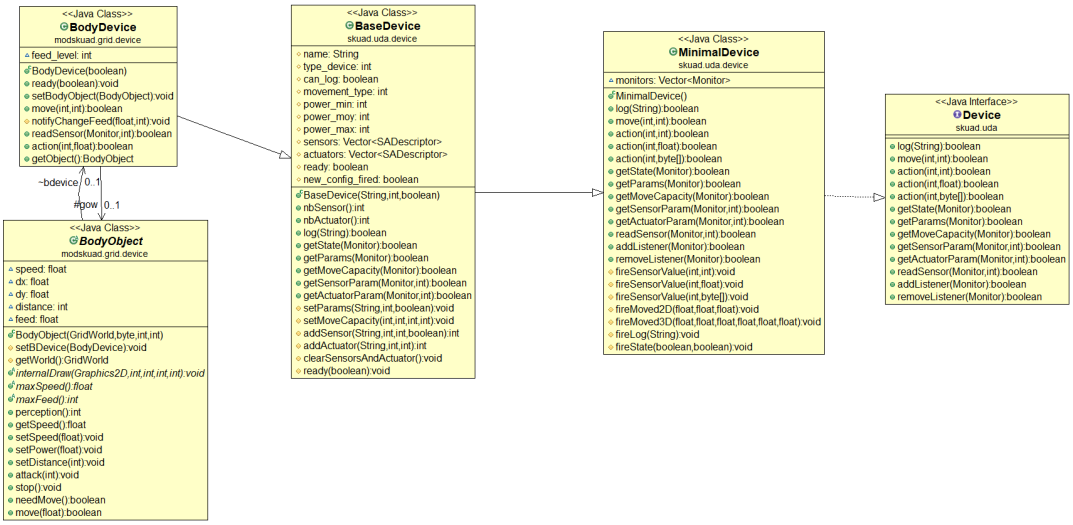


FIGURE 5.6 – La représentation de l'agent dans l'environnement physique de SimSKUAD : le device.

Lorsqu'un agent dispose d'une déclaration de slots physiques, cela signifie que durant son existence il pourra manipuler des dispositifs matériels par l'intermédiaire de ces slots. Cependant,

pour concrétiser ces possibilités de manipulations il faut également que les dispositifs existants s'associent effectivement, à un moment ou un autre, à ces slots. Nous qualifions ces associations de câblage (ou de plug en anglais). Ce câblage se fait par l'intermédiaire de l'une des méthodes **addPhysicalPlugRule()** de l'objet ServerAU qui exécute l'agent. Il existe plusieurs variantes de cette méthode. Chaque variante se distingue par le type de donnée qui permet de décrire la règle de câblage à appliquer. Cette règle définit un ensemble de champs permettant un filtrage combiné sur plusieurs niveaux : au niveau des devices, au niveau des contextes d'exécution et au niveau des nœuds réseau. Une fois qu'une règle de câblage est fixée pour l'un des slots d'un agent, le système d'exécution va garantir l'association automatique au slot correspondant dès qu'un device est compatible avec les filtres définis dans la règle.

5.1.4.2 Environnement social

Contrairement à la représentation de l'agent dans l'environnement physique qui nécessite la création d'une classe particulière, la création de la représentation de l'agent dans l'environnement social se fait automatiquement une fois le câblage des slots sociaux effectué. Ce câblage social permet à l'agent de disposer des méthodes lui permettant d'interagir avec d'autres agents notamment par le biais d'envois de messages.

Un slot social permet à l'agent d'être connecté à un espace d'interaction social. Nous appelons plus simplement ces espaces : des spaces.

Definition 5.1.4. Space. Un space est un espace d'interaction social au sein duquel l'agent peut communiquer explicitement (par envoi de message) ou implicitement (par observation) avec les autres agents connectés à ce même space.

En réalité, ces interactions ne sont pas exercées directement entre agents, mais passent par une représentation intermédiaire appelée participant.

Definition 5.1.5. Participant. Un participant est la représentation de l'agent au sein d'un space.

En d'autres termes, quand un agent envoie des messages, ou observe un autre agent, c'est son participant qui effectue ces actions sur l'instance participant de l'agent cible. Cela est en accord avec le principe de fonctionnement du modèle influence/réaction que nous avons énoncé dans le chapitre 3. L'agent est libre d'exprimer toutes les caractéristiques du participant qui le représente au sein du space. Il peut ainsi prendre l'apparence sociale qu'il souhaite.

L'agent peut être connecté à plusieurs spaces en même temps. Pour ce faire, il doit avoir plusieurs slots sociaux. Il peut alors avoir une apparence sociale différente dans chacun de ces spaces. Cependant, il ne peut naturellement pas modifier les caractéristiques des instances de participant des autres agents.

Dans l'absolu, les instances de participant ne sont pas modifiables. Ainsi, pour que l'agent propriétaire d'une de ces instances puisse modifier celle qui lui appartient, un objet d'écriture spécifique appelé avatar lui est fourni.

Definition 5.1.6. Avatar. Un avatar est un objet qui peut être vu comme un appareillage qui encapsule une instance de participant, et qui fournit des outils permettant d'opérer des modifications sur cette instance.

Ainsi, quand un agent est connecté à un space, il reçoit une instance d'avatar. Cette dernière lui permet d'agir en écriture sur l'instance de participant qui le représente au sein du space. Dans le space il peut observer les instances participants des autres agents, sans avoir la possibilité de les modifier. La figure 5.7 montre les relations entre ces différentes notions.

Un space est une abstraction permettant de faciliter les modalités de mise en relation, et les échanges d'information, entre les agents. Aussi les spaces peuvent être vus comme des réseaux sociaux similaires à ceux que nous utilisons quotidiennement : Facebook, LinkedIn, etc. En effet, les réseaux sociaux ont le même objectif : faciliter les mises en relations et les échanges d'information entre les individus. Nous sommes libres de nous représenter sous la forme que l'on veut dans un réseau social. De même, l'avatar de l'agent lui permet de choisir librement la façon dont il doit être représenté dans le space.

Dans un système SKUAD en cours d'exécution, il peut exister plusieurs spaces distincts. Chaque space est identifié par un nom. Si deux (2) spaces portent le même nom, il s'agira en réalité d'un seul et même space.

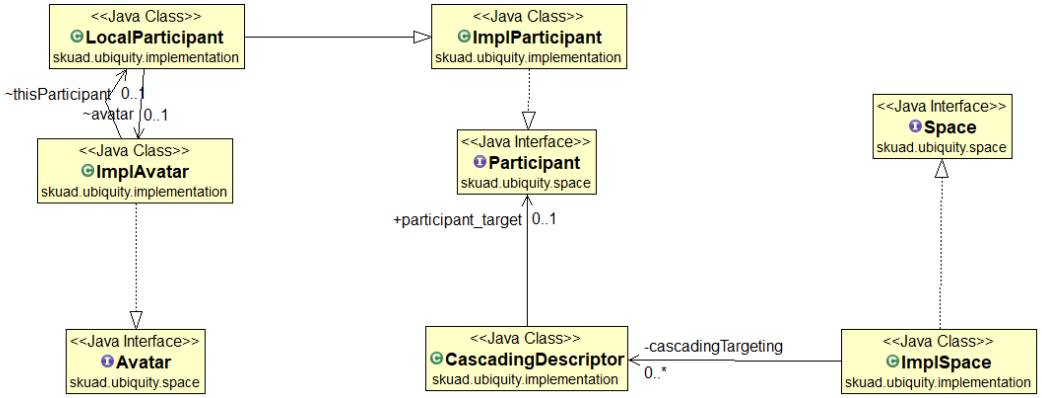


FIGURE 5.7 – La dimension sociale dans SimSKUAD

Ainsi dans SKUAD, si deux agents sont connectés au même space, et si au niveau physique il existe une connectivité réseau reliant chacun des deux contextes d'exécution de ces agents, alors ces agents se *verront* mutuellement et pourront communiquer entre eux. Dans le cas où la connectivité réseau est inexistante (ou rompue à un instant donné), ces agents seront quand même dans le même space, mais celui-ci ne sera pas (ou plus) *connexe* : il sera composé de deux îlots non interconnectés. Dans ce cas, les agents ne se verront donc pas l'un l'autre. Cependant, dès que la connectivité réseau sera (ré-)établie, alors les îlots du space seront automatiquement fusionnés et les deux agents seront (à nouveau) en mesure d'interagir entre eux.

Il est également possible d'utiliser des spaces protégés pour le cas ou les échanges portent sur des aspects critiques. Ces spaces ont la particularité d'être verrouillés par un mot de passe, et seuls les agents qui disposent du bon mot de passe pourront se connecter et interagir au sein de ces spaces. De plus, en interne, les échanges réseau concernant ces spaces sont cryptés.

Comme évoqué précédemment, le principe du câblage des slots sociaux est bien plus simple que celui des slots physiques, dans la mesure où il n'y a pas de contraintes sur l'existence d'un space. En effet, un space existe toujours, quelles que soient les circonstances. Dans le pire des cas il sera simplement non-connexe, c'est-à-dire composé de plusieurs îlots disjoints.

Le câblage effectif des slots sociaux dépend uniquement de l'état de l'agent. Lorsqu'un agent est dans un état *actif*, ses slots sociaux (si son type en définit) seront automatiquement tous câblés. Les avatars correspondants à chacun de ces slots seront alors tous instanciés. Lorsqu'il sort de l'état actif (même pour une pause temporaire), tous ses slots sociaux seront déconnectés, et donc tous ses avatars seront détruits (et les participants associés seront retirés des spaces correspondants). Au niveau du code, il n'y a donc rien de spécial à faire de plus concernant la gestion des slots sociaux de l'agent une fois qu'ils ont été définis par l'attribut $social\_specif$. Toute cette architecture déjà en place dans SKUAD montre le potentiel que peut avoir l'implémentation d'une approche basée sur les réseaux sociaux que nous avons mentionnés dans 4.3.2.2 et 3.5.2.1 et qui figure parmi la liste des perspectives intéressantes sur lesquels nous aimerions continuer à travailler dans la suite de nos recherches. Dans ce cadre, une explication plus détaillée des travaux que nous avons menés jusque là et des perspectives pour la suite est faite dans 7.2.2.

5.1.5 Le modèle de simulation SkuadCityModel

SkuadCityModel est une version améliorée et tournant sur SimSKUAD d'un modèle appelé SmartCityModel. SmartCityModel est un modèle tournant sur Repast Simphony [65], sur lequel nous travaillons en relation avec des chercheurs de l'ICL. Cette partie de nos travaux a fait l'objet de deux publications [76] et [75] dont l'une [76] en collaboration avec des chercheurs de l'ICL.

SmartCityModel a été utilisé pour différentes études de cas autour des systèmes urbains. Exemple : pour simuler les interactions entre l'utilisation des sols, les transports et la demande de recharge des véhicules électriques [10], la demande résidentielle d'électricité et de chaleur [11], pour estimer la flexibilité de la demande de recharge des véhicules électriques rechargeables [9] ou pour modéliser l'utilisation des infrastructures d'eau et d'assainissement dans les systèmes urbains [1]. Il a été le modèle original sur lequel devaient s'implémenter les contributions de cette thèse. Dans ce contexte, il a été utilisé pour simuler les flux de déplacement de transport en voitures élec-

triques, en fonction du calendrier d'activité des agents afin de mieux gérer les infrastructures de recharges. Cependant, pour des raisons d'adéquation avec les stratégies de notre équipe de recherche qui vise au développement d'une plateforme hybride et sur une perspective d'implémentation de nos contributions en environnement réel, nous avons décidé de basculer sur une implémentation sur SKUAD. Néanmoins, une implémentation partielle a déjà été effectuée sur Repast Simphony (SmartCityModel) avec un prototype qui fonctionne et qui a déjà donné de premiers résultats. Une implémentation complète des approches figure parmi nos perspectives.

Notre prototype de SkuadCityModel a également fait l'objet d'une publication [3]. SkuadCityModel est utilisé dans le cadre de différentes études comme l'étude de l'impacte des déplacements de véhicules thermiques et électriques sur la pollution ou les embouteillages. Contrairement à SmartCityModel, SkuadCityModel prend en compte, en plus des déplacements de véhicules individuels, le déplacement des bus.

5.2 Mise en œuvre de l'environnement temporel dans SkuadCityModel

Afin de permettre une portabilité et une généricité de la solution que nous proposons, nous mettons en oeuvre les mécaniques de l'environnement temporel basées sur modèle à temporalité sous la forme d'une librairie java indépendante. Cette librairie fournit un ensemble de méthodes indépendantes de SKUAD et de SkuadCityModel. Elle peut être réutilisée dans d'autres modèles et au niveau d'autres plateformes de simulation. Nous y revenons plus en détail dans 5.2.2.3. Ainsi, bien que l'implémentation de l'environnement varie selon la plateforme de simulation, l'implémentation des mécaniques environnementales au niveau du modèle à temporalité se fait par réutilisation de la librairie générique citée précédemment.

Comme évoqué dans 5.1.1, l'implémentation de l'environnement temporel dans SkuadCityModel se fait par câblage physique. La spécification du slot physique correspondant est la suivante :

```
1 public final static String physical_specif = "temporalitySlot: TEMPORALITYSLOT";
```

Dans SimSKUAD, l'environnement temporel prend alors la forme d'un environnement physique, au même titre que l'environnement spatial.

5.2.1 Fonctionnement général

5.2.1.1 Modèle et représentation de l'agent dans l'environnement temporel

Selon la description faite dans 3.1, l'agent est représenté dans son environnement temporel par sa temporalité. Comme nous l'avons expliqué dans 5.1.4.1, un agent est représenté dans un environnement physique dans SimSKUAD par son device et son corps. Ainsi, dans SkuadCityModel, la temporalité d'un agent est décrite par un ensemble de deux entités :

- le device temporel (**TemporalityDevice**) : il s'agit d'un dispositif matériel qui intègre les capteurs et les actionneurs permettant à l'agent d'interagir avec l'environnement temporel ;
- le corps temporel (**TemporalityObject**) : qui traduit la présence physique de l'agent dans l'environnement temporel.

Ces deux entités sont complémentaires et liées dans le sens où l'une est indispensable à l'autre.

Fenêtre temporelle ou horizon temporel de perception. Comme décrit dans 3.3.3, la perception d'un agent de son environnement temporel est contrainte par son horizon temporel de perception. Dans SkuadCityModel, cet horizon temporel de perception est géré au niveau des entités constituant la temporalité de l'agent. Les valeurs de ses bornes sont définies au niveau du corps temporel de l'agent, par deux attributs :

```
1 float min_horizon; //Borne inférieure de l'horizon de perception
2 float max_horizon; //Borne supérieure
```

En fonction de leur besoin, ces valeurs peuvent être modifiées par l'agent en utilisant l'actionneur correspondant. Cet actionneur produit des influences internes, c'est-à-dire des influences qui visent à modifier l'état interne de l'agent. Dans notre cas, l'objectif est de modifier la valeur de l'horizon temporel de perception. Cet actionneur est intégré au device temporel. Voici un morceau de code correspondant :

```
public final static int ACTUATOR_MODIFYHORIZON= 4;
...
public boolean action(int actuator_id, float minHorizon, float maxHorizon) {
    if(actuator_id==ACTUATOR_MODIFYHORIZON) {
  temporalityObject.setHorizon(minHorizon, maxHorizon);
  return true;
 }
```

setHorizon(minHorizon, maxHorizon) modifie la valeur des attributs **min_horizon** et **max_horizon** du corps temporel (temporalityObject) de l'agent.

Cet horizon temporel contraint la perception de l'agent. Cette perception est gérée au niveau du device temporel par le capteur correspondant :

```
public final static int SENSOR_SETOF_POSITIONS=1;
...

public boolean readSensor(Monitor mon, int sensor_id){
 switch(sensor_id) {
  ...
      case SENSOR_SETOF_POSITIONS:mon.sensorValue(this, sensor_id,
      temporalityObject.getsetOfInternalTemporalRules(target));
      return(true);
 }
  ...
}
```

La méthode **getsetOfInternalTemporalRules(target)** du corps temporel de l'agent retourne l'ensemble des localisations temporelles comprises dans la fenêtre temporelle de perception.

Des valeurs par défaut de l'horizon temporel de perception sont définies en début de simulation :

```
final float DEFAULT_MIN_HORIZON;
final float DEFAULT_MAX_HORIZON;
```

Ces valeurs sont arbitraires et sont définies au moment de la création du device, par passage de paramètre au constructeur du device temporel de l'agent. Ainsi, il est possible de définir un horizon temporel par défaut propre à chaque agent, à chaque type d'agents, ou commun à tous les agents du système.

Règles de visibilité/d'accessibilité. Les règles de visibilité sont un ensemble de contraintes rendant des localisations temporelles accessibles ou pas à d'autres agents. La question des règles de visibilité est abordée dans 3.3.3.5. Ces règles sont définies par les agents et sont gérées au niveau de l'environnement temporel. Il est important de noter que par respect de la propriété d'autonomie d'un agent, une localisation temporelle est modifiable uniquement par son propriétaire. Dans SkuadCityModel, la perception des localisations temporelles est contrainte par la fenêtre temporelle de l'agent et par un certain nombre d'attributs de la localisation temporelle. Dans l'usage que nous en faisons, deux attributs permettent de déterminer les règles de visibilité d'une localisation temporelle :

```
PulseTarget target;
String label;
```

L'attribut **target** permet de déterminer l'agent ayant défini la localisation temporelle. **label** contient la valeur d'étiquette de la localisation temporelle. Nous avons alors défini deux types de visibilité : **publique** et **privée**.

Par défaut, toute localisation temporelle est de visibilité *privée*. Cela signifie que seul l'agent l'ayant définie y a accès. L'attribut *target* permet de déterminer le propriétaire de la localisation temporelle, c'est-à-dire l'identité de l'agent ayant définie la localisation temporelle.

La règle de visibilité *publique* est déterminée par l'étiquette de la localisation temporelle. Dans notre exemple, nous considérons que toutes les localisations temporelles dont la valeur de l'étiquette est égale à "R" (recharger) sont publiques. Ces localisations temporelles sont définies par les

bornes de recharge électrique. Nous les rendons publiques, car les informations qu'elles contiennent servent au mécanisme d'anticipation au niveau des automobilistes. Le but est de susciter des comportements qui tendent à satisfaire aux objectifs collectifs : une meilleure gestion de l'accès aux bornes de recharge électrique. Nous abordons cet aspect plus en détail dans la sous-section 5.3.

Comme piste d'amélioration, il est tout à fait envisageable, si cela est nécessaire, de mettre en place un système de gestion de l'accessibilité aux localisations temporelles plus poussé. Ces règles d'accessibilité et de visibilité plus poussées peuvent prendre la forme d'un système de réseau social. Pour cela, nous pourrons nous baser sur les concepts de l'environnement social dans SKUAD qui sont décrites dans 5.7. Cela n'est pas l'objet de cette thèse, mais reste tout de même une piste très intéressante à explorer dans les perspectives.

La perception de l'agent est également contrainte par la fenêtre de stockage. La fenêtre temporelle de stockage est une caractéristique de l'environnement temporel. Nous en parlons dans la sous-section 5.2.1.2 suivante.

5.2.1.2 Modèle d'environnement temporel

Environnements contraints et non contraints par le temps. En tant qu'environnement physique, l'implémentation de l'environnement temporel dans SkuadCityModel se base sur la même architecture que celle d'un environnement spatial. Cette démarche est en accord avec un de nos objectifs : celui de prendre en compte le temps comme un milieu d'interaction au même titre que l'espace physique. La figure 5.8 montre un diagramme de classe de cette implémentation. Comme nous l'avons expliqué dans 5.1.4, la classe principale qui décrit un environnement physique dans SimSKUAD doit implémenter la réalisation de l'interface *VirtualEnv*. Cependant, à l'origine, dans SimSKUAD, comme dans la plupart des plateformes de simulation multi-agent, aucune distinction n'est faite concernant les environnements contraints par le temps et les environnements non contraints par le temps. En effet, la plupart des concepteurs considèrent que tous les environnements sont contraints par le temps. Ainsi, dans le cycle d'activation de la simulation, l'ordonnanceur active tous les environnements en début du cycle d'activation de la simulation.

Cependant, comme nous l'avons expliqué dans 3.3.2, la mise en place de l'environnement temporel nous fait prendre conscience de la possibilité d'existence d'au moins deux types d'environnements :

— les environnements contraints par le temps comme l'environnement spatial qui s'activent en début de cycle ;
— les environnements non contraints par le temps comme l'environnement temporel qui s'activent en fin de cycle.

Au niveau de l'implémentation, cette distinction fait évoluer le cycle classique d'activation des environnements par l'ordonnanceur de la simulation. Il faut désormais distinguer la phase d'activation des environnements contraints par le temps et la phase d'activation des environnements non contraints par le temps.

Nous distinguons alors au niveau de la plateforme de simulation SimSKUAD :

— Les **environnements contraints par le temps** qui se définissent par une classe qui implémente l'interface **VirtualEnv** ;
— Les **environnements non contraints par le temps** qui se définissent par une classe qui implémente l'interface **NonConstrainedVirtualEnv**. Cette interface contient la signature des méthodes permettant la gestion du cycle d'activation de l'environnement non contraint par le temps, par l'ordonnanceur de la simulation (activation, arrêt, mis à jour, etc.).

L'existence de ces deux types d'environnement vient bouleverser le cycle classique d'activation d'une simulation qui consiste désormais en trois étapes au lieu de deux :

1. Une phase d'activation des environnements contraints par le temps sur l'intervalle de temps $]t - dt; t]$;
2. Une phase d'activation des agents sur le slot temporel courant t ;
3. Une phase d'activation des environnements non contraints par le temps sur la considération que t vient d'être passé.

Au niveau la plateforme de simulation SimSKUAD, ce nouveau cycle est géré au niveau de la classe **SimContext**. Au niveau du modèle de simulation SkuadCityModel, la classe **TemporalWorld** est la classe principale qui implémente l'interface *NonConstrainedVirtualEnv* décrite précédemment.

Fenêtre temporelle ou horizon temporel de stockage. Le stockage des localisations temporelles au niveau de l'environnement temporel est contraint par la fenêtre temporelle de stockage. Son fonctionnement est décrit dans 3.3.4. Dans l'implémentation de l'environnement temporel dans SkuadCityModel, la fenêtre temporelle de stockage est gérée à travers deux attributs au niveau de la classe **TemporalEnvironmentManager** :

```
float minTimeStorage;
float maxTimeStorage;
```

Ces attributs sont initialisés en début de la simulation (phase d'initialisation), en tant que paramètres du constructeur de la classe. Ils sont mis à jour à chaque cycle d'activation de la simulation. Plus précisément, lors de la phase d'activation des environnements non contraints par le temps (cf 3.3.2). Cette phase se situe après le calcul de la réaction de l'environnement et avant le début du prochain cycle. Cela se fait par appel de la méthode **updateEnvironment()**. Cette méthode met également à jour l'ensemble des données stockées au niveau de l'environnement temporel.

5.2.2 Description de l'architecture

La figure 5.8 schématise un diagramme UML simplifié correspondant à l'implémentation de l'environnement temporel dans SimSKUAD. La manière dont l'environnement s'implémente peut varier d'une plateforme à une autre. Par exemple, dans Repast Simphony, un environnement est construit sur la base de Context et de projection.

Par conséquent dans ce diagramme, nous distinguons

— les classes qui relèvent du modèle de simulation SkuadCityModel. Ces classes sont contenues dans les packages de couleur bleue. Elles sont réutilisables et/ou adaptables.

— les classes qui relèvent de la plateforme de simulation SimSKUAD. Ces classes sont contenues dans le package de couleur verte.

— les classes qui sont génériques et qui peuvent être réutilisées en tant que librairie Java. Ces classes sont contenues dans le package de couleur jaune.

Nous expliquons en détail le contenu de chaque package dans les sous-sections suivantes.

5.2.2.1 Le package Agent

Le package Agent appartient au modèle de simulation. Dans notre cas, ce modèle est SkuadCityModel. Le package Agent contient toutes les classes relatives à la description et la création des agents du modèle. Dans notre exemple, ce package contient la classe **Agent**. Cette classe hérite de la classe *AgentUAdapter* décrite plus haut dans 5.1.2. *AgentUAdapter* est propre à la plateforme de simulation SimSKUAD.

La classe Agent contient la spécification des slots permettant de relier l'agent aux environnements de la simulation. Dans notre cas, l'environnement qui nous intéresse est l'environnement temporel. Dans le cas de l'environnement temporel, ce lien se fait par le moyen d'un slot physique, comme pour l'environnement spatial.

Voici un exemple de morceau de code relatif à la spécification d'un slot physique temporalitySlot de type TEMPORALITYSLOT.

```
public final static String physical_specif = "temporalitySlot: TEMPORALITYSLOT";
```

Une fois cette spécification faite et le câblage effectué (cf. 5.1.2), l'agent peut désormais interagir avec son environnement temporel par l'intermédiaire de son **device temporel** (cf. 5.2.2.2).

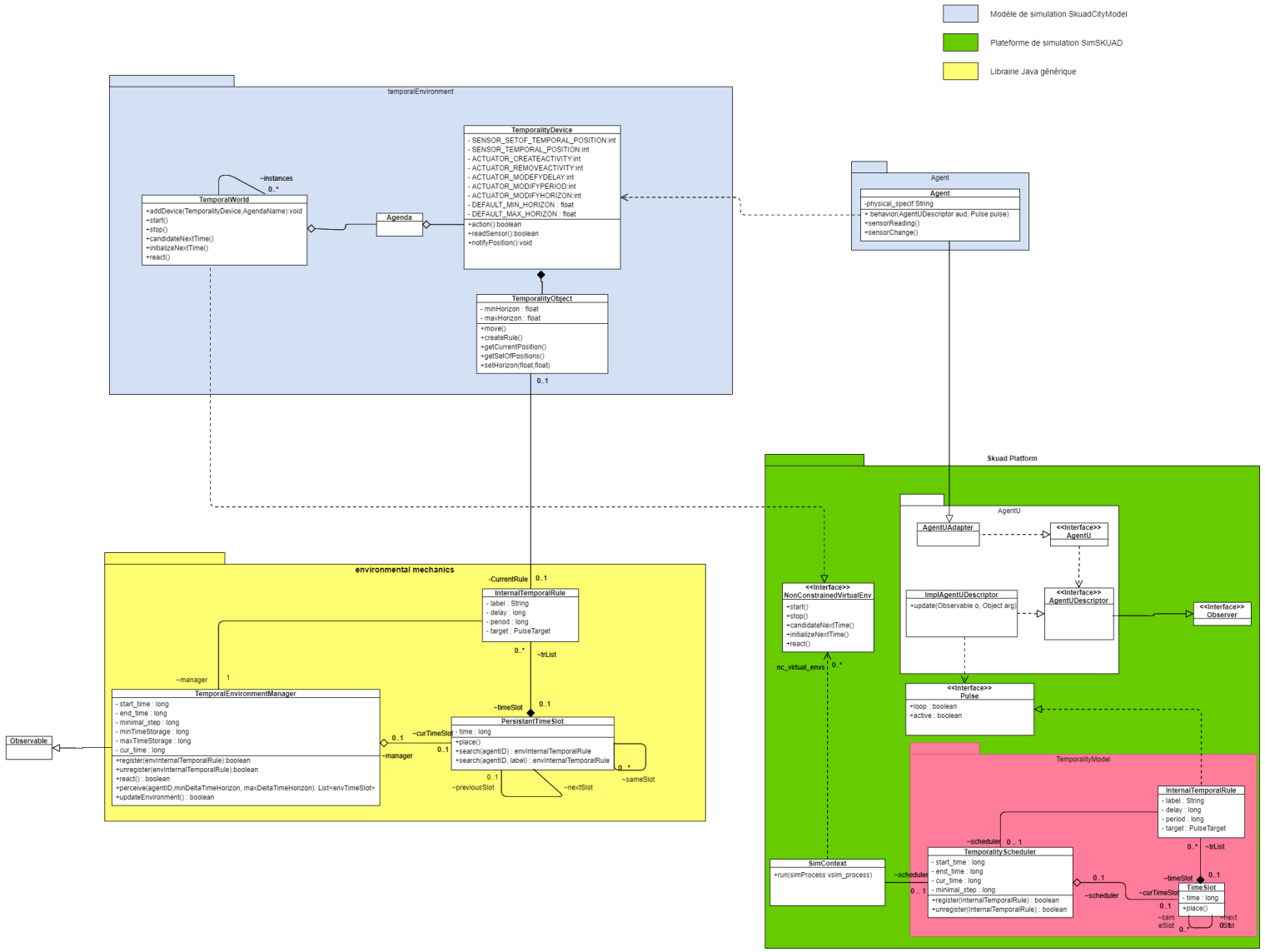


FIGURE 5.8 – Modélisation de l'environnement temporel dans SimSKUAD

5.2.2.2 Le package temporalEnvironment

Ce package fait aussi partie du modèle de simulation. Contrairement à la classe agent décrite précédemment qui est propre à SkuadCityModel, les classes contenues dans ce package sont assez génériques pour être réutilisées comme telle dans d'autres modèles de simulation développés sur la plateforme de simulation SimSKUAD. Ce package contient une partie des classes relatives à l'implémentation de l'environnement temporel proprement dite qui sont relatives au modèle AGRET : TemporalityDevice, Agenda, TemporalWorld et TemporalityObject. Ces classes et les relations qui les relient décrivent une implémentation de l'environnement temporel telle que décrite par le méta-modèle UML dans 3.1.

TemporalWorld Il s'agit de la classe contenant les descriptions du *monde* temporel. C'est la classe qui implémente l'interface *NonConstrainedVirtualEnv* de SimSKUAD comme mentionné dans 5.2.1.2. Elle contient les différentes méthodes permettant la gestion et la synchronisation de l'environnement temporel avec l'ordonnanceur de la simulation.

Agenda Cette classe décrit l'*espace*. Dans l'environnement temporel, les espaces sont des objets de type Agenda. Dans notre implémentation, un agenda est identifié par son nom.

TemporalityDevice Dans notre cas, l'environnement temporel est implémenté sous forme d'un environnement physique. L'interaction entre l'agent et l'environnement temporel se fait donc par l'intermédiaire d'un device. Un device temporel est une instance de TemporalityDevice. Cette classe décrit un dispositif matériel intégrant des capteurs et des actionneurs (cf. 5.1.4.1) ainsi que les méthodes permettant sa manipulation.

Dans une implémentation basique de l'environnement temporel, un device temporel intègre :

Des actionneurs. Les actionneurs permettent à l'agent d'essayer de créer, de supprimer ou de modifier des localisations temporelles (influences). Ils permettent également de modifier l'horizon temporel de perception.

```
//Création d'une localisation temporelle
public final static int ACTUATOR_CREATEACTIVITY= 0;

//Suppression d'une localisation temporelle
public final static int ACTUATOR_REMOVEACTIVITY= 1;

//Modification d'une localisation temporelle
public final static int ACTUATOR_MODIFY =2;

//Modification de l'horizon temporel de perception
public final static int ACTUATOR_MODIFYHORIZON= 3;
```

Des capteurs. Dans notre cas, nous définissons deux capteurs : un capteur permettant de percevoir la localisation temporelle actuelle de l'agent et un capteur retournant des percepts temporels en fonction de l'horizon temporel de perception.

```
public final static int SENSOR_TEMPORALPOSITION=0;
public final static int SENSOR_SETOF_TEMPORALPOSITIONS=1;
```

TemporalityObject Cette classe décrit le *mode* ou le corps d'un agent dans l'environnement temporel.

La classe contient les variables :

```
float min_horizon;
float max_horizon;
```

qui stockent les valeurs actuelles des bornes de l'horizon temporel de perception. Par défaut, leurs valeurs sont égales aux valeurs de *DEFAULT_MIN_HORIZON* et *DEFAULT_MAX_HORIZON*. Ces valeurs sont modifiables par l'agent en fonction de ses besoins par utilisation de l'actionneur *ACTUATOR_MODIFYHORIZON*.

La classe TemporalityObject contient également toutes les méthodes relatives à la véritable application des actions et perceptions des agents. Cela voudrait dire que lorsque l'agent agit dans l'environnement temporel, c'est son TemporalityObject qui exécute véritablement l'action (cf modèle influence/réaction 3.3). Des exemples de ces méthodes sont :

```
public void createRule(){}; //Création d'une localisation temporelle
public void removeRule(){};//Suppression d'une localisation temporelle

//Modifications d'une localisation temporelle
public void modifyDelay(){};
public void modifyPeriod(){};

//Modification de la valeur de l'horizon temporel de perception
public void setHorizon(float min, float max){};

//Méthodes pour la perception
public byte[] getCurrentInternalTemporalRule(){};
public byte[] getSetOfCurrentInternalTemporalRule(){};
```

5.2.2.3 Le package environmentalMechanics

Ce package constitue une librairie java indépendante. Les classes qui y sont contenues sont réutilisables avec d'autres modèles et plateformes de simulations. Ces classes sont relatives à la mise en place de la mécanique environnementale de l'environnement temporel : InternalTemporalRule, PersistantTimeSlot et TemporalEnvironmentManager. Comme expliqué dans 3.3.3.1, cette mécanique de l'environnement temporel se base sur la mécanique de l'approche d'ordonnancement de type modèle à temporalité. Comme nous pouvons le constater sur la figure 5.8, nous faisons le parallèle entre les classes qui décrivent la mécanique du modèle à temporalité, implémentée au niveau de la plateforme de simulation SimSKUAD (Package TemporalityModel) et les classes relatives à la mécanique de l'environnement temporel (Package EnvironmentalMechanics) au niveau du modèle de simulation SkuadCityModel. Nous tenons tout de même à noter que le contenu de ces packages n'est pas complètement identique. En effet, nous nous sommes inspirés de la mécanique du modèle à temporalité pour la mise en place de celle de l'environnement temporel. La mécanique de l'environnement temporel est gérée au niveau du modèle de simulation. Elle sert au stockage, à l'accès aux informations temporelles ainsi qu'à la dynamique propre à l'environnement temporel. En complément à cela, les mécaniques d'ordonnancement de la simulation sont gérées UNIQUEMENT au niveau de la plateforme de simulation SimSKUAD, c'est-à-dire au niveau du package TemporalityModel. Ces deux mécaniques sont complémentaires.

Nous allons maintenant décrire les différentes classes responsables de la mécanique propre à l'environnement temporel. Ces classes sont contenues dans le package *environmentalMechanics*.

TemporalEnvironmentManager. Le contenu de cette classe est inspiré de celle de la classe TemporalityScheduler décrite dans la section 3. La définition et la gestion de l'horizon temporel de stockage se font au niveau de cette classe. Les attributs correspondants sont :

```
float minTimeStorage;
float maxTimeStorage;
```

Les méthodes :

```
public boolean register(){};
public void unregister(){};
```

sont des méthodes d'influences. Elles correspondent à la demande de création et de suppression de localisations temporelles (InternalTemporalRule) au niveau de l'environnement temporel. Dans le cas d'une localisation temporelle périodique, ces méthodes prennent en compte la fenêtre temporelle de stockage pour rattacher une **empreinte** de la localisation temporelle périodique au slot correspondant.

Definition 5.2.1. Empreinte. Une empreinte est une référence vers une localisation temporelle.

Le calcul de la réaction de l'environnement se fait par appel de la méthode **react()**. Le descripteur de l'agent (AgentUDescriptor) observe cette classe afin qu'à l'issue de cette réaction de l'environnement, une fois la localisation temporelle véritablement créée ou supprimée, celui-ci procède à la création d'un *pulse*.

En fin de cycle, le système met à jour l'environnement. Cela entraîne la mise à jour de la fenêtre temporelle de stockage et des données stockées au niveau de l'environnement temporel. La méthode correspondante est la méthode **updateEnvironment()**. Elle est appelée au niveau de la classe *TemporalWorld*.

Concernant la gestion des perceptions, la méthode

```
1 public Vector<PersistantTimeSlot> perceive(float minHorizon,float maxHorizon){};
```

renvoie une liste de slots temporels en fonction de l'horizon temporel de perception et des règles de visibilité. L'agent pourra donc percevoir les localisations temporelles qui lui sont accessibles et qui sont rattachées à ces slots.

Definition 5.2.2 (Accessible.)**.** Une localisation temporelle est accessible par un agent lorsqu'elle est comprise dans sa fenêtre temporelle de perception et lorsque les règles de visibilité lui autorisent sa consultation.

PersistantTimeSlot. Cette classe est responsable de la gestion et de la création de slots temporels. Son implémentation se fait sous forme d'une liste chaînée à double sens. Ce mode de fonctionnement est différent de celui du modèle à temporalité où l'on n'a qu'une liste chaînée orientée vers un seul sens : vers le futur. Cela est dû au fait que les données passées sont immédiatement supprimées. Contrairement à cela, dans l'environnement temporel, nous stockons des données passées, présentes et futures. Les attributs **previousSlot, nextSlot, sameSlot** permettent respectivement la navigation vers le passé, le futur et au sein du même slot. Cette classe contient la méthode **place()** qui gère le rattachement d'une localisation temporelle à un slot. Elle contient également diverses méthodes de recherche **search()** qui servent lors de la phase de perception pour récupérer le ou les localisations temporelles accessibles par l'agent.

InternalTemporalRule Par défaut, cette classe est la même que celle qui se situe au niveau du modèle à temporalité. Elle correspond à la définition de la localisation temporelle de l'agent. Cependant, en fonction de la simulation et des choix du concepteur, il est possible de surcharger la classe et d'y ajouter des attributs facultatifs. C'est le cas de la longueur prévisionnelle de la file d'attente ou de la liste des abonnés que nous avons mentionné dans 4.2.3.4. Nous y reviendrons lorsque nous aborderons l'implémentation des mécanismes d'anticipation dans 5.3.

5.2.3 Exemple

Cette sous-section vient compléter les explications que nous avons faites dans les sous-sections précédentes. Nous retraçons le fonctionnement général des appels de méthodes lors d'une interaction entre l'agent et son environnement temporel. Ce fonctionnement est illustré par le diagramme de séquence schématisé par la figure 5.9. Comme pour la figure 5.8, nous distinguons ce qui relève du modèle de simulation (en bleu), ce qui relève de la plateforme de simulation (en vert) et ce qui relève de la librairie java générique (en jaune).

5.2.3.1 Phase d'activation des environnements contraints par le temps.

En début de cycle, l'ordonnanceur active les environnements contraints par le temps. Dans notre cas, il s'agit de l'environnement spatial. Pour plus d'information sur ce sujet, nous pouvons nous référer à la section 3.3.2.

5.2.3.2 Phase d'influence.

La phase d'influence commence par l'activation des agents. Cette activation s'enclenche par l'appel de la méthode **behavior()** de l'agent. Une fois réveillé, l'agent exécute son cycle perception-décision-influence.

En premier lieu, il perçoit son environnement temporel (**readSensor()**). Par le biais de cette perception, il collecte sa localisation temporelle actuelle (**getCurrentInternalTemporalRule()**).

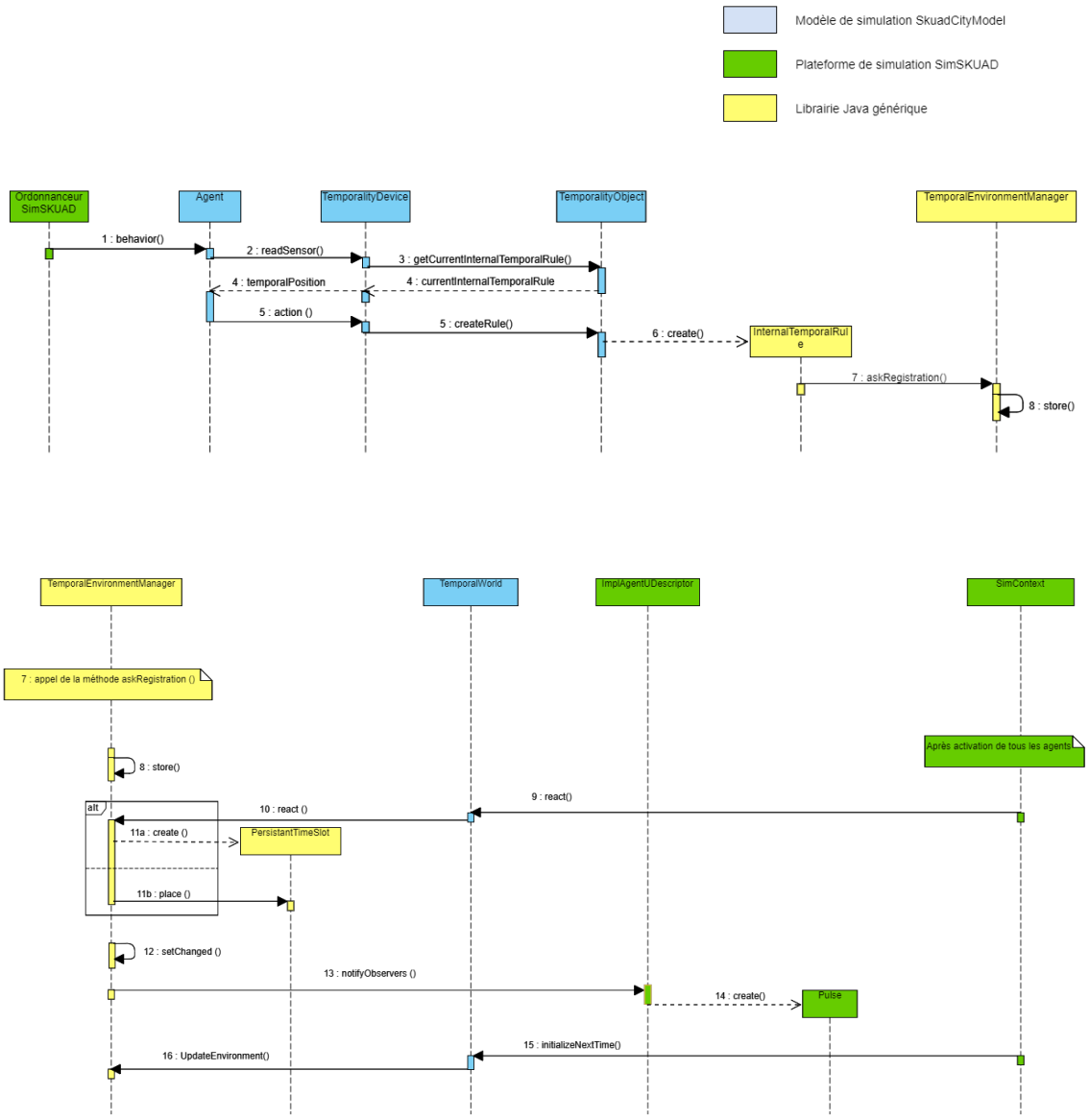


FIGURE 5.9 – Diagramme de séquence illustrant l'interaction entre un agent et l'environnement temporel.

Cela lui permet de prendre connaissance de son contexte d'activation, et donc de l'action qu'il a prévue à cet instant. Si nous faisons le parallèle avec la réalité : c'est comme si l'agent consultait son agenda. Il peut également percevoir l'ensemble des localisations temporelles auxquelles il a accès en utilisant la méthode **getSetOfCurrentInternalTemporalRule()**. Cette méthode est contrainte par l'horizon temporel de perception. Les percepts peuvent provenir de localisations temporelles passées, présentes ou futures, définies par lui-même ou partagées par d'autres agents du système.

En prenant en compte son état actuel et les percepts, il décide de l'action ou des actions qu'il doit exécuter.

Le nombre d'actions qu'il peut effectuer simultanément dépend du nombre d'actionneurs dont son device est doté et des actionneurs qu'il peut activer simultanément. Ces actions ou plutôt ces influences peuvent concerner tous types d'environnements qu'il soit spatial, social ou temporel. Dans le cas de l'environnement temporel, l'influence consiste à demander la création, la suppression ou la modification d'une ou plusieurs localisations temporelles. Ce type d'influence est appelée influences externes, car il vise à modifier l'état de l'environnement. Il peut également consister à modifier l'état interne de l'agent. Dans ce cas, il s'agit d'une influence interne. Un exemple est la modification de son horizon temporel de perception. Afin de respecter le principe de l'approche

d'interaction de type influence/réaction (cf. 3.3), ces demandes de création ou de modifications de localisations temporelles (InternalTemporalRule) sont stockées temporairement au niveau de l'environnement temporel avant d'être traitées durant la phase de réaction.

5.2.3.3 Phase de réaction.

Une fois l'activation de tous les agents terminée, l'environnement temporel procède au calcul de la réaction. Cela s'enclenche par l'appel de la méthode *react()* du TemporalWorld par le SimContext qui provoque à son tour l'appel de la méthode *react()* du TemporalEnvironmentManager. C'est durant cette phase de réaction, en tenant compte de toutes les influences et des lois environnementales (fenêtre de stockage actuelle, passé, présent, futur, etc.) que les localisations temporelles seront définitivement créées ou supprimées. Ces localisations temporelles sont par la suite rattachées aux slots temporels. Tout ce processus est géré au niveau des trois classes : TemporalEnvironment-Manager, InternalTemporalRule et PersistantTimeSlot. Ces classes sont responsables de la gestion de la mécanique propre à l'environnement temporel. C'est également à ce moment, par le moyen d'un système d'observateur (patron de conception), que le descripteur de l'agent crée ou supprime véritablement le Pulse correspondant.

5.2.3.4 Phase d'activation des environnements non contraints par le temps.

Les environnements non contraints par le temps sont mis à jour, une fois la phase de réaction terminée. Dans notre cas, la mise à jour de l'environnement temporel se fait par le biais de la classe TemporalWorld. Cette classe contient les méthodes relatives à la gestion du cycle d'activation de l'environnement temporel par l'ordonnanceur de la simulation. Cette gestion se fait au niveau de la classe SimContext qui appartient à la plateforme de simulation. En effet, avant le début du prochain cycle (en fin du cycle courant), l'ordonnanceur via la classe SimContext exécute la méthode *initialiseNextTime()* de tous les environnements non contraints par le temps. Ces derniers sont des instances des classes qui implémentent l'interface NonConstrainedEnvironnement. La méthode *initialiseNextTime()* contient les instructions permettant l'appel de la méthode *UpdateEnvironment()* de TemporalEnvironmentManager. Cela entraîne la mise à jour de l'environnement temporel. L'ordonnanceur passe ensuite au prochain cycle.

5.2.4 Synchronisation entre l'ordonnanceur et l'environnement temporel

La synchronisation des mécaniques de l'environnement temporel et ceux de l'ordonnanceur de la simulation se fait en trois étapes :

1. Après l'activation de tous les agents du système, l'ordonnanceur de la simulation, par le biais de la classe SimContext fait appel à la méthode react() de l'instance de TemporalEnvironmentManager au niveau de la classe TemporalWorld. Cela enclenche le calcul de la réaction de l'environnement temporel par le TemporalEnvironmentManager.

2. Une fois la réaction de l'environnement calculée, l'ordonnanceur de la simulation est notifiée suite à la création ou la modification d'une localisation temporelle. Cette notification se fait par le biais d'un système d'observateurs (patron de conception). Cela entraîne la création ou la modification d'un pulse au niveau de l'ordonnanceur de la simulation. Dans notre implémentation de l'environnement temporel, la classe ImplAgentUDescriptor est l'observateur (implémente l'interface observer). Cette classe se met à jour suite au changement au niveau de la classe TemporalEnvironmentManager qui est l'observable (hérite de la classe Observable).

3. Avant chaque avancement du temps dans le cycle d'activation la simulation, l'ordonnanceur met à jour l'environnement temporel (environnement non contraint par le temps). L'enclenchement de cette mise à jour est géré au niveau de la classe Simcontext qui appelle la méthode initialiseNextTime() de la classe TemporalWorld. Cette méthode contient les instructions permettant l'appel de la méthode UpdateEnvironment() de la classe TemporalEnvironmentManager au niveau de la mécanique de l'environnement temporel. L'environnement temporel se met à jour suite à cette notification.

5.3 Mise en œuvre du raisonnement anticipatif basé sur l'environnement temporel dans le cadre de SkuadCityModel

Après avoir implémenté la représentation du temps sous forme d'environnement temporel, nous présentons dans cette section une implémentation d'un raisonnement temporel qui en résulte : l'anticipation. Comme expliqué dans le chapitre 4 précédent, le raisonnement anticipatif que nous mettons en place exploite la perception des trois environnements du modèle AGRET : l'environnement spatial, l'environnement social et l'environnement temporel. Notre implémentation du raisonnement anticipatif se base sur l'exemple du rechargement des véhicules électriques avec des bornes de recharge publiques décrit dans la section 1.1.2, au tout début de ce manuscrit (chapitre 1).

5.3.1 Modèle d'environnements

Si dans la section 5.4 précédente, nous nous sommes principalement concentrés sur l'environnement temporel. Dans cette section, nous prenons en compte les trois dimensions du modèle AGRET : la dimension spatiale, la dimension sociale et la dimension temporelle. Dans SimSKUAD, l'implémentation de ces trois dimensions se classe en deux grands groupes :

— la dimension physique qui regroupe l'environnement spatial et l'environnement temporel ;
— la dimension sociale.

5.3.1.1 La dimension physique

L'environnement spatial. L'environnement spatial est implémenté sous la forme d'un environnement physique. Dans SkuadCityModel, il est défini à partir de données réelles SIG (fichiers shapes) et de statistiques. Ces données sont utilisées pour modéliser le territoire y compris les routes, les emplacements des différentes zones d'activités, des agents et des objets de l'environnement. Cet environnement est disposé en plusieurs couches. Chaque couche correspond à un ensemble de zones d'activités spécifiques (zone résidentielle, zone de travail, zone commerciale, zone de loisir, etc.). Comme décrit dans la section 5.4 précédente, l'interaction entre les agents et leur environnement spatial se fait par câblage sur un slot physique.

L'environnement temporel. L'implémentation de l'environnement temporel se fait également sous la forme d'un environnement physique. Cela a été décrit précédemment dans 5.2. L'interaction entre les agents et leur environnement temporel se fait également par câblage sur un slot physique.

Ajout de paramètres optionnels à la localisation temporelle. En fonction de la simulation et des objectifs du modélisateur, des paramètres optionnels peuvent venir s'ajouter aux paramètres obligatoires de la localisation temporelle. Dans notre scénario relatif au rechargement des véhicules électriques avec des bornes publiques, nous rajoutons trois (3) paramètres facultatifs que nous avons déjà notamment évoqués dans le chapitre 4 : l'appréciation (appreciation), la longueur prévisionnelle de la file d'attente (queue) et la liste des abonnés (subscribers). L'appréciation et la liste des abonnés sont des paramètres génériques qui peuvent être réutilisés dans n'importe quel autre modèle de simulations. Par contre, la longueur prévisionnelle de la file d'attente est un paramètre optionnel propre à la gestion de ressource partagée et limitée dans le temps et de l'espace. Sa pertinence dépend donc du contexte applicatif.

Ces paramètres facultatifs servent à renseigner le coût des actions et à améliorer l'arbitrage de la décision au niveau du raisonnement anticipatif. Par défaut, ces paramètres prennent une valeur nulle. La figure 5.10 montre la classe InternalTemporalRule avec l'ensemble de tous les attributs obligatoires et optionnels.

Appréciation (appreciation). Ce paramètre indique pour chaque action passée le niveau de satisfaction global d'un agent par rapport à l'exécution de l'action. Ce niveau de satisfaction est stocké au niveau de la localisation temporelle correspondante. Ce paramètre se renseigne uniquement sur des localisations temporelles passées. Il s'agit d'ailleurs, dans notre cas, de la seule modification qu'un agent peut effectuer sur une localisation temporelle passée. Nous avons choisi une notation simple 0 ou 1. 1 pour dire que l'action a bien été exécutée et 0 si elle n'a pas pu être exécutée.

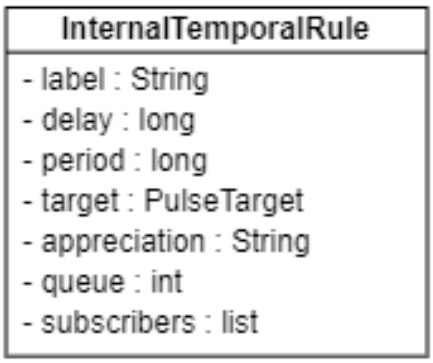


FIGURE 5.10 – La classe InternalTemporalRule

Cependant, il est tout à fait envisageable de définir un système de notation plus détaillé avec des valeurs variant entre 0 et 1.

La liste des abonnés (subscribers). Cette liste contient l'ensemble des agents qui souhaitent être informés ou notifiés des évènements relatifs à une localisation temporelle d'un agent. Si nous prenons l'exemple du rechargement de véhicules électriques avec des bornes de recharges publiques, cette liste peut contenir l'ensemble des véhicules qui prévoient de se recharger à cette borne au slot temporel t. Elle peut également inclure les agents qui aimeraient être notifiés des changements au niveau de l'état de la borne de recharge. Cette deuxième option (la prise en compte des agents autres que ceux qui prévoient de se recharger) n'est pas prise en compte dans notre implémentation, mais reste tout de même une piste intéressante à explorer.

Longueur prévisionnelle de la file d'attente (queue). Le paramètre optionnel *queue* permet de renseigner la longueur prévisionnelle de la file d'attente au niveau d'une borne. La valeur de ce paramètre n'est pas majorée. Cela voudrait dire que l'agent gestionnaire de borne ne limite pas le nombre de véhicules voulant se recharger sur la borne à un instant donné. Cependant, la capacité d'accueil d'une borne est limitée en fonction de la capacité de la borne de recharge (nombre de prises électriques et puissance de la borne) et de la valeur du pas de temps minimum (qui permet de déterminer la valeur minimum de l'espacement de deux slots temporels différents). En fonction de cette capacité, la mise à jour de la file d'attente au niveau d'une localisation temporelle pourrait affecter la longueur des files d'attente au niveau des localisations temporelles suivantes.

Par exemple : Supposons que nous ayons une borne de recharge rapide, capable de recharger un véhicule à 80% en 30 min, que la borne comporte 3 prises électriques et que le pas de temps minimum est d'une heure. Sa capacité maximale d'accueil est alors de 6 véhicules par slot temporel/par localisation temporelle, en supposant que les batteries de tous ces véhicules sont à plat à l'arrivée à la borne. Cette capacité d'accueil peut être perçue par les agents véhicules électriques par le biais de l'environnement spatial.

La borne de recharge ne fixe pas le nombre maximum de véhicules pouvant s'inscrire dans la file d'attente. C'est à chaque véhicule individuel d'estimer si s'inscrire ou rester sur cette file est avantageux pour lui ou s'il serait préférable qu'il remette en question sa décision. L'agent effectue ce choix en utilisant son mécanisme d'anticipation.

5.3.1.2 La dimension sociale

La dimension sociale est utilisée dans SkuadCityModel pour les communications explicites entre agents. Ces communications se font par envoi de message en utilisant un système de boîte mail. L'interaction entre les agents et l'environnement social se fait par câblage sur un slot social.

Voici les lignes de code correspondant aux différents slots :

```
1 //environnement spatial et environnement temporel
2 public final static String physical_specif =
3 "mobile : MOBILE;
```

```
4  temporalitySlot: TEMPORALITYSLOT";
5
6  //environnement social
7  public final static String social_specif =
8  "map : space_map";
```

5.3.2 Modèle d'agent

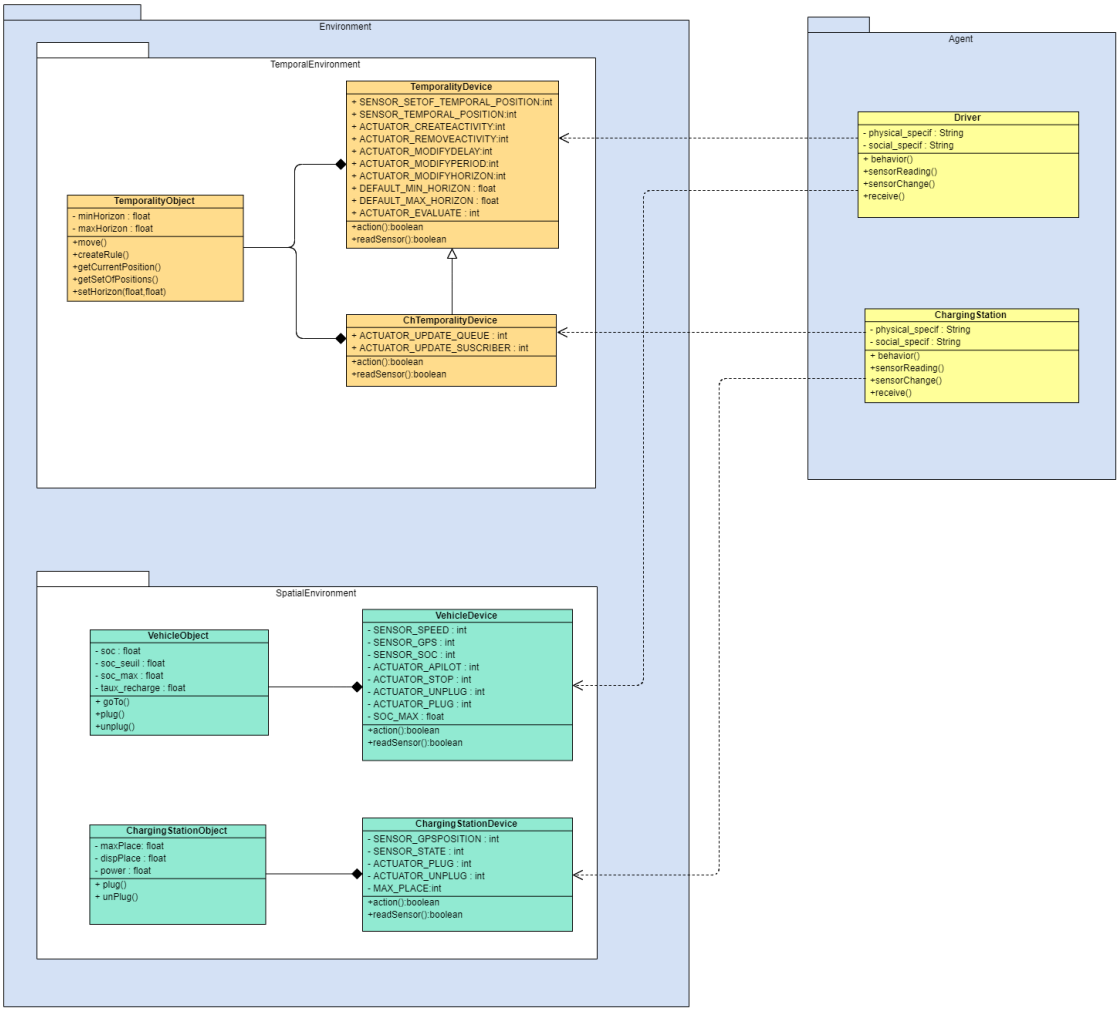


FIGURE 5.11 – Agents et Environnements Temporel et Spatial dans SkuadCityModel

Comme le montre la figure 5.11, nous distinguons, dans SkuadCityModel, deux principaux types d'agents :

1. Les agents conducteurs de voiture électrique (**Driver**) qui ont pour objectif principal d'effectuer l'ensemble des activités qu'ils ont prévu dans leur emploi du temps en utilisant leur voiture électrique. Dans ce cadre, ils sont amenés à optimiser leur rechargement de manière à ce que ce dernier ne perturbe pas leur emploi du temps.

   Un agent est représenté au niveau de l'environnement temporel par un TemporalityDevice et un TemporalityObject. En plus des actionneurs que nous avons déjà présentés dans 5.2.2.2, le device de type TemporalityDevice est doté d'un actionneur supplémentaire ACTUATOR_EVALUATE qui permet de renseigner l'attribut *appréciation* d'une localisation temporelle passée. Au niveau de l'environnement spatial, ces agents sont représentés par un VehicleDevice et un VehicleObject.

2. Les agents gestionnaires de bornes de recharge électrique (**ChargingStation**) dont l'objectif principal consiste à optimiser la répartition de l'occupation des bornes. En d'autres termes,

ils essaient d'éviter au mieux l'apparition des périodes de pic de recharge et des périodes où la borne est inoccupée. De même, au niveau de l'environnement spatial, ils essaient d'éviter les situations où certaines bornes sont vides pendant que d'autres à proximité sont sujettes à de très longues files. Ces agents sont représentés au niveau de l'environnement temporel par un ChTemporalityDevice et un TemporalityObject. La particularité de ce device de type ChTemporalityDevice est qu'il intègre deux actionneurs supplémentaires :

— Un actionneur permettant de renseigner et de mettre à jour la longueur de la file d'attente ACTUATOR_UPDATE_QUEUE ;
— Un actionneur permettant de stocker la liste des abonnés à la borne (cf 4.2.3.4) : ACTUATOR_UPDATE_SUSCRIBER.

Dans SkuadCityModel, chaque type d'agent adapte son comportement individuel de manière à satisfaire à ses objectifs.

Le raisonnement anticipatif est coûteux en temps et en ressources computationnelles. Dans notre scénario, bien que l'interaction avec l'environnement temporel soit exploitée au niveau de tous les agents, nous choisissons d'exploiter le raisonnement anticipatif uniquement au niveau des automobilistes. Par rapport à nos objectifs, ce mécanisme d'anticipation déclenché est différent selon l'activité prévue. Pour une action nécessitant un déplacement, ce qui est le cas de la plupart des actions dans SkuadCityModel, le raisonnement anticipatif se fait uniquement sur le spatial. Il s'agit du calcul de la quantité d'énergie nécessaire pour effectuer le trajet. Le raisonnement anticipatif déclenché à la phase de vérification des préconditions des actions. C'est dans le cas de l'action élémentaire *se recharger* que l'intégralité du raisonnement anticipatif comme nous l'avons décrit dans le chapitre 4 est déclenchée. Nous avons choisi ce type d'implémentation, car nous estimons que dans notre scénario le raisonnement anticipatif complet, sur les trois (3) environnements n'est indispensable que dans le cas d'un évènement qui risquerait de perturber le emploi du temps de nos agents. Ce qui est le cas de l'activité *se recharger*. Nous en parlons plus en détail dans la partie 5.3.6 suivante. Néanmoins, la mise en place d'un mécanisme d'anticipation au niveau des agents gestionnaires de bornes de recharges reste une perspective qui est facilement implémentable et qui ne nécessite pas de restructuration important au niveau du code déjà en place. Il en est de même pour le déclenchement du raisonnement anticipatif dans le cadre de toutes les autres activités de l'agent (travailler, se divertir, faire ses course, etc.).

5.3.3 Agents conducteurs de véhicules électriques (Drivers)

Les automobilistes effectuent des déplacements selon une approche basée sur les activités (activity-based). En d'autres termes, leur déplacement est piloté par les activités qu'ils souhaitent entreprendre. Comme dit précédemment, chaque agent est représenté dans l'environnement spatial par un device **VehicleDevice** et un corps **VehicleObject**. L'ensemble représente un véhicule électrique avec une autonomie de batterie limitée. L'activité de *se recharger* devient alors une contrainte.

5.3.3.1 Gestion de la batterie

Le vehicleObject contient trois attributs permettant la gestion de la batterie du véhicule : **soc, soc_seuil, soc_max**. Ces variables correspondent respectivement au niveau de la charge actuelle de la batterie, la valeur seuil en dessous de laquelle le niveau de la batterie sera considéré comme faible et la valeur maximale du niveau de la batterie. Les valeurs de soc_seuil et soc_max sont arbitraires et dépendent des caractéristiques et du type de véhicule. Elles sont paramétrables par le concepteur du modèle lors de la phase d'initialisation de la simulation, par le biais du constructeur du VehicleDevice.

Le déplacement du véhicule est géré au niveau de la méthode **goTo()** du VehicleObject. La décharge de la batterie du véhicule est également gérée à ce niveau. Une fois, le soc_seuil atteint, une notification est envoyée à l'agent pour lui signaler un niveau de batterie faible. Cela se fait par l'intermédiaire du capteur correspondant : SENSOR_BATTERY_LOW, intégré à son VehicleDevice. Il pourra par la suite adapter son comportement par rapport à cela. Nous y revenons plus en détail lorsque nous aborderons le processus d'anticipation proprement dit dans 5.3.6.

Lorsque le véhicule ne dispose plus de batterie, son corps (VehicleObject) s'arrête automatiquement (**stop()**). Voici un exemple de portion de code correspondant à cette fonctionnalité :

```
public void goTo(...){
...
    if(isElectric) {
  soc = soc - (float)dist;
  if(soc<=soc_seuil) {
   vehicleDevice.notifyBatteryLow(soc);
    if(soc<=0) {
     this.stop();
    }
   }
  }
...
}
```

La recharge de la batterie quant à elle est gérée par les méthodes **plug()** et **unPlug()** du VehicleObject. **plug()** permet de brancher le véhicule sur une borne et de le recharger. **unPlug()** le débranche. Ces méthodes se déclenchent via les actuateurs du VehicleDevice : ACTUATOR_PLUG et ACTUATOR_UNPLUG.

5.3.3.2 Groupes d'agents et activités

Comme précisé auparavant, le comportement de l'agent tend à la réalisation de son emploi du temps. Ce emploi du temps permet d'établir les localisations temporelles donc la dynamique d'activation temporelle de l'agent. Les activités d'un agent sont organisées selon un emploi du temps. Cet emploi du temps est établi sur la base d'un profil d'activité. Ce profil d'activité est défini par le modélisateur. Le profil d'activité AP_i d'un groupe d'agent i est défini par une liste de tuples :

$$AP_i = (ACT_j, MDT_j, PD_j) \tag{5.1}$$

tel que ACT_j représente l'activité j, MDT_j le temps moyen de départ (retard), et PD_j la probabilité de départ pour l'activité j.

Les automobilistes se divisent en deux grands groupes selon leur profil d'activité type : les **travailleurs** et les **non-travailleurs**. Chaque type d'agent a ses propres préférences sur toutes les actions qu'il peut adopter. Ces préférences se traduisent par des probabilités. Ces probabilités indiquent les comportements que ces types d'agents aiment adopter, et ceux qu'ils n'aiment pas, sans prendre en compte le contexte. Les deux groupes se distinguent principalement par leur tendance à aller travailler pendant les heures de travail. Par exemple : Un agent de type travailleur $a1$ possède le profil d'activité suivant $AP_{a1} = (\text{"}travailler\text{"}, 8, 0.8)$. Cela indique qu'à 8 h, il y a 80% de chance pour que cet agent choisisse l'action d'aller travailler. Contrairement à cela, un agent non travailleur $a2$ aura un profil d'activité de type $AP_{a2} = (\text{"}travailler\text{"}, 8, 0.1)$. Cela indique 10% de chance de choisir l'action d'aller travailler.

Comme dans le principe de fonctionnement de l'approche d'ordonnancement de type modèle à temporalité, la méthode **start()** permet à l'agent de définir au démarrage de la simulation sa dynamique d'activation temporelle initiale. Pour ce faire, l'agent active l'actionneur ACTUATOR_CREATE_ACTIVITY de son TemporalityDevice. Voici un exemple de morceau de code correspondant :

```
//Récupérer le TemporalityDevice correspondant
//aud est le descripteur de l'agent
TemporalityDevice tmpD = (TemporalityDevice) aud.getDevice("TemporalitySlot");

//Définir une localisation temporelle avec les paramètres période period,
//le retard delay, et l'identifiant id
tmpD.action(TemporalityDevice.ACTUATOR_CREATE_ACTIVITY, delay,period, id ,aud,
... );
```

5.3.3.3 Actionneurs et préconditions

Comme décrit dans 4.2.3.1, notre modèle d'action est caractérisé par un ensemble de paramètres dont les préconditions et les actionneurs. Le nombre d'actions qu'un agent peut réaliser simultanément est déterminé par le nombre d'actionneurs qu'intègre son device. Comme nos agents sont

situés physiquement dans un environnement spatial et un environnement temporel, ils possèdent deux devices : un pour chaque environnement. Dans l'environnement temporel, l'agent conducteur de véhicule électrique possède un TemporalityDevice comme décrit plus haut dans 5.2.2.2. Le tableau 5.1 résume les actionneurs, les actions correspondantes et les préconditions spécifiques à chaque action.

Actionneur	Action	Préconditions
ACTUATOR_CREATE_ACTIVITY	création d'une localisation temporelle	-
ACTUATOR_REMOVE_ACTIVITY	suppression d'une localisation temporelle	Existence de la localisation temporelle
ACTUATOR_MODIFY_ACTIVITY	modification des paramètres d'une localisation	Existence de la localisation temporelle
ACTUATOR_MODIFY_HORIZON	modification de l'horizon de perception	-
ACTUATOR_EVALUATE	mettre à jour la valeur du paramètre appréciation	Date de déclenchement passée

Table 5.1 – Actions correspondantes à l'environnement temporel

Le tableau 5.2 résume les actionneurs, les actions correspondantes et les préconditions spécifiques à chaque action au niveau de l'environnement spatial.

Actionneur	Action	Préconditions
ACTUATOR_PILOT	aller d'un point à une destination	Avoir assez de charges de batterie pour effectuer le trajet
ACTUATOR_START	démarrer	Être à l'arrêt.
ACTUATOR_STOP	s'arrêter	Être dans un état démarré
ACTUATOR_PLUG	brancher le véhicule	Être à proximité de la borne et que la borne soit libre
ACTUATOR_UNPLUG	débrancher le véhicule	Être branché à une borne.

Table 5.2 – Actions correspondantes à l'environnement spatial

5.3.4 Agents gestionnaires de bornes de recharge (Charging Stations)

Le deuxième type d'agent regroupe les gestionnaires de bornes de recharge électrique. Leur rôle consiste à gérer l'accès des conducteurs de véhicules aux bornes de recharge. Pour cela, chaque agent est représenté dans l'environnement spatial par un device **ChargingStationDevice** et un corps **ChargingStationObject**. L'ensemble représente une borne de recharge électrique avec un nombre limité de prises disponibles représenté par le paramètre **state**.

Au niveau de l'environnement temporel, chaque agent est représenté par une **ChTemporalityDevice** et un **TemporalityObject**.

Leur principal objectif est d'optimiser l'occupation des bornes de recharge dans le temps et dans l'espace. Comme mentionné précédemment, dans cette implémentation, afin de montrer une possibilité de mise en œuvre de notre approche, nous nous concentrons particulièrement sur les automobilistes. Les agents gestionnaires de bornes sont donc situés dans l'environnement temporel, mais n'intègrent pas de comportement anticipatif.

L'activation d'un agent gestionnaire de borne de recharge est rythmée par les localisations temporelles qu'il a prédéfinies ou par des notifications. L'agent gestionnaire de borne de recharge définit et met à jour une localisation temporelle à chaque fois qu'il reçoit une notification de souhait de rechargement de la part d'un véhicule électrique. Cela voudrait dire qu'à chaque fois qu'un véhicule prévoit de venir se recharger sur une borne de recharge, il en avertit le gestionnaire de borne correspondant. Cela permet à ce dernier, de renseigner ou de mettre à jour **la longueur prévisionnelle de la file d'attente** et **la liste des abonnées**. Un device ChTemporalityDevice intègre donc deux actionneurs en plus des actionneurs présents sur un TemporalityDevice : ACTUATOR_UPDATE_QUEUE et ACTUATOR_UPDATE_SUBSCRIBER.

Une localisation temporelle définie par une borne de recharge est de **visibilité publique**. Cela voudrait dire qu'elle est perceptible par tous les autres agents du système. Cela permet aux automobilistes de percevoir les informations nécessaires à l'arbitrage lors du processus d'anticipation. Nous y revenons plus en détail dans 5.3.6.

Le tableau 5.3 résume les actionneurs, les actions correspondantes et les préconditions spécifiques à chaque action au niveau de l'environnement spatial.

Actionneur	Action	Préconditions
ACTUATOR_PLUG	brancher le véhicule	disposer d'une prise libre.
ACTUATOR_UNPLUG	débrancher le véhicule	disposer d'au moins une prise occupée.
ACTUATOR_UPDATE_QUEUE	mettre à jour la longueur prévisionnelle de la file d'attente	-
ACTUATOR_UPDATE_SUBSCRIBER	mettre à jour le nombre d'abonnés	-

TABLE 5.3 – Actions correspondantes à l'environnement spatial

5.3.5 Dimension sociale : Notification et abonnement

La dimension sociale joue un rôle particulièrement important dans notre système. Elle permet de faire communiquer les agents entre eux. C'est à ce niveau que sont gérés les notifications et les abonnements. Cela se fait par envoi de message via le système de boîte mail qui est déjà en place.

Les agents gestionnaires de borne de recharge peuvent envoyer des notifications aux automobilistes situés à proximité, de manière régulière. Dans ce cas, son objectif est de susciter au niveau des automobilistes une remise en question de leur décision actuelle ou future. Par exemple : lorsqu'une prise électrique est libre au niveau d'une borne et que la file d'attente est vide, l'agent borne envoie une notification aux automobilistes situés à proximité spatiale. L'objectif est d'inciter ces véhicules à venir se recharger.

Voici un exemple de code correspondant à un envoi de message d'une borne vers tous les conducteurs de véhicules identifiés par "*driver". Les lignes numéro 1, 2 et 3 décrivent le contenu du message : la position géographique de la borne et le nombre de prises libre (state). La quatrième ligne est le code correspondant à l'envoi du message dont l'objet est "free". Il s'agit d'un message de notification d'une borne libre.

```
1 Value message=Value.parse("{position : { x : "+position.x+",
2                                          y : "+position.y+"},
3                                          state : "+state+"}");
4 avatar.mailBox().send("*driver", "free", message);
```

De la même manière, lorsqu'un changement se produit au niveau d'une localisation temporelle future ou de la situation actuelle d'une borne. La borne en avertit ses abonnées. Nous avons expliqué plus haut que lorsqu'un automobiliste souhaite se recharger sur une borne, il en informe l'agent en charge de la gestion de cette dernière. Le gestionnaire de borne procède alors à la création ou la modification de la localisation temporelle correspondante et inscrit l'automobiliste dans la liste des abonnés (subscribers) à cette localisation temporelle. Ainsi, à chaque fois qu'un changement s'opère au niveau de cette localisation temporelle, le gestionnaire de borne en informe les automobilistes concernés. Par exemple, lorsque le nombre de véhicules au niveau de la file d'attente change, une notification est envoyée à chacun des véhicules de la file.

Dans ce deuxième cas de figure, la remise en question de la décision par un automobiliste pourrait aboutir à :

- Ne rien faire, c'est-à-dire rester dans la file d'attente ;
- Reporter son activité *se recharger* à une date ultérieure ou au contraire avancer la date de déclenchement de l'activité ;
- Changer de borne de recharge cible, en d'autres termes, décider de se recharger sur une autre borne de recharge.

Dans les deux derniers cas de figure, l'automobiliste informe la borne de sa décision pour que celle-ci puisse mettre à jour la localisation temporelle correspondante et notifier ces abonnés de ce changement. Toutes ces notifications se font sous forme d'envoi de message par le biais de l'environnement social.

5.3.6 Mise en œuvre du raisonnement anticipatif proprement dit

Le processus d'anticipation est mis en œuvre au niveau des automobilistes. Son déclenchement se fait de deux manières :

- **Automatiquement**, durant le cycle d'activation d'un agent par l'agent lui-même.
- **Suite à une influence externe** exercée par un autre agent. Dans notre cas, il s'agit d'une notification que l'automobiliste reçoit d'un agent gestionnaire de borne de recharge. L'automobiliste peut donc suite à cette notification déclencher son cycle d'activation et donc son raisonnement anticipatif afin de remettre en question ses décisions. Par exemple, si l'agent a prévu d'aller se divertir, mais qu'il reçoit une notification selon laquelle une

borne à proximité est libre. En fonction de son état actuel et de ses percepts, il pourrait d'aller d'abord se recharger et reporter son activité de se divertir pour un instant ultérieur, dans le futur.

Nous allons présenter dans la suite de ce chapitre l'implémentation des différents blocs composant notre modèle de perception, notre modèle de prédiction et notre modèle de décision que nous avons décrit dans le chapitre 4.

5.3.6.1 Mise en oeuvre de la perception

Comme mentionné dans 5.1.2, le cycle d'activation d'un agent se déclenche suite à l'appel de la méthode **behavior()** du descripteur de l'agent par l'ordonnanceur de la simulation. La première étape du cycle d'activation d'un agent consiste à la perception de son contexte. Dans AGRET, cette perception se fait au niveau des trois environnements : spatial, temporel et social. Dans SkuadCityModel, la perception des informations au niveau des environnements physiques (spatial et temporel) se fait par le biais des capteurs intégrés dans les devices (VehicleDevice et TemporalityDevice). Voici un exemple de portion de code correspondant :

```
public boolean readSensor(Monitor mon, int sensor_id){
 switch(sensor_id) {
     // perception de la localisation spatiale
        case SENSOR_CURRENT_POSITION:mon.sensorValue(this, sensor_id,
        temporalityObject.getCurrentInternalTemporalRule());
        ...
        //perception de la localisation temporelle
        case SENSOR_SETOF_POSITIONS:mon.sensorValue(this, sensor_id,
        temporalityObject.getsetOfInternalTemporalRules(target));
        ...
    }
...
}
```

Le recueil des informations au niveau de la dimension sociale quant à lui s'effectue par lecture de message par le biais du système de boîte mail. Voici un autre exemple de portion de code correspondant :

```
public void receive(AgentUDescriptor aud, Avatar avatar, MailBox box, Message mess){
    if(mess.context().equals("found_path")) { //chemin trouvé
 Value content = mess.value();
  if(content.getBool("found")) { //un chemin a bien été trouvé
    ...

  }
  mess.consume(true);
 }
}
```

Plus particulièrement, au niveau de l'environnement temporel, comme nous l'avons expliqué dans 4.2.2, la perception permet aux agents de prendre connaissance de l'ensemble des activités qu'il a prévu d'effectuer à l'instant présent. En fonction de l'étendue de son horizon temporel de perception et des règles d'accessibilité, il peut également percevoir l'ensemble des activités que lui et les autres agents du système prévoient d'effectuer dans le futur ou/et qui ont été effectuées dans le passé. Les capteurs correspondants sont SENSOR_TEMPORAL_POSITION et SENSOR_SETOF_TEMPORAL_POSITION. Nous pouvons les voir sur la figure 5.11.

La figure 5.12 montre les attributs et méthodes de la classe permettant de définir l'agent conducteur de véhicule électrique que nous expliquons dans la suite de ce chapitre.

5.3.6.2 Mise en oeuvre du modèle prédictif

Détermination d'actions élémentaires. L'objectif de cette étape est de déterminer l'enchaînement d'actions élémentaires correspondant aux percepts reçus de l'environnement temporel.

La méthode correspondante est la méthode **getElementaryActionList()** qui prend en entrée les percepts temporels **temporalPercepts** et retourne une liste d'actions élémentaires.

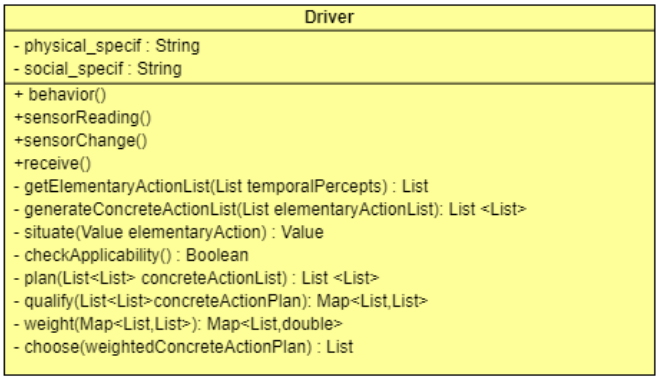


FIGURE 5.12 – L'agent conducteur de véhicule électrique

Instanciation de plans. Une fois le plan d'actions élémentaires fixé, nous passons à l'étape de la déclinaison de ces actions élémentaires en actions concrètes. Pour cela, l'agent situe chaque action élémentaire dans les environnements et vérifie les préconditions nécessaires à son exécution. À l'issue de cette étape, l'agent devrait avoir, dans l'idéal, un ensemble de plans composés chacun d'un enchaînement d'actions concrètes. La méthode correspondante est **generateConcreteActionList()** qui prend en entrée la liste d'actions élémentaires générée précédemment et retourne en sortie un ensemble de listes d'actions concrètes. Cette méthode fait appel à l'exécution de deux méthodes : **situate()** et **checkApplicability()**. L'algorithme 1 montre un pseudo-code correspondant :

Algorithme 1 : Algorithme de détermination d'actions concrètes

Entrées : *elementaryActionList* une liste d'actions élémentaires
Sorties : *concreteActionPlanList* une liste de liste de plans d'actions concrètes
Données : $nMax$ nombre maximum d'itérations

```
1  pour chaque elementaryAction de elementaryActionList faire
2  |  tempConcreteActionList(situate(elementaryAction)) ;
3  fin
4  tempConcreteActionList ← plan(tempConcreteActionList);
5  pour chaque concreteActionPlan de tempConcreteActionList faire
6  |  pour chaque concreteAction de concreteActionPlan faire
7  |  |  si checkApplicability(concreteAction) alors
8  |  |  |  tempConcreteActionPlan ← concreteAction; sinon
9  |  |  |  si ¬timeOut() || nIteration < nMax alors
10 |  |  |  |  checkForPriorAction() ;
11 |  |  |  |  nIteration++ ;
12 |  |  |  sinon
13 |  |  |  fin
14 |  |  fin
15 |  fin
16 |  concreteActionPlanList ← tempConcreteActionPlan;
```

situate () : situe une action élémentaire au niveau de l'environnement. Cette méthode prend en entrée une action élémentaire **elementaryAction** et se sert des différents percepts spatiaux, temporels et sociaux pour retourner une action concrète. Une boucle est utilisée pour passer chaque action élémentaire de la liste **elementaryActionList** à la fonction situate(). Cela permet de décliner chaque action élémentaire en une ou plusieurs actions concrètes candidates. La méthode **plan()** est appelée par la suite afin de réarranger la liste de déclinaisons d'actions concrètes candidates en une liste de plans d'actions concrètes candidates. Chaque élément de la liste est appelé *plan*. Exemple : si la liste de déclinaisons d'actions concrètes candidates est composée des éléments suivants $\{\underbrace{\{\text{“se recharger à } b_1 \text{ à } t_1\text{”, “se recharger à } b_2 \text{ à } t_1\text{”}\}}_{deux\ diffrentes\ déclinaisons\ d'une\ action\ concrète}$, “aller travailler à x à t_2”$\}$, la liste de

plans d'actions concrètes correspondante est {{“se recharger à b_1 à t_1”, “aller travailler à x à t_2”}, {“se recharger à b_2 à t_1”,“aller travailler à x à t_2”}}.

Plus particulièrement dans SkuadCityModel, au niveau spatial, chaque agent possède déjà, par défaut, un lieu de travail fixe et au moins un domicile fixe. Les percepts spatiaux sont alors utilisés uniquement dans le cas des actions élémentaires de type “se divertir”, “se recharger”, “faire les courses”, etc. Dans notre exemple, nous nous concentrons uniquement sur les comportements relatifs à l'action “se recharger”. Il s'agit, dans notre cas, de l'unique action élémentaire pouvant se décliner en plusieurs actions concrètes. Ainsi pour l'action “se divertir”, nous sélectionnons aléatoirement dès cette étape un seul lieu de loisir parmi l'ensemble des lieux de loisir que l'agent peut percevoir au niveau de l'espace.

Contrairement à cela, l'action élémentaire “se recharger” pourra être décliné en plusieurs actions concrètes candidates et seront sujettes à un choix au moment de l'arbitrage.

checkApplicability() : Cette méthode effectue la vérification des préconditions nécessaires pour l'exécution de chaque action concrète. Dans notre scénario, au niveau spatial, ces préconditions sont visibles dans le tableau 5.2.

Au niveau temporel, les préconditions consistent à vérifier que la date de déclenchement d'une action doit être présente ou future. La seule exception est pour l'action “renseigner appréciation” qui concerne uniquement une localisation temporelle passée. Ces préconditions sont visibles dans le tableau 5.1. Dans le cas où les préconditions ne sont par remplies. L'algorithme cherche une action ou un ensemble d'actions à exécuter au préalable. La méthode correspondante est **checkForPriorAction()**.

Conditions d'arrêt. La question de la condition d'arrêt du processus de vérification de l'applicabilité des actions concrètes est primordiale. En effet, les itérations sont coûteuses en temps et en ressources computationnelles. Dans des cas extrêmes où aucune précondition ne peut être satisfaite, il se peut que l'algorithme tombe sur une boucle infinie. Notre proposition consiste alors à définir un nombre maximum d'itérations et/ou un délai (time out). Ces critères d'arrêt sont définis arbitrairement par le concepteur. Il en est de même pour leur valeur par défaut. Dans notre algorithme, il s'agit de l'appel de la méthode **timeOut()** et de la vérification de la valeur de **nIteration**. La méthode fonctionne comme un minuteur et *timeout*() retourne un booléen dont la valeur est *true* au-delà du délai fixé.

Qualification des actions concrètes Cette étape consiste à renseigner l'ensemble des coûts correspondant à chaque déclinaison d'action concrète. Elle correspond à l'appel de la méthode **qualify()**. Cette méthode prend en entrée la liste de plans d'actions concrètes candidates **concreteActionPlanList** précédente et génère une collection de liste de plans d'action concrète et l'ensemble de coûts correspondants **qualifiedActionPlanCollection**.

Pour ce faire l'agent se base sur les informations qu'il peut récolter des trois dimensions : espace, temps et organisation. L'espace et le temps étant implémentés sous forme d'environnement physique, la récolte d'information à ce niveau se fait alors par le biais des capteurs intégrés aux devices. Au niveau de l'environnement social, la récolte d'informations se fait par lecture de message.

Dans notre scénario, les coûts sont :

— La criticité c : calculée à partir du paramètre appréciation récoltée par perception de l'environnement temporel (passé) ;

— Le coût correspondant à la longueur prévisionnelle de la file d'attente q' : récolté par perception de l'environnement temporel (futur) et par message au niveau de l'environnement social (interrogation de la borne de recharge) ;

— Le coût énergétique e : calculé à partir de percepts captés au niveau de l'environnement spatial.

5.3.6.3 Mise en oeuvre du modèle de décision

Une fois ces coûts renseignés, la méthode **weight()** se charge de la pondération et de la somme des coûts pondérés. Ces pondérations sont des valeurs dont le total est égal à 1. Par exemple $c = 0.25, q' = 0.5, e = 0.25$ signifie que l'agent accorde 25% d'importance aux coûts criticité et

quantité d'énergie nécessaire, tandis que la longueur prévisionnelle de la file d'attente est beaucoup plus importante (50%). Elle retourne une collection de plans d'action et la somme des coûts correspondants **weightedConcreteActionPlan**.

L'agent sélectionne par la suite le plan le plus pertinent. La méthode correspondante est la méthode **choose()**. Elle prend en entrée la liste **weightedConcreteActionPlan** précédente et retourne un plan. Le plan sélectionné est celui dont la somme des coûts pondérés est minimale. Chaque action concrète contenue dans le plan sélectionné est envoyée aux actionneurs correspondant pour être transformée en influence. La première action qui doit immédiatement être exécutée est envoyée aux actionneurs correspondants. Les actions qui doivent être exécutées plus tard sont envoyées aux actionneurs de l'environnement temporel afin qu'elles puissent être transformées en localisations temporelles. De cette manière, l'agent obtient un emploi du temps plus riche et plus précis.

La figure 5.13 montre le diagramme de classe correspondant à l'implémentation complète des agents. Cette implémentation intègre les méthodes relatives au raisonnement anticipatif dans SkuadCityModel, ainsi que de leur représentation dans l'environnement spatial et dans l'environnement temporel.

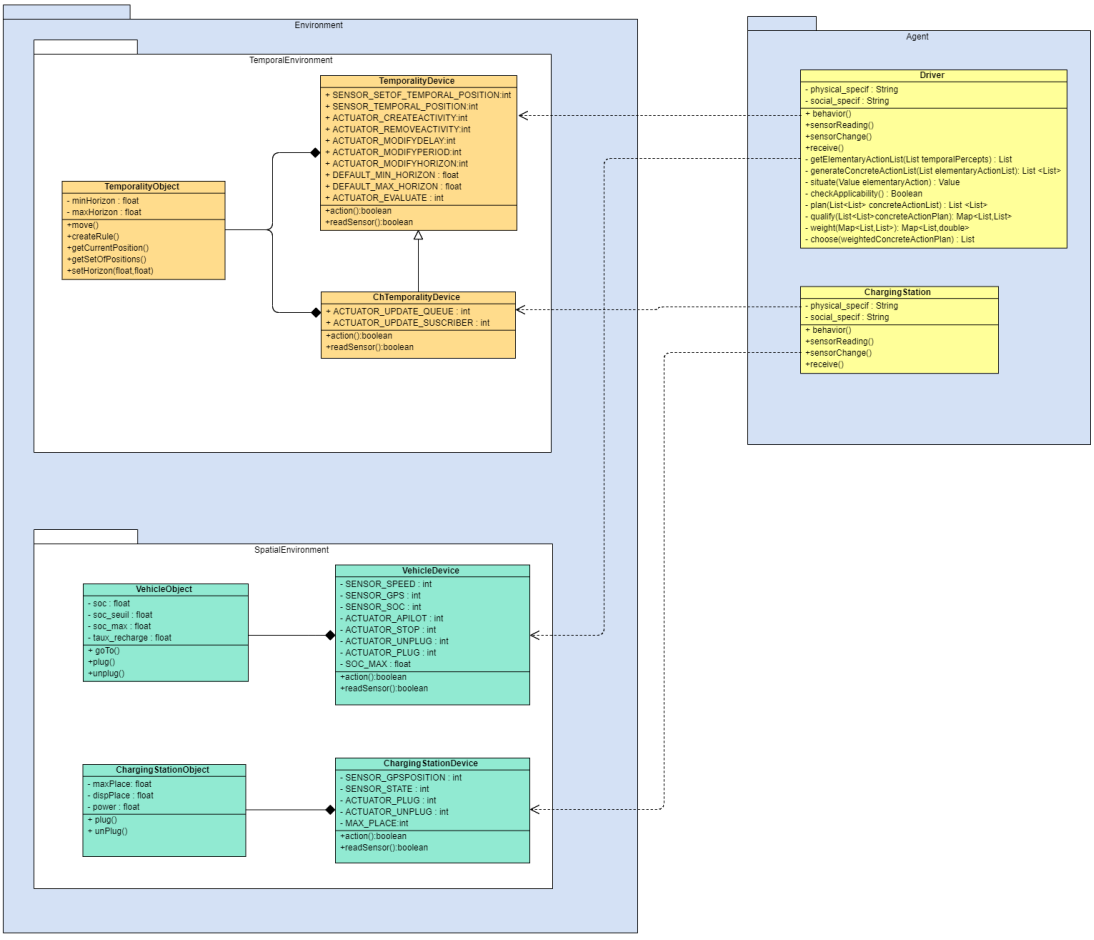


FIGURE 5.13 – Les agents dans SkuadCityModel, intégrant un raisonnement anticipatif basé sur AGRET.

5.4 Conclusion

5.4.1 Synthèse

Dans cette section, nous avons présenté une mise en oeuvre de nos contributions dans le cadre du modèle de simulation SkuadCityModel et de la plateforme de simulation SimSkuad.

5.4.1.1 Implémentation de l'environnement temporel sur SimSKUAD et SkuadCityModel

Nous avons tout d'abord réalisé une implémentation de l'environnement temporel. Cela a induit des implémentations sur trois niveaux : au niveau du modèle de simulation, au niveau de la plateforme de simulation et en tant que librairie java indépendante. Nous avons donc bien distingué ce qui relève du modèle de simulation de ce qui relève de la plateforme de simulation et de ce qui est générique.

Au niveau du modèle de simulation. L'implémentation de l'environnement varie en fonction des plateforme de simulation. La mise en oeuvre l'environnement temporel tel que décrit par le modèle AGRET est donc effectué au niveau du modèle de simulation. Cet environnement temporel prend la forme d'un environnement physique, au même titre que l'environnement spatial. Le lien entre l'agent et son environnement temporel est alors un lien physique. La temporalité est la représentation de l'agent dans son environnement temporel. Cette temporalité a été implémentée par le biais de la combinaison de deux objets : le corps de l'agent et son device temporel. Le corps est la représentation physique de l'agent dans l'environnement temporel. Le device temporel est un objet de l'environnement temporel qui intègre les capteurs et les actionneurs. Il vient en complément au corps pour attribuer à l'agent les capacités nécessaires pour qu'il puisse interagir avec l'environnement.

Au niveau de la librairie java. Les mécaniques environnementales sont quant à elles génériques. Elles sont indépendantes du modèle et de la plateforme de simulation. Nous l'implémentons donc sous la forme d'une bibliothèque java indépendante. Elle est réutilisable au niveau d'autres modèles et plateformes de simulation.

Ces mécaniques sont basées sur le modèle à temporalité. Nous y avons rajouté la gestion du stockage et de la perception ainsi que la possibilité de rajouter des paramètres optionnels à la localisation temporelle.

Au niveau de la plateforme de simulation. L'introduction de l'environnement temporel a apporté des changements au niveau de la gestion du cycle d'activation de la simulation. Ce cycle se gère au niveau de l'ordonnanceur de la plateforme de simulation. Une amélioration a donc été faite à ce niveau. Il s'agit de la déclinaison de la phase d'activation des environnements en deux phases : l'activation des environnements contraints par le temps en début de cycle et l'activation des environnements non contraints par le temps en fin de cycle.

Synchronisation entre le modèle de simulation et la plateforme de simulation. Une particularité de l'environnement temporel est qu'il détermine la dynamique d'écoulement du temps. Ainsi, à chaque cycle d'activation de la simulation, il est indispensable de synchroniser l'environnement temporel et l'ordonnanceur de la simulation. Pour ce faire, après l'activation des agents, l'environnement temporel calcule sa réaction. Cette réaction entraîne la création ou la modification de localisations temporelles. Ces localisations temporelles se situent au niveau du modèle de simulation. La mise à jour de ces localisations temporelles au niveau de l'environnement temporel (au niveau du modèle de simulation) entraîne la mise à jour des temporalités au niveau de l'ordonnanceur de la simulation (au niveau de la plateforme de simulation). Ces mises à jour consistent en la création ou la modification de temporalités. Avant de passer au cycle suivant, l'ordonnanceur procède par la suite à la mise à jour des environnements non contraints par le temps. Dans notre cas il s'agit de l'environnement temporel. Cette deuxième mise à jour permet de traiter le stockage des informations, les nouvelles valeurs de l'horizon temporel de stockage et du slot temporel courant, etc.

5.4.1.2 Implémentation du raisonnement anticipatif sur SkuadCityModel

Nous avons par la suite réalisé une implémentation de notre approche d'anticipation. Dans notre exemple, cette approche a concerné principalement les automobilistes, même si une partie du système a également été mise en place au niveau des bornes de recharges (notamment le système de

notification). Contrairement à l'implémentation de l'environnement temporel qui repose presque uniquement sur l' environnement physique, notre implémentation du raisonnement anticipatif repose sur la dimension physique et la dimension sociale dans lesquels les agents sont plongés. La dimension physique comprend l'environnement spatial et l'environnement temporel. Dans ce cadre, un ensemble d'informations spatio-temporelles est perçu au niveau de ces deux environnements via un réglage de l'horizon spatial et de l'horizon temporel de perception. Pour compléter cela, nous avons géré au niveau de la dimension sociale les systèmes de notifications et d'abonnements. Ces systèmes se basent sur de la communication explicite : envoi et lecture de message via boîte mail. Les informations échangées au niveau de la dimension sociale servent dans l'arbitrage des décisions.

5.4.2 Limites et perspectives

Ces implémentations nous ont permis de montrer concrètement la faisabilité technique de nos concepts théoriques. Nous avons notamment montré une implémentation du modèle AGRET au niveau de la plateforme de simulation SimSKUAD qui est indépendante du modèle de simulation. Elle devrait donc être facilement transférable sur d'autres modèles de simulation tournant sur SimSKUAD. Les mécaniques propres à l'environnement temporel sont quant à elles implémentées sous forme d'une librairie java indépendante. De cette manière, elle peut être facilement réutilisable avec d'autres modèles et plateformes de simulation. Des modifications ont également été effectuées au niveau de la plateforme de simulation. Il s'agit notamment de la séparation de l'activation des environnements contraints et non contraints par le temps. Nous avons par la suite montré un exemple d'implantation de notre approche d'anticipation au niveau du modèle de simulation SkuadCityModel.

Plusieurs améliorations pourraient être apportées à ce niveau. En effet, nous avons effectué une mise en oeuvre dans le cadre d'un modèle utilisant une approche d'ordonnancement de type modèle à temporalité. Il serait également intéressant, afin de montrer l'adaptabilité de l'approche, de l'implémenter sur d'autres modèles utilisant d'autres approches d'ordonnancement comme l'approche événementielle ou l'approche à pas de temps constant. Dans ce cadre, une perspective intéressante serait une implémentation sur SmartCityModel et sur la plateforme de simulation Repast Simphony. Ce sont d'ailleurs le modèle et la plateforme sur lesquelles nous avons travaillé en début de cette thèse.

D'autres pistes intéressantes consisteraient aussi à améliorer le système de notation au niveau du paramètre appréciation. Cela consisterait par exemple à utiliser des valeurs plus précises que 0 ou 1. Il est aussi possible d'améliorer le système d'abonnement en permettant à n'importe quel agent de s'abonner à un autre agent. Une autre amélioration qui pourrait également être apportée au niveau du système d'abonnement serait d'exploiter la dimension sociale. De manière très concrète, une utilisation assez simple serait par exemple de permettre à la borne de notifier d'autres agents qui sont à *proximité sociale* de lui. La mise en place d'un raisonnement d'anticipatif au niveau de tous les agents du système pourrait aussi être envisageable si cela est nécessaire.

Chapitre 6

Evaluation

Dans le chapitre 5 précédent, nous avons montré une mise en oeuvre de nos contributions sur SkuadCityModel. Dans ce cadre, nous nous sommes concentrés sur le point de vue du modélisateur et du concepteur de modèle. Pour compléter cet aspect technique, nous décrivons, dans ce chapitre, la mise en oeuvre de nos contributions, du point de vue de l'utilisateur. Pour ce faire, nous revenons sur l'exemple du rechargement de véhicules électriques avec des bornes publiques, que nous avons énoncé en début de ce manuscrit, dans 1.1.2. Ce chapitre est structuré comme suit :

Nous commençons par une présentation de l'interface utilisateur de SkuadCityModel. Nous prêtons une attention particulière aux entrées et sorties du modèle ainsi qu'à l'ensemble des fenêtres graphiques et les fonctionnalités qui peuvent s'y trouver. Ensuite, nous définissons, dans la section 6.2, notre scénario de départ. À partir de ce scénario, nous faisons ressortir les limites du système sur lesquels nous positionnons nos contributions. Ces limites rejoignent et illustrent les deux besoins que nous avons énoncés dans 1.2 en tant que problématiques principales de cette thèse. Nous présentons, dans la section 6.3, un ensemble des scénarios qui prend en compte nos contributions ainsi que les résultats correspondants. Pour le moment, nous avons effectué des tests théoriques. Néanmoins, nous prévoyons, dans les perspectives à court terme de nos travaux de recherche, d'effectuer des tests réels au niveau de la simulation. Enfin, dans la section 6.4, nous terminons, comme à chaque fin de chapitre par une synthèse et par l'énoncé des limites de nos propositions. Ces limites donnent lieu à des perspectives pour la suite de nos travaux.

6.1 Le modèle de simulation SkuadCityModel

SkuadCityModel est un modèle de simulation multi-agent qui tourne sur la plateforme de simulation SimSKUAD. Le modèle comporte différents types d'agents :

- les automobilistes qui représentent des conducteurs de véhicules thermiques, des conducteurs de véhicules électriques ou des conducteurs de bus ;
- les bornes qui gèrent l'accès aux bornes de recharge électrique ;
- les feux de signalisations qui comme leur nom l'indique gèrent les feux tricolores ;
- l'agent cartographe qui joue le rôle d'un Global Positioning System (GPS) ;
- les agents statisticiens qui se chargent du traitement et de l'affichage des données statistiques au niveau des différentes fenêtres graphiques ;
- l'agent météo qui gère l'alternance entre le jour et la nuit.

Ces différents agents se situent dans un environnement spatial qui est le reflet de l'environnement physique réel. Cet environnement est modélisé à partir de données réelles sous format SIG. Dans le cadre de cette thèse, nous rappelons que nous nous intéressons essentiellement aux conducteurs de véhicules électriques et aux bornes de recharge électrique.

6.1.1 Entrées-Sorties

SkuadCityModel prend en entrée :

- Un ensemble de fichiers SIG (format shape) qui permettent de modéliser l'environnement. Ces fichiers contiennent également un ensemble de données que nous utilisons pour le paramétrage de la simulation comme le sens de circulation ou la valeur du dénivelé de chaque portion de route ;

— Des documents de sortie de feuilles de calcul (.xls) qui contiennent des données concernant les lignes de bus, les bornes de recharge, les feux de signalisation, etc. Ces données servent au paramétrage de la simulation ;
— Des images qui sont utilisées pour l'affichage graphique des différents composants (bus, voitures, bornes, feux de signalisation, etc.) ;
— D'autres réglages peuvent être effectués, d'autres types de données peuvent également être renseignées, manuellement, par l'utilisateur par le biais de l'interface graphique de la simulation. Par exemple : le pourcentage de véhicules électrique, le pourcentage de la population d'agent à simuler, etc.

SkuadCityModel génère en sortie un ensemble de données. Ces données sont affichées sur des fenêtres graphiques. Elles peuvent également être stockées dans une base de données (MongoDB), dans un document .pdf ou feuille de calcul .xls. Le modélisateur peut choisir ou définir les données à générer en fonction du scénario qu'il souhaite implémenter et des objectifs de la simulation. Des exemples de données que nous avons actuellement en sortie sont la distance totale parcourue, le taux de pollution, l'état des bus ou les embouteillages. Des fenêtres graphiques affichent l'évolution de la simulation. L'utilisateur peut alors suivre, en temps réel, les déplacements des véhicules ainsi que l'évolution des différents paramètres de la simulation. Il a également la possibilité d'interagir avec celui-ci par le biais de ces fenêtres graphiques.

6.1.2 Interface graphique

La figure 6.1 montre une capture d'écran d'une partie de l'interface utilisateur de SkuadCityModel.

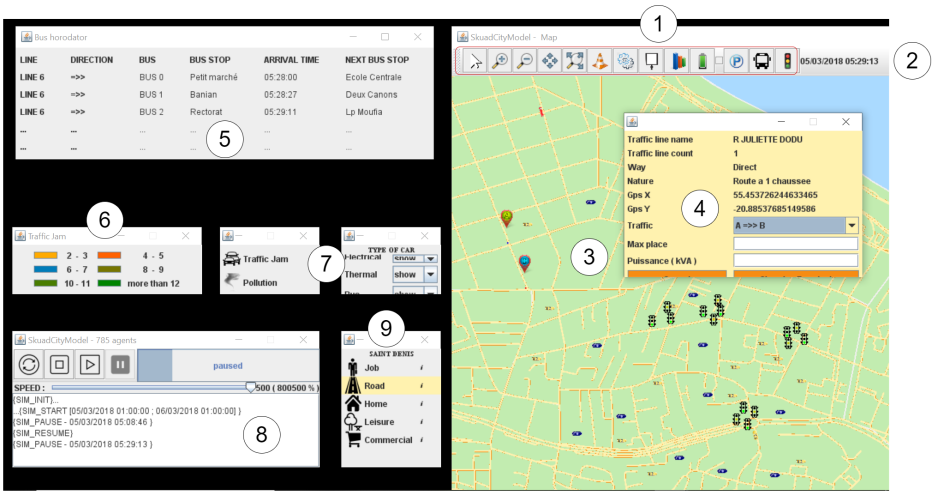


FIGURE 6.1 – L'interface utilisateur de SkuadCityModel

Le modèle dispose d'une interface graphique permettant l'interaction entre l'utilisateur et la simulation. "Interaction" signifie que l'interface graphique de SkuadCityModel permet non seulement l'affichage de données en temps réel, mais aussi le paramétrage de la simulation par l'utilisateur. Par exemple : l'utilisateur peut choisir, par le biais d'une interface graphique, les couches de l'environnement physique (route, lieu de résidence, lieu de travail, etc.) à afficher (9). Il peut également choisir le type de données statistiques à afficher en temps réel comme les embouteillages ou la pollution (7). Il peut également ajouter des compteurs de véhicule ou des barrages sur les routes (1). De manière générale, l'interface utilisateur de SkuadCityModel se compose des éléments suivants :

Une fenêtre principale de contrôle (8). Sur la figure 6.1, la fenêtre principale de contrôle permet le lancement, la relance, la mise en pause, et l'arrêt de la simulation. Elle contient également

une console pour l'affichage de l'historique des événements (logs) ainsi qu'une réglette pour le paramétrage de la vitesse de simulation.

Des fenêtres d'entrée (1) (4) (7) (9). Elles permettent le paramétrage de la simulation. Exemple : le paramétrage de l'affichage du réseau de routes, des bâtiments, des embouteillages, de la pollution, etc.

Des fenêtres d'affichage ou de sortie (5). Ce type de fenêtre permet de suivre en temps réel l'état de la simulation. Par exemple : (5) est une fenêtre de sortie qui permet le suivi en temps réel de l'état du réseau de bus.

Des fenêtres hybrides (1) (3) (4). Une fenêtre hybride est une fenêtre qui sert à la fois d'entrée et de sortie. (4) est une fenêtre hybride . Elle s'affiche lorsque l'utilisateur ajoute une borne de recharge électrique sur le territoire. Elle affiche les caractéristiques de la borne et permet son paramétrage. L'utilisateur peut également suivre l'évolution de la simulation en temps réel par le moyen de la fenêtre d'affichage (3). Nous pouvons y voir la carte avec les routes et les différentes zones du territoire, des véhicules individuels électriques (en jaune) et thermiques (en bleu), des bus (rouge), des bornes de recharge, des feux de signalisation, etc. Les boutons en haut de la fenêtre (1) permettent d'interagir directement avec cet affichage. Par exemple : pour ajouter des points de blocages sur certaines routes, pour ajouter des compteurs de véhicules, pour mettre en place des bornes de recharge électrique, etc.

6.2 Limites et propositions

6.2.1 Scénario 1 : Scénario de départ

Le scénario sur lequel nous nous penchons consiste en l'utilisation de SkuadCityModel pour la gestion de l'occupation des bornes de recharge électrique dans le temps et dans l'espace. Nous nous basons sur l'exemple décrit dans 1.1.2. Pour ce faire, comme expliqués dans le chapitre 5, nous nous intéressons particulièrement aux comportements des automobilistes. Les automobilistes possèdent un emploi du temps plus ou moins défini à l'avance. Cet emploi du temps se compose de différentes activités de la vie quotidienne nécessitant des déplacements. L'objectif principal de l'automobiliste est d'accomplir l'ensemble des activités qu'il a prévu dans son emploi du temps. Pour ce faire, il se déplace en utilisant sa voiture électrique. Cette dernière est dotée d'une capacité de charge limitée. L'automobiliste doit alors la recharger lorsque son niveau de batterie atteint un seuil. Cela l'amène à réviser son emploi du temps pour y intégrer l'activité *se recharger*, tout en tenant compte de l'ensemble des activités principales qu'il a déjà prévues auparavant. L'accès aux bornes de recharge est géré par les agents gestionnaires de borne de recharge électrique que nous appelons bornes. L'objectif d'une borne est d'optimiser son taux d'occupation.

Dans les différents scénarios que nous décrivons dans la suite de ce chapitre, nous nous concentrons uniquement sur les déplacements domicile-travail de 50 automobilistes en une journée ouvrable. Nous simulons une journée de déplacement (24h). Le territoire simulé dispose uniquement de 2 bornes de recharge publiques. Chaque borne peut accueillir un véhicule à la fois. Ces bornes sont des bornes de recharge rapide. En moyenne, elles prennent 30 min pour recharger un véhicule à 80%. Nous supposons que les automobilistes privilégient le rechargement sur ces bornes publiques, en dehors de leur période de travail. Nous appelons ces créneaux : *créneaux de pause*. Nous supposons que lorsque les automobilistes profitent du moment durant lequel ils effectuent leur course ou pratiquent leur loisir pour recharger leur véhicule dans les bornes publiques situées à proximité. De plus, en supposant que l'automobiliste dispose d'une borne de recharge privée chez lui ou au sein de l'entreprise dans laquelle il travaille, les horaires de courses ou de loisirs correspondent à des horaires où il n'est généralement ni chez lui ni à son lieu de travail. Par conséquent, ce sont des plages horaires où il n'a pas forcément accès aux bornes de recharge privées.

Ainsi, l'automobiliste choisit de recharger son véhicule soit avant de partir au travail, soit pendant la pause de midi, soit en début de soirée après le travail, avant de rentrer chez lui. Le choix des créneaux s'effectue donc majoritairement dans les plages horaires suivantes : 6h à 8h, 11h30 à 14h et 17h30 à 20h. Au début, chaque automobiliste choisit aléatoirement la borne de recharge ainsi que son créneau de rechargement. Le territoire simulé est la ville de Saint-Denis à La

Réunion. À La Réunion, un automobiliste parcourt en moyenne 25 km par jour [1]. Par conséquent, l'autonomie moyenne de la batterie d'un véhicule électrique devrait être amplement suffisante pour lui permettre d'effectuer son trajet. Nous estimons alors, dans nos scénarios, qu'un seul rechargement par véhicule suffit pour une simulation d'une journée. Pour chaque scénario, nous surveillons l'évolution de l'occupation des bornes de recharge dans le temps et l'espace.

Ce scénario de départ n'inclut aucune de nos contributions. La figure 6.2, montre les résultats de l'expérience. Les courbes affichent la longueur de la file d'attente au niveau de chaque borne en fonction du créneau horaire. La courbe correspondante à la variation de la longueur de la file d'attente au niveau de la borne 1 dans le temps est affichée en bleu tandis que celle de la borne 2 est en orange.

Nous remarquons plusieurs pics qui correspondent à des moments de forte affluence au niveau de chacune des bornes. Les courbes présentent également des périodes creuses où il n'y a aucune file d'attente. Les valeurs de la longueur de la file d'attente s'étendent de 0 à 6 véhicules pour la borne 1 et de 0 à 8 véhicules pour la borne 2.

Si nous nous focalisons uniquement sur les créneaux de pause, les écarts-types sont de 2,052 pour la borne 1 et de 2,093 pour la borne 2. Ces valeurs assez élevées indiquent une dispersion des données. Cela montre une répartition inégale de l'occupation des bornes de recharge dans le temps. Nous constatons également que les courbes n'ont pas la même allure, les heures de pointe et les périodes creuses ne coïncident pas forcément au niveau des deux bornes. Cela montre une répartition inégale de l'occupation au niveau de l'espace.

L'intégration de nos contributions devrait permettre de lisser un peu plus ces pics. Elle devrait également faire coïncider un peu plus l'allure des deux courbes. Cela signifierait une répartition plus uniforme de l'occupation des bornes de recharge. Elle devrait aussi permettre de réduire la longueur maximale de la file d'attente.

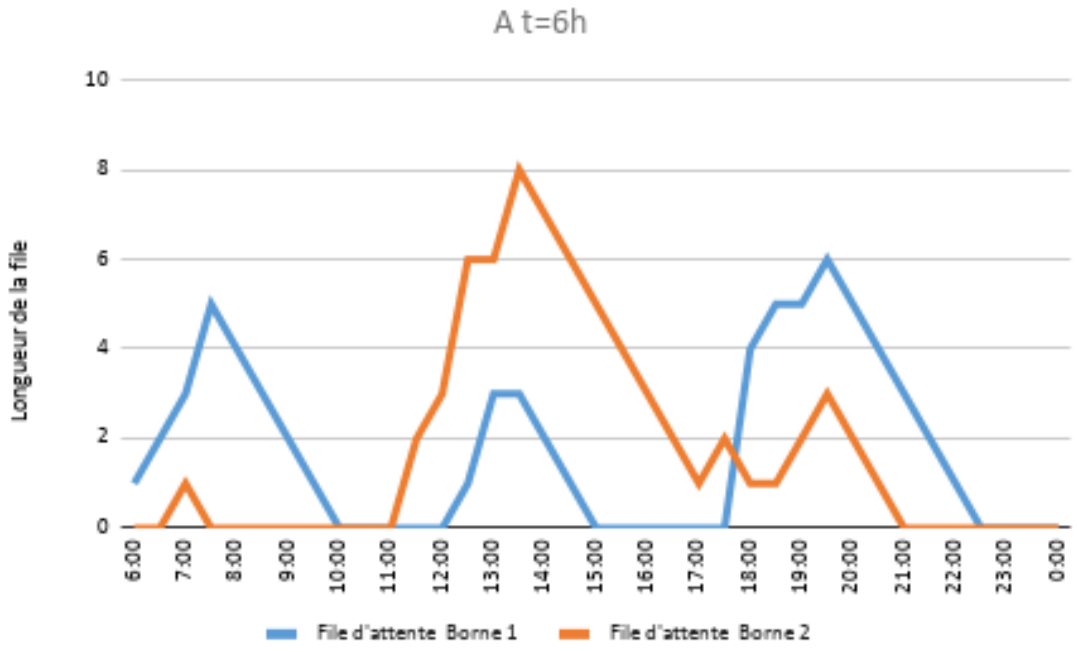


FIGURE 6.2 – Scénario 1 : modèle de référence, emploi du temps prévisionnel

6.2.2 Limites

Ce premier scénario fait ressortir certaines limites du système :

1. Le choix des créneaux de rechargement et des bornes de recharge se fait de manière aléatoire. Les agents n'intègrent aucune prise en compte des prévisions (emploi du temps) dans leur raisonnement ;

1. https ://www.insee.fr/fr/statistiques/1285585

2. L'automobiliste n'a aucune visibilité sur l'emploi du temps des bornes et sur celui des autres automobilistes. Les seules informations qu'il peut se procurer concernent uniquement l'état actuel de la borne (occupée ou libre) et la longueur actuelle de la file d'attente lorsqu'il arrive à la borne pour se recharger. Ces informations concernent l'instant présent. Il lui est donc difficile de savoir ou d'estimer l'état prévisionnel d'une borne à un instant donné dans le futur. Pourtant, ces informations existent et font partie de l'emploi du temps prévisionnel des agents. Cependant, aucun support n'est prévu pour les partager, les stocker et les consulter. En rendant ces informations accessibles au niveau collectif et en prenant en compte les informations partagées par les autres agents du système, l'automobiliste pourrait optimiser la programmation de son rechargement, dans le futur ;
3. Il n'existe aucun système qui permette à la borne d'avoir une emprise sur le nombre de véhicules qui viennent ou qui souhaitent se recharger sur lui.

Ces limites illustrent les besoins que nous avons énoncés dans les problématiques principales de cette thèse :

— Le besoin d'un support d'interaction pour l'échange d'informations spatiales, temporelles et sociales entre chaque individu au bénéfice du collectif ;
— Le besoin d'un raisonnement temporel qui prend en compte ces informations spatiales, temporelles et sociales partagées au niveau collectif.

Nous sommes face à un problème de gestion de ressources limitées et partagées. Nous démontrons dans les prochains scénarios décrits dans 6.3.1 et 6.3.2 comment nous nous affranchissons de ces limites. Nous montrons également des exemples de résultats.

6.2.3 Propositions

6.2.3.1 Considération de l'espace, des organisations et du temps comme des environnements

Cette première solution répond au premier besoin cité précédemment dans 6.2.2. Son but est de fournir aux agents du système un support d'interaction et de partage d'informations sur les trois dimensions : spatiale, sociale et temporelle. Notre modèle AGRET répond à ce besoin. Il est constitué d'un ensemble de trois types d'environnements qui permettent l'accès en écriture et en lecture à trois types d'informations : les informations spatiales, les informations sociales et les informations temporelles. Dans notre exemple de chargement de voitures :

— L'environnement temporel est utilisé pour le partage et la perception de l'emploi du temps et d'une partie de l'état des véhicules et des bornes. Cet emploi du temps et ces états concernent le passé, le présent et le futur.
— L'environnement spatial permet le partage et la perception d'informations spatiales comme la position spatiale des bornes de recharge et du véhicule.
— L'environnement social permet, via un système d'envoi et de réception de message, d'échanger des informations de manière asynchrone. Dans notre exemple, il s'agit de compléments d'information concernant l'état d'une borne : nombre actuel de requêtes similaires (combien d'automobilistes sont actuellement en train de réserver le même créneau horaire sur la même borne).

6.2.3.2 Raisonnement temporel

Cette deuxième contribution vient compléter la première. Il s'agit d'un raisonnement anticipatif. Il se base sur les échanges d'informations entre les automobilistes et les bornes de recharge. Cet échange se fait par le biais des environnements à leur disposition : espace physique, temps et organisation. La première particularité de notre proposition est qu'elle exploite les informations sur la position future du temps. Une autre particularité est qu'il s'agit d'une approche distribuée. En effet, notre proposition n'intègre pas d'agents destinés à contrôler le bon déroulement de l'activité collective. De plus, dans une perspective de faire également tourner le système en environnement ambiant, nous préférons nous abstenir de la mise en place de ce contrôle central. En effet, dans un environnement ambiant, il n'est pas toujours possible d'ajouter un élément externe pour piloter l'ensemble. Même dans le cas où c'est possible, centraliser le contrôle limite la robustesse du système. Le contrôle peut être interrompu à tout moment par des événements imprévisibles [93]. Par conséquent, dans notre cadre d'application, chaque entité du système est amenée à considérer la dimension collective. Dans notre approche, la considération de cette dimension collective est

intégrée au niveau de chaque agent du système, au niveau du mécanisme d'anticipation. Cette considération de la dimension collective se fait sur deux niveaux :

— au niveau du partage d'informations : chaque agent, individuellement, partage des informations et les rend accessibles au niveau collectif ;
— au niveau du raisonnement anticipatif de l'agent : chaque agent exploite la perception des informations partagées au niveau collectif pour satisfaire aux objectifs individuels et collectifs.

Dans notre exemple, le premier niveau équivaut à la borne qui rend son emploi du temps visible par les automobilistes. Tandis que le deuxième équivaut à l'automobiliste qui prend en compte au niveau du processus de raisonnement, les informations concernant l'état des bornes de recharge électrique qu'il perçoit de l'environnement. C'est-à-dire que d'un côté, plus une borne est saturée, plus le conducteur ne la considérera pas comme prioritaire dans son choix. En complément à cela, la mise en place du système d'abonnement et de notification permet aux bornes d'influencer les conducteurs pour qu'ils ajustent leur comportement en fonction des états de l'environnement. Ce système permet également aux automobilistes d'être informés des changements au niveau de la borne et de prendre cela en compte dans sa prise de décision.

Comme précisé précédemment, ce raisonnement anticipatif se base sur des échanges d'informations entre les automobilistes et les bornes de recharge. Cet échange se fait par le biais des environnements à leur disposition : espace, temps et organisation. Il existe plusieurs manières de récolter les informations :

— Par perception : c'est-à-dire que l'automobiliste décide volontairement de récolter des informations au niveau des environnements spatial, temporel ou social ;
— Par abonnement et notification au niveau de l'environnement social : lorsqu'un automobiliste s'abonne à un créneau au niveau d'une borne, il reçoit automatiquement des informations concernant les mises à jour au niveau de cette dernière. Par exemple : en cas de changement de la longueur prévisionnelle de la file d'attente.

L'accès aux bornes de recharge est géré par les agents gestionnaires de borne de recharge électrique (bornes). L'objectif d'une borne est de répartir au mieux son occupation. Pour se faire, elle utilise un système de notification. Pour éviter les périodes creuses, c'est-à-dire les périodes où elle est inoccupée, la borne essaie d'influencer les automobilistes à proximité. Elle leur envoie une notification afin que ces derniers remettent en question leur décision et décident de venir se recharger. L'envoi de cette notification se fait de manière périodique jusqu'à ce qu'elle reçoive une réponse positive. De la même manière, la borne notifie ses abonnées en cas de changements. Ces changements peuvent être par exemple une augmentation ou une diminution de la longueur prévisionnelle de la file d'attente.

6.3 Scénarios et résultats

6.3.1 Scénario 2 : intégration du raisonnement anticipatif et élargissement maximal de l'horizon temporel de perception

Afin de nous affranchir des différentes limites citées précédemment, nous mettons en place le mécanisme d'anticipation que nous avons décrit dans 4. Ce mécanisme se divise en deux parties :

— Un raisonnement anticipatif intégré au niveau des automobilistes. Ce raisonnement permet la remise en question de l'emploi du temps. Dans notre scénario, nous nous concentrons exclusivement sur la planification des recharges ;
— Un mécanisme d'abonnement et de notifications. La borne notifie l'automobiliste de son état. Cet automobiliste est soit abonné à la borne, soit situé aux alentours de la borne. Un automobiliste est considéré comme abonné à une borne lorsqu'il a réservé un créneau de recharge sur cette dernière.

Dans ce deuxième scénario, nous supposons que le véhicule a 50% de chance de remettre en question un créneau de recharge (choix du créneau et de la borne). Nous réglons l'horizon temporel de perception de manière à ce que l'agent puisse percevoir toutes les informations publiques partagées au niveau de l'environnement temporel. Ces informations sont la longueur prévisionnelle de la file d'attente et le nombre de requêtes en cours. Le nombre de requêtes en cours correspond au nombre de véhicules ayant signalé à la borne une volonté de se recharger au même créneau. Ces requêtes sont traitées à la fin du cycle d'activation courant. Cela entraîne la mise à jour du taux

d'occupation et donc de la longueur de la file d'attente au niveau de la borne au prochain cycle d'activation.

La figure 6.19 montre les résultats de l'expérience. Nous constatons que l'allure des courbes se stabilise au bout de la vingtième itération (15h30). Nous pouvons notamment constater une optimisation de la longueur maximale de la file d'attente. Les longueurs maximales de la file d'attente sont passées de 6 à 7 au niveau de la borne 1 et de 8 à 7 au niveau de la borne 2. Cela montre déjà une première amélioration de la répartition et un équilibre au niveau du taux l'occupation des bornes. L'écart-type passe de 2,052 à 1,883 au niveau de la Borne 1. Par contre, elle reste à 2,093 au niveau de la Borne 2. La diminution de l'écart-type au niveau de la Borne 1 indique que les données sont moins dispersées. Cela montre une optimisation du taux d'occupation de la borne de recharge dans le temps. Lorsque nous comparons visuellement l'allure des deux courbes (file d'attente borne 1 et borne 2), nous constatons une ressemblance et des coïncidences au niveau de certains créneaux. Cela indique une amélioration de la répartition de l'occupation des bornes au niveau de l'espace.

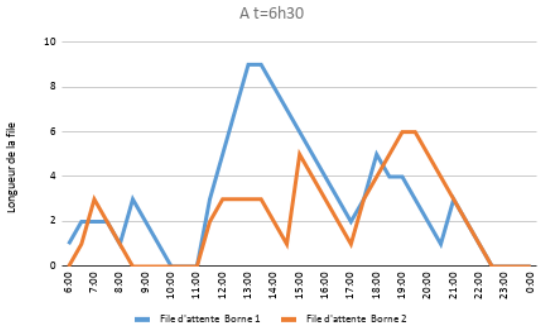


FIGURE 6.3 – Scénario 2 : Résultats à t_0

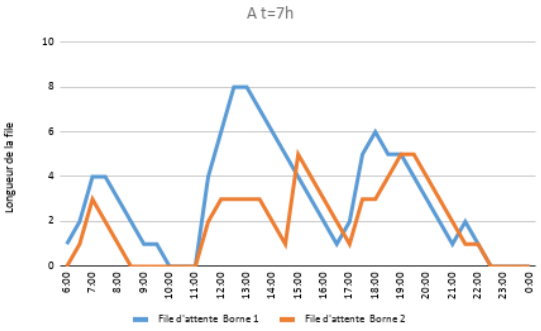


FIGURE 6.4 – Scénario 2 : Résultats à t_1

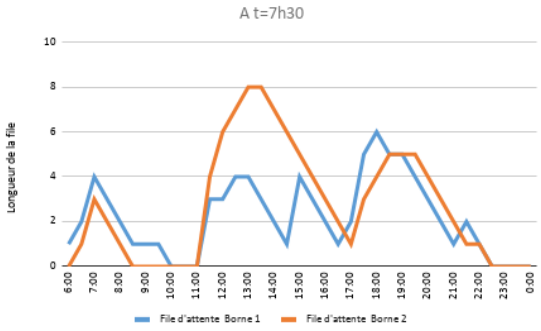


FIGURE 6.5 – Scénario 2 : Résultats à t_2

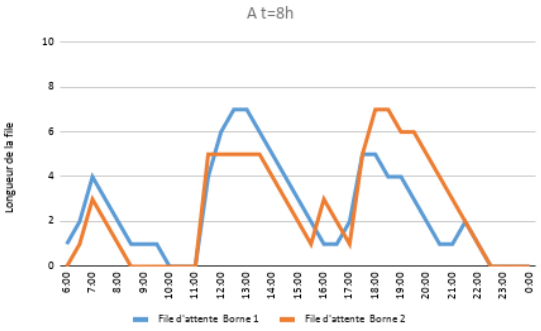


FIGURE 6.6 – Scénario 2 : Résultats à t_3

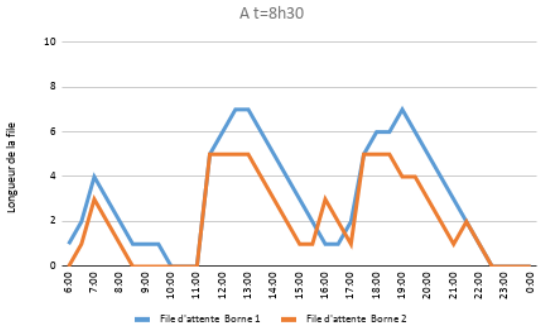


FIGURE 6.7 – Scénario 2 : Résultats à t_4

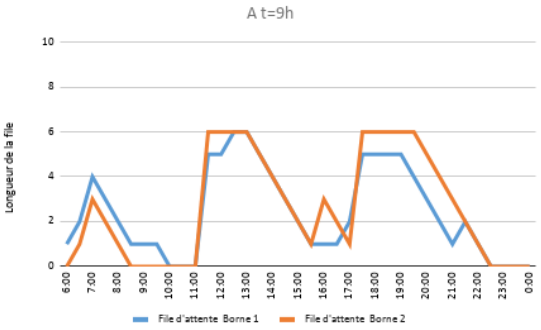


FIGURE 6.8 – Scénario 2 : Résultats à t_5

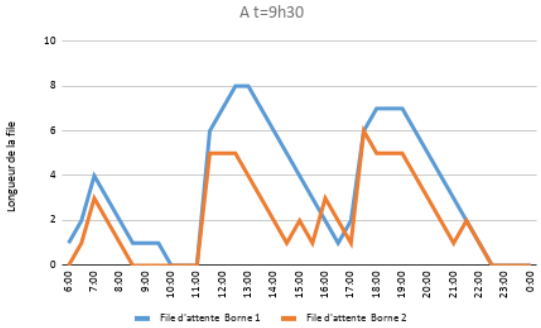


FIGURE 6.8 – Scénario 2 : Résultats à t_6

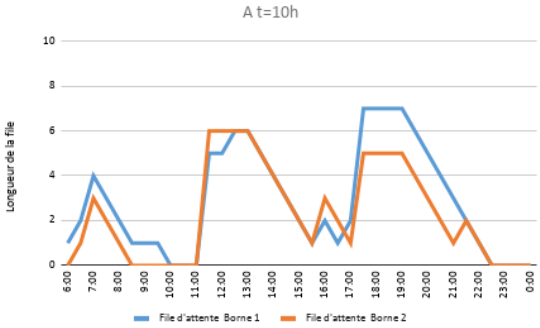


FIGURE 6.9 – Scénario 2 : Résultats à t_7

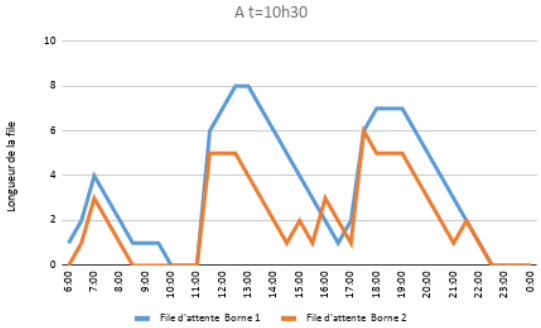


FIGURE 6.10 – Scénario 2 : Résultats à t_8

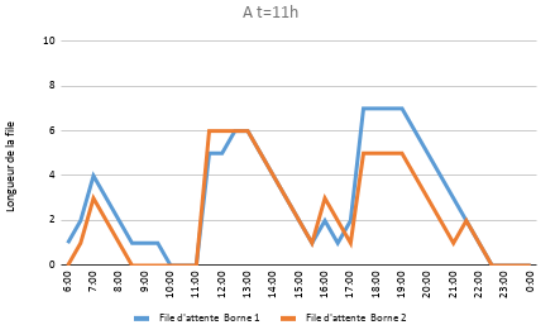


FIGURE 6.11 – Scénario 2 : Résultats à t_9

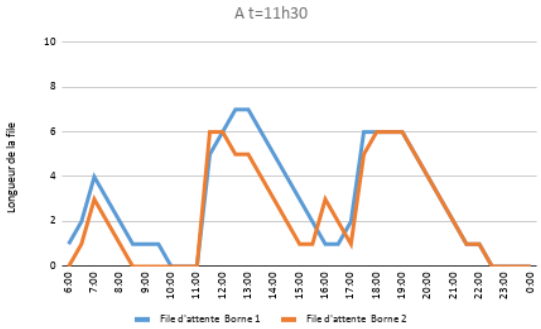


FIGURE 6.12 – Scénario 2 : Résultats à t_10

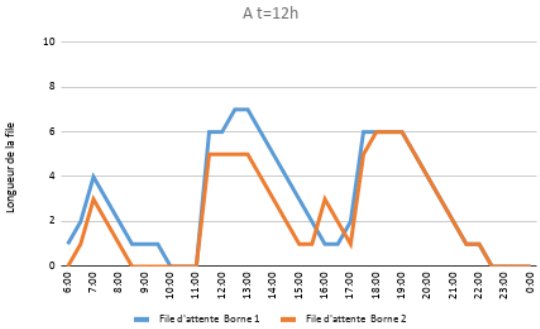


FIGURE 6.13 – Scénario 2 : Résultats à t_11

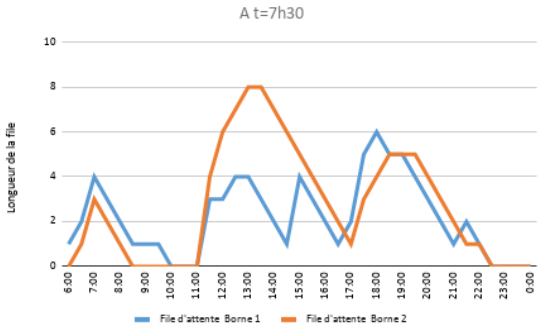


FIGURE 6.14 – Scénario 2 : Résultats à t_12

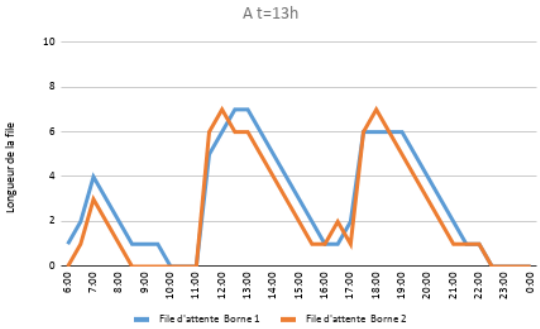


FIGURE 6.15 – Scénario 2 : Résultats à t_13

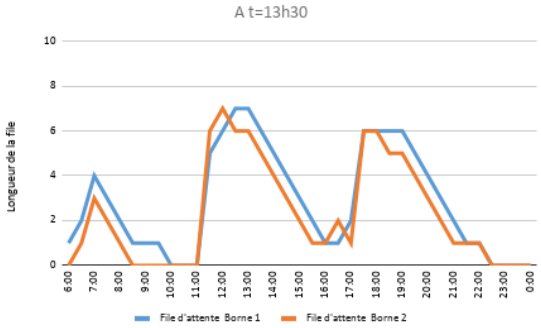


FIGURE 6.15 – Scénario 2 : Résultats à $t_1 4$

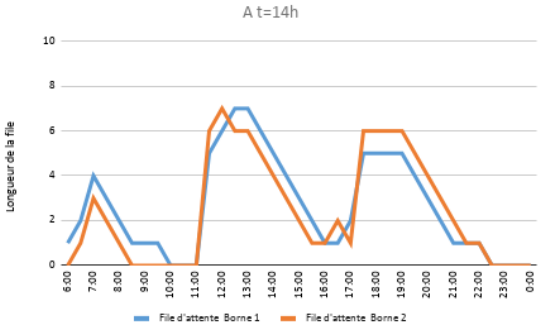


FIGURE 6.16 – Scénario 2 : Résultats à $t_1 5$

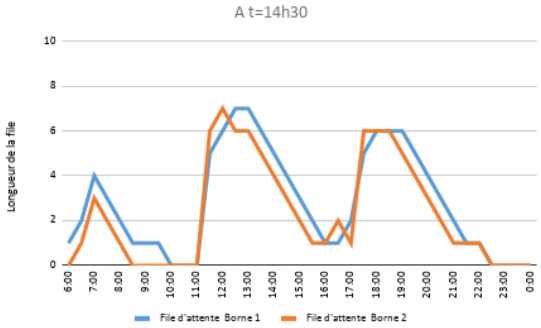


FIGURE 6.17 – Scénario 2 : Résultats à $t_1 6$

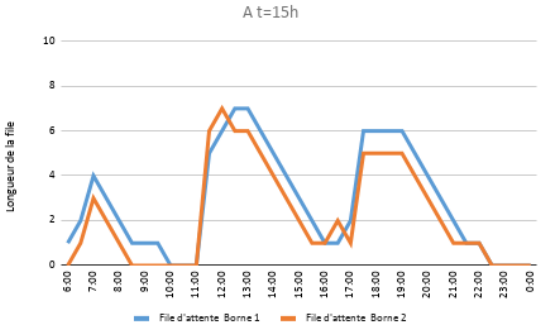


FIGURE 6.18 – Scénario 2 : Résultats à $t_1 7$

FIGURE 6.19 – Résultats du deuxième scénario : Intégration du raisonnement anticipatif, élargissement maximal de l'horizon temporel de perception.

6.3.2 Scénario 3 : Intégration du raisonnement anticipatif et réduction de l'horizon temporel de perception

Dans ce troisième scénario, nous réduisons la valeur de l'horizon temporel de perception. Dans notre exemple, nous lui affectons une valeur $\Delta_p = [0, 8]$. Cela signifie que l'agent peut percevoir uniquement les informations temporelles contenues dans un intervalle de 8h à compter de l'heure actuelle.

La figure 6.39 montre les différents résultats. Comparée au deuxième scénario, l'allure des deux courbes se stabilise au bout de la quinzième itération. Cela s'explique par le fait que l'agent ne peut avoir de visibilité totale sur l'ensemble des créneaux disponibles qu'au bout de 12 itérations, sachant que le pas de temps est de 30 min.

Dans ce scénario, nous constatons une diminution de l'étendue de la longueur maximale de la file d'attente au niveau des bornes. La longueur maximale de la file d'attente passe de 8 à 13 puis retombe à 7 au niveau de la Borne 1. Au niveau de la Borne 2, elle passe de 12 à 13 puis retombe à 8. En début d'itération (t=6h), les files d'attente s'étendent jusqu'à 19h bien que les créneaux de 17h30 à 20h ne soient pas perceptibles par les agents. En effet, ces créneaux ne rentrent pas dans l'horizon temporel de perception des agents. Cependant, les longueurs importantes des files d'attente pendant la plage horaire entre 12h à 14h font que le rechargement de tous les véhicules s'étend jusqu'à 19h.

Au fur et à mesure de l'avancement du temps, nous constatons une optimisation de l'occupation des bornes dans le temps et dans l'espace. À partir de 9h30, les créneaux de 17h30 à 20h deviennent visibles. Visuellement, nous constatons alors un changement au niveau de l'allure de la courbe quelque temps après, à partir de 10h30. En fin de cycle, nous constatons une forte baisse des écarts-types durant les créneaux de pause. Ils passent de 3,361 à 1,215 au niveau de la Borne 1 et de 3,797 à 1,847 au niveau de la borne 2. Les valeurs de départ assez élevées indiquent une dispersion des données en début du scénario. Comme dans le premier scénario, cela montre une répartition inégale de l'occupation des bornes de recharge dans le temps. La forte baisse indique

une amélioration considérable de la répartition de l'occupation des bornes dans le temps. L'allure des deux courbes présente des chevauchements au niveau de certains créneaux. Cela indique une optimisation de la répartition de l'occupation des bornes dans l'espace.

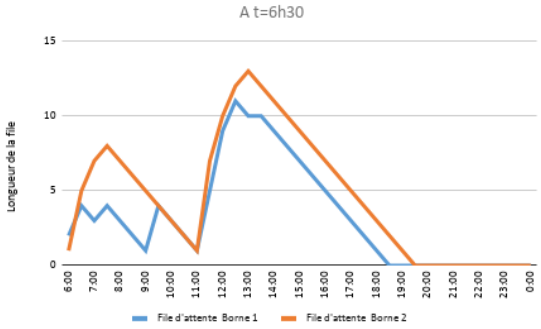


FIGURE 6.20 – Scénario 3 : Résultats à t_0

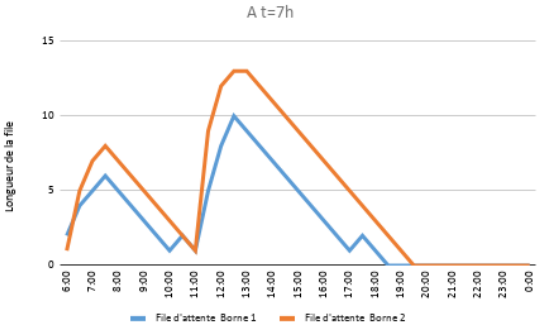


FIGURE 6.21 – Scénario 3 : Résultats à t_1

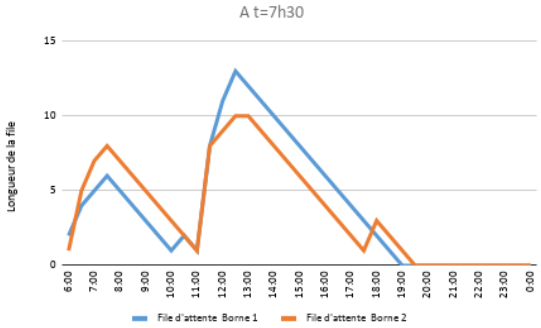


FIGURE 6.22 – Scénario 3 : Résultats à t_2

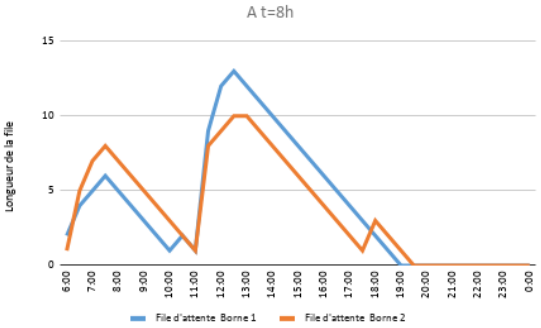


FIGURE 6.23 – Scénario 3 : Résultats à t_3

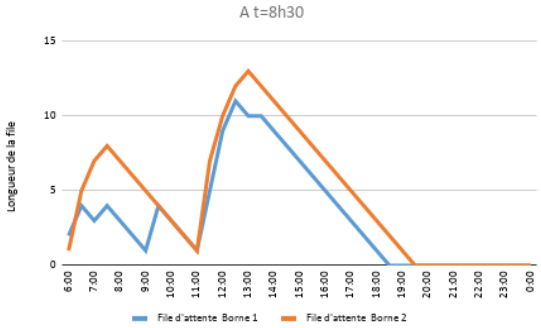


FIGURE 6.24 – Scénario 3 : Résultats à t_4

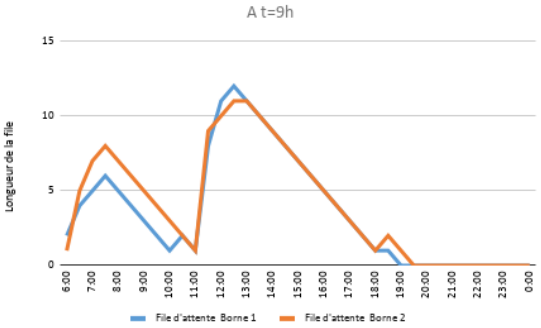


FIGURE 6.25 – Scénario 3 : Résultats à t_5

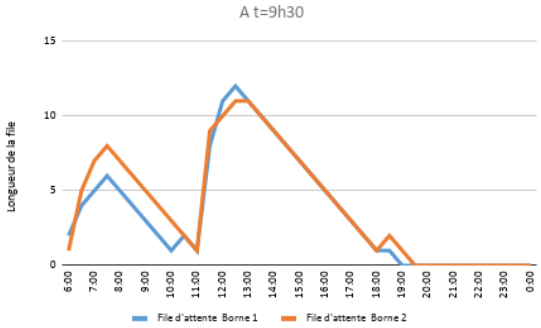


FIGURE 6.25 – Scénario 3 : Résultats à t_6

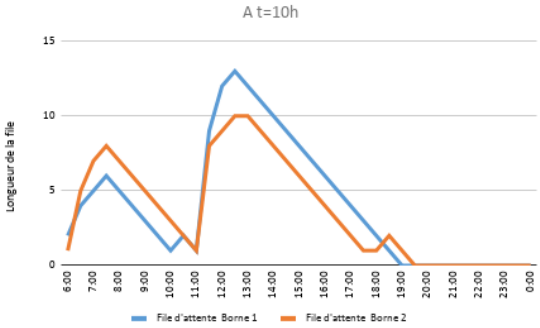


FIGURE 6.26 – Scénario 3 : Résultats à t_7

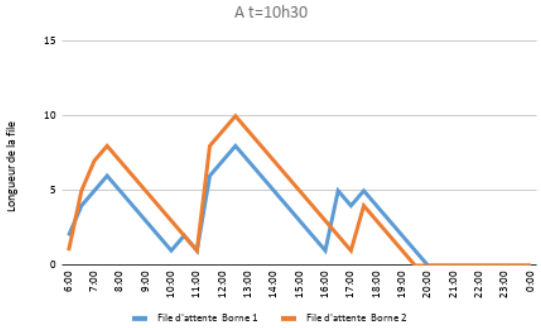


FIGURE 6.27 – Scénario 3 : Résultats à t_8

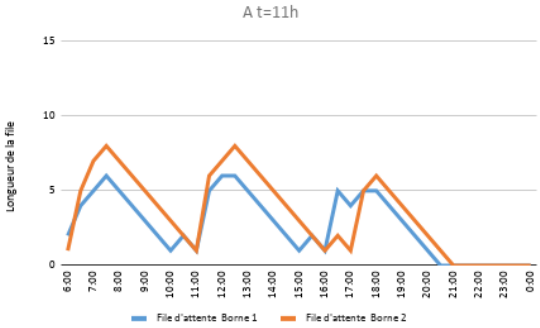


FIGURE 6.28 – Scénario 3 : Résultats à t_9

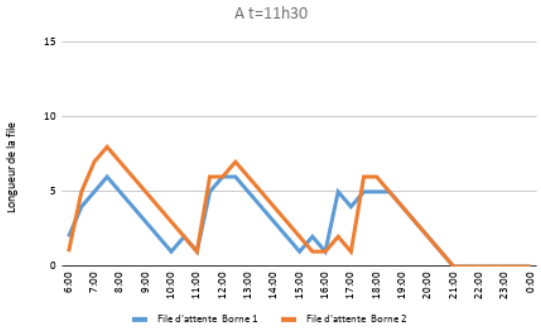


FIGURE 6.29 – Scénario 3 : Résultats à t_10

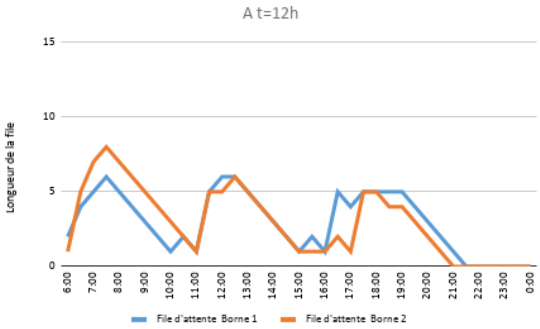


FIGURE 6.30 – Scénario 3 : Résultats à t_12

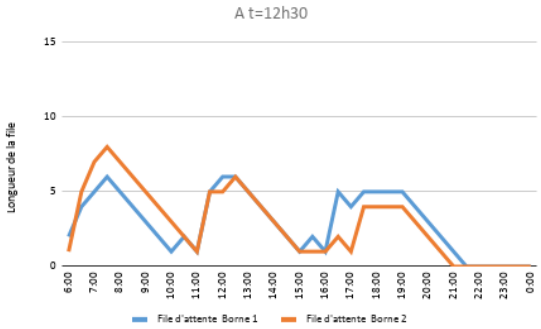


FIGURE 6.31 – Scénario 3 : Résultats à t_13

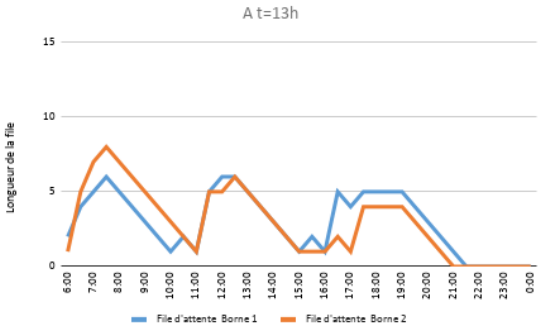


FIGURE 6.32 – Scénario 3 : Résultats à t_14

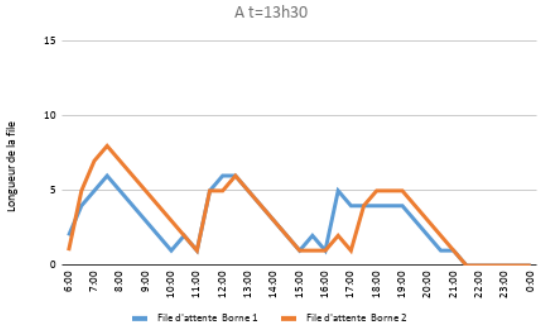


FIGURE 6.32 – Scénario 3 : Résultats à t_15

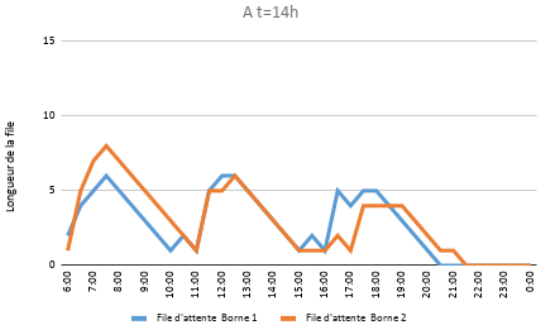


FIGURE 6.33 – Scénario 3 : Résultats à t_16

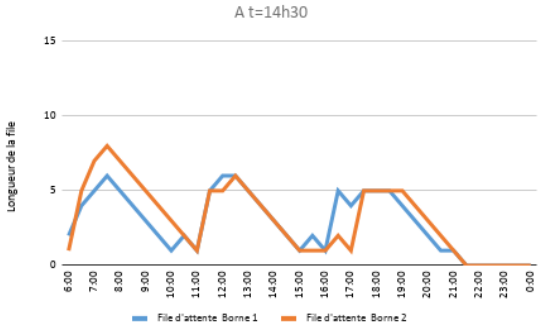


FIGURE 6.34 – Scénario 3 : Résultats à t_17

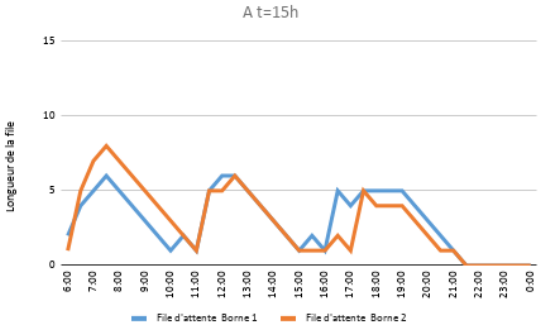


FIGURE 6.35 – Scénario 3 : Résultats à t_18

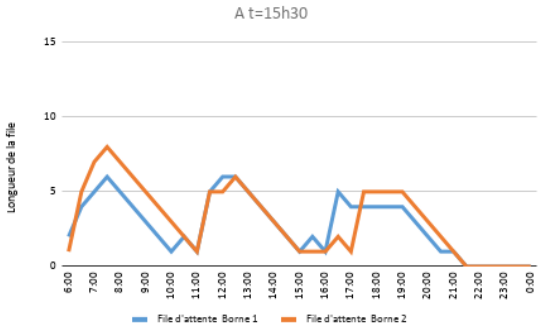


FIGURE 6.36 – Scénario 3 : Résultats à t_19

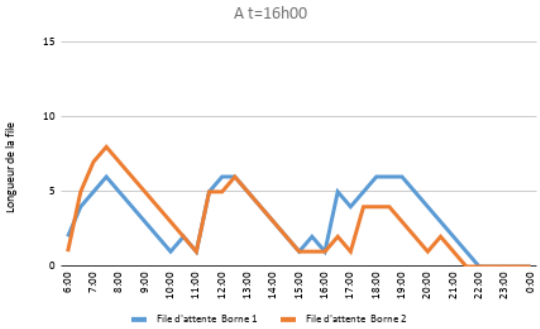


FIGURE 6.37 – Scénario 3 : Résultats à t_20

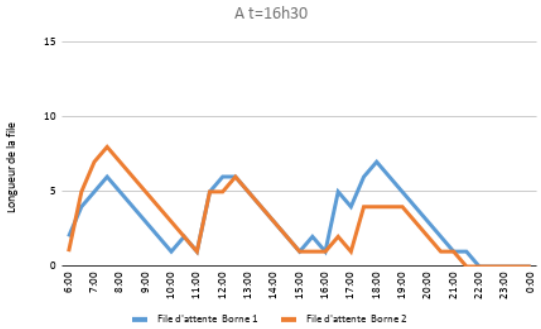


FIGURE 6.38 – Scénario 3 : Résultats à t_21

FIGURE 6.39 – Résultats du troisième scénario.

6.4 Conclusion

6.4.1 Synthèse

Les évaluations que nous avons effectuées dans ce chapitre viennent compléter la mise en oeuvre de nos solutions que nous avons présentées dans le chapitre 5 précédent. En effet, dans le chapitre précédent, nous avons présenté les aspects techniques de nos contributions, dans ce chapitre nous montrons les aspects pratiques. Dans ce cadre, nous avons repris l'exemple du rechargement des véhicules électriques sur des bornes de recharge publiques que nous avons présenté tout au début de ce manuscrit (cf. 1.1.2). Cet exemple a été mis en oeuvre dans le cadre du modèle de simulation SkuadCityModel que nous avons décrit dans 6.1. Il s'agit d'un modèle multi-agent composé de différents types d'agents : automobilistes, bornes de recharge, feux de signalisation, cartographe, statisticiens et météo. Dans le cadre de cette thèse, nous nous sommes intéressés uniquement aux conducteurs de véhicules électriques et aux bornes de recharge. Le modèle prend en entrée des données réelles et statistiques sous différents formats (SIG, xls, valeurs rentrées manuellement par l'utilisateur, etc.). Il génère, en sorties, différentes données dont le format et le contenu peuvent varier en fonction du scénario simulé. Les données en sorties peuvent également être déterminées par le concepteur du modèle. SkuadCityModel dispose d'une interface utilisateur composée d'un ensemble de fenêtres graphiques. Ces fenêtres ne sont pas seulement des fenêtres d'affichages, certaines d'entre elles permettent à l'utilisateur d'interagir avec la simulation.

6.4.1.1 Scénario 1 : Scénario de départ

Nous avons présenté dans la section 6.2 les optimisations qui ont été permises par l'implémentation de nos solutions. Pour illustrer cela, nous nous sommes basés sur un scénario de départ que nous avons présenté dans 6.2. Ce premier scénario simule le déplacement journalier domicile-travail des automobilistes. Dans ce cadre, nous nous sommes uniquement focalisés sur le rechargement sur des bornes publiques. Nous supposons alors que le choix des créneaux de recharge s'est fait parmi les plages horaires disponibles en dehors des heures de travail et des heures où l'automobiliste est chez lui. Le choix de la borne de recharge et du créneau s'est fait de manière aléatoire. Les résultats de ce scénario de base ont montré une répartition inégale du taux d'occupation des bornes au niveau de l'espace et du temps. Il y a également eu un écart important entre la valeur minimale de la longueur de la file d'attente et sa valeur maximale. Notre objectif a été d'optimiser ces résultats en intégrant nos contributions.

Ce scénario de base a fait ressortir certaines limites du système. Ces limites ont illustré les besoins que nous avons définis au début de ce manuscrit en tant que problématiques principales de cette thèse :

— Le besoin d'un support d'interaction pour l'échange d'informations spatiales, temporelles et sociales ;
— Le besoin d'un raisonnement temporel prenant en compte ces informations spatiales, temporelles et sociales.

6.4.1.2 Scénario 2 : intégration du raisonnement anticipatif et élargissement maximal de l'horizon temporel de perception

Notre deuxième scénario, que nous avons présenté dans 6.3.1, intègre nos contributions : la prise en compte de l'espace, du temps et des organisations en tant qu'environnement et un raisonnement anticipatif basé sur la perception des informations échangées par le biais de ces environnements. Dans ce cadre, nous avons défini une valeur de l'horizon temporel de perception qui est maximale. Cela veut dire que l'agent a pu percevoir toutes les informations de visibilité publique au niveau de l'environnement temporel. Le raisonnement anticipatif a été implémenté au niveau des automobilistes. Les informations visibles de manière publique au niveau de l'environnement temporel sont donc les informations concernant le emploi du temps des bornes de recharge. Les résultats ont montré une amélioration au niveau de la longueur de la file d'attente. Cela s'est traduit par une réduction de l'écart entre la longueur maximale et minimale de la file d'attente au niveau de chaque borne. Nous avons également constaté une amélioration de la répartition du taux d'occupation des bornes dans le temps et l'espace.

6.4.1.3 Scénario 3 : Intégration du raisonnement anticipatif et réduction de l'horizon temporel de perception

Notre dernier scénario ou scénario 3 a été le même que le scénario précédent. Cependant, dans ce scénario, nous avons réduit la portée de l'horizon temporel de perception. Nous lui avons affecté une valeur $\Delta_p = [0, 8]$. Comparés aux résultats du premier scénario, les résultats ont également montré une optimisation de la longueur de la file d'attente et de la répartition des recharges au niveau des bornes. Cependant, comparé au scénario 2 précédent, nous avons constaté une amélioration lente (qui prend du temps), mais tout aussi intéressante à long terme, dans le sens où les résultats en termes de répartition, comparés aux résultats du premier scénario sont concluants.

6.4.1.4 Bilan

D'une manière générale, les résultats du scénario 2 semblent plus prometteurs. En effet, en plus des avantages en termes d'amélioration de la répartition de l'occupation des bornes dans l'espace et dans le temps apportés par l'implémentation de nos contributions, un réglage plus large de l'horizon temporel temporel, comme celui mis en place dans le scénario 2 offre aux agents un choix plus large de plages horaires. De plus, l'élargissement de l'horizon temporel de perception leur permet de prendre en compte, dans leur raisonnement anticipatif, un ensemble d'informations plus riche. Cela leur permet également de remettre en question des activités éloignées dans le futur. Ces bénéfices se reflètent au niveau des résultats du scénario 2 qui sont meilleurs comparés à ceux du scénario 1 et scénario 3. Si nous comparons les trois de scénario, nous pouvons conclure que le scénario 2 est le meilleur scénario tandis que le scénario 1 est le pire scénario.

Nous estimons donc que nos propositions répondent bien aux objectifs que nous nous sommes fixés en début de ce manuscrit :

- Au niveau individuel :
  - Au niveau de l'automobiliste, l'objectif a été de permettre à chacun de respecter son emploi du temps. Cet emploi du temps se compose d'activités nécessitant des déplacements. L'automobiliste doit alors intégrer le rechargement de son véhicule dans son emploi du temps. Cela ne devrait pas perturber ce dernier. Nous avons considéré que cet objectif est atteint, car environ la moitié des véhicules, c'est-à-dire 54% pour le scénario 2 et 48% pour le scénario 3, ont pu se recharger durant les périodes de pauses. Ces rechargements ne perturbent pas l'emploi du temps de l'automobiliste, car ils se font hors période d'activité. Nous pouvons supposer que pour avoir un équilibre de charge au niveau du réseau électrique, la moitié restante pourrait se recharger dans des bornes de recharge privées. Par exemple : au niveau des entreprises, durant les heures de travail ou à domicile, durant les périodes où l'automobiliste est chez lui.
  - Au niveau de chaque borne, l'objectif a été d'optimiser la répartition du taux l'occupation dans le temps et dans l'espace. Nous avons également considéré cet objectif comme atteint lorsque nous comparons les résultats obtenus en début et en fin de chaque scénario.
- Au niveau collectif : Comme pour la borne, l'objectif a été d'optimiser le taux d'occupation des bornes dans le temps et dans l'espace. Les résultats ont également été prometteurs à ce niveau.

6.4.2 Limites et perspectives

Les évaluations que nous avons effectuées dans ce chapitre nous ont permis de montrer, d'un point de vue pratique, les avantages de nos propositions. Ces évaluations ont été effectuées "à la main", de manière théorique. Cependant, elles montrent déjà, un ensemble de résultats et des tendances prometteuses. Une de nos perspectives à ce niveau consiste alors à effectuer nos tests sur SimSKUAD. Cela nous permettra également de prendre en compte un nombre plus élevé d'agents et de lancer plusieurs tests. Nous pourrons aussi considérer un plus grand nombre de créneaux. Par ailleurs, une autre piste intéressante serait aussi de lancer une simulation sur plusieurs jours et d'intégrer, dans le raisonnement anticipatif, la perception des localisations temporelles passées. Cela permettra de montrer les avantages que cela pourrait apporter en plus des fonctionnalités que nous avons déjà testées dans les scénarios 2 et 3.

La mise en place de ces scénarios de test ne requiert pas de modification majeure au niveau de l'architecture et de l'algorithme de SkuadCityModel. Nous pensons en effet que notre modèle est

assez évolutif pour que tout cela puisse être intégré sans déstabiliser le fonctionnement général du système. Pour la plupart, il s'agit en réalité de paramétrages qui peuvent être faits :

— directement par l'utilisateur par le biais des fenêtres d'interactions mis à disposition de l'utilisateur au niveau de l'interface graphique ;

— par le concepteur du modèle en effectuant des modifications mineures au niveau du code source du modèle, sans restructuration ni modification majeure de l'algorithme.

De plus, les approches que nous avons proposées intègrent une prise en compte naturelle du contexte spatial, du contexte social et du contexte temporel par l'agent. Cela permet à l'agent de s'adapter plus facilement aux perturbations qui peuvent être introduites au niveau de ces trois environnements. Ces perturbations peuvent être par exemple la coupure des routes, l'ajout ou la suppression de véhicules ou de bornes. Dans ces cas-là, l'agent peut prendre connaissance de ces changements, les prendre en compte dans son raisonnement et s'y adapter par le moyen de la perception de l'environnement spatial. En cas de modifications d'agendas ou d'emploi du temps des autres agents, la mise en place de l'environnement temporel permet à l'agent d'avoir une visibilité sur ces types d'événements et de les prendre en compte naturellement. De même, l'agent adapte automatiquement son comportement suite à des notifications de changement qu'il reçoit par le biais de l'environnement social.

Chapitre 7

Conclusion et perspectives

Ce travail de thèse s'inscrit dans le contexte général de la modélisation et la simulation multi-agent pour l'élaboration et la planification de villes intelligentes. Notre contexte applicatif concerne principalement la modélisation et la simulation de flux de déplacement quotidien de véhicules électriques individuels sur un territoire. Dans l'exemple que nous avons choisi pour illustrer nos propositions, l'objectif est d'optimiser la gestion du rechargement des véhicules. Cet exemple illustre un problème plus général qui est la gestion d'une ressource partagée et limitée dans l'espace et dans le temps. Il s'agit d'un problème central qui fait partie de ceux à l'origine même de la création du concept des villes intelligentes. Une première étude autour de ce sujet nous a permis de relever deux besoins complémentaires sur lesquels nous avons apporté nos contributions :

- Un besoin de support d'interaction pour l'échange d'informations spatiales, temporelles et sociales ;
- Un besoin de raisonnement anticipatif prenant en compte les informations passées, présentes et futures planifiées par les agents.

Nous résumons dans la section 7.1 suivante nos propositions relatives à ces besoins.

7.1 Bilan

Après avoir établi un état de l'art autour des besoins cités précédemment, nous avons conclu que notre problématique relève de la non-prise en compte du temps simulé en tant que dynamique du modèle dans les SMA. Dans notre réflexion, nous avons distingué deux aspects de ce temps simulé : la représentation du temps et le raisonnement temporel. Chaque aspect nous a permis de répondre à chacun des besoins énoncés précédemment.

7.1.1 Un support d'interaction temporel : l'environnement temporel

7.1.1.1 Le modèle AGRET

Notre première contribution concerne la représentation du temps dans les simulations multi-agent. Dans ce cadre, notre objectif a été de considérer le temps simulé, au même titre que l'espace et les organisations. Nous avons donc représenté le temps comme un environnement, tout comme l'environnement spatial et l'environnement social. Cet environnement s'appelle l'environnement temporel. Pour articuler le tout, nous avons proposé le modèle AGRET (Agent-Groupe-Rôle-Environnement-Temps). AGRET est une extension du modèle AGRE (Agent-Groupe-Role-Environnement). Il rajoute l'aspect situé dans le temps aux deux aspects situés dans l'espace et dans la dimension sociale d'AGRE. AGRE est elle-même une extension du modèle générique d'organisation AGR. Il ajoute la prise en compte de l'environnement spatial au modèle AGR qui ne considère que les organisations.

Une des originalités de notre proposition, comparée à d'autres approches de gestion du temps est la représentation du temps simulé comme un milieu d'interaction. Cet environnement temporel vient en complément à l'ordonnanceur de la simulation.

En effet, dans la plupart des approches classiques dans les SMA, le temps simulé résulte d'une mécanique gérée par l'ordonnanceur de la simulation. Cette mécanique sert au cycle d'activation de la simulation. Ce cycle consiste en l'activation des environnements puis en l'activation des agents.

L'activation des agents par l'ordonnanceur se fait par lien direct entre ce dernier et l'agent à activer. Dans l'approche que nous avons proposée, nous avons cassé ce lien direct et nous avons interfacé l'environnement temporel entre l'agent et l'ordonnanceur. Ainsi, la gestion du temps au niveau de la simulation se trouve divisée en deux parties complémentaires :

— la représentation et le stockage des informations sur la dynamique temporelles sont gérés au niveau de l'environnement temporel. Cet environnement temporel est mis en place au niveau du modèle de simulation. Il offre une nouvelle dimension d'expression et de partage entre les agents ;

— la gestion de l'écoulement du temps et du cycle d'activation de la simulation est faite au niveau de l'ordonnanceur. L'ordonnanceur se situe au niveau de la plateforme de simulation.

L'environnement temporel fonctionne selon une mécanique inspirée de l'approche d'ordonnancement de type modèle à temporalité [70] et un modèle d'interaction de type influence/réaction.

La mécanique du modèle à temporalité est intéressante par son expressivité. En d'autres termes, elle permet aux agents de définir eux-mêmes et de partager leur dynamique d'activation temporelle. Elle intègre également un support permettant de représenter cette dynamique sous forme de temporalités, tempos et d'axe temporel. Nous avons donc là une approche qui permet le partage d'informations temporelles. Cependant, cette mécanique ne permet ni le stockage des informations ni leur perception. Cela se justifie par le fait qu'elle a été à l'origine conçue uniquement pour l'ordonnancement de la simulation. Ces fonctionnalités n'étaient donc pas nécessaires. Au contraire, leur intégration ne ferait qu'alourdir les traitements. Pourtant, elles sont indispensables dans le cadre de l'utilisation que nous en faisons, c'est-à-dire pour la structuration de l'environnement temporel. En effet, un support d'interaction doit permettre aux agents non seulement de partager et de structurer les informations, il doit également leur permettre de les stocker et de les percevoir.

Nous avons donc mis en place le stockage des informations passées, présentes et futures. Ce stockage au niveau de l'environnement temporel est contraint par un horizon temporel de stockage. Cet horizon temporel de stockage permet de garder uniquement les informations jugées pertinentes et non obsolètes. Son fonctionnement est similaire à l'évaporation des phéromones au niveau de l'environnement spatial. Cela permet non seulement une optimisation des performances, mais permet aussi au modélisateur et à l'utilisateur d'avoir la main sur le paramétrage du stockage des informations qu'il juge pertinent en fonction des objectifs et des besoins de la simulation.

Afin de permettre un réel échange d'informations, nous avons complété ce stockage par une perception des informations qui sont partagées au niveau de l'environnement temporel. Cette perception est contrainte, au niveau de l'environnement, par les règles d'accessibilité et au niveau de l'agent par un horizon temporel de perception. Dans l'usage que nous en faisons, nous définissons uniquement deux règles d'accessibilité :

— Publique : c'est-à-dire que l'information est accessible par tous les agents situés dans l'environnement temporel ;

— Privée : c'est-à-dire que l'information est accessible uniquement par l'agent qui l'a partagé.

Nous envisageons, en tant que perspective de cette thèse, la mise en place d'un système plus poussé de gestion de l'accessibilité aux informations temporelles, basé sur les réseaux sociaux. Nous en reparlons dans la section suivante.

La fenêtre temporelle de perception est l'équivalent temporel du champ de vision dans l'environnement spatial. Elle illustre une caractéristique de l'agent. Celle d'avoir une perception limitée de son environnement. Cette fenêtre temporelle est réglable par l'agent et par le modélisateur ou l'utilisateur de la simulation.

À ce milieu d'interaction qui est l'environnement temporel vient s'ajouter un modèle d'interaction qui fait le lien entre l'agent et l'environnement temporel. Nous avons utilisé pour cela, un modèle d'interaction de type influence/réaction particulièrement conçu pour la simulation multi-agent appelé IRM4S [32]. Ce modèle d'interaction de type influence/réaction présente l'avantage de respecter les contraintes d'intégrités environnementales et d'autonomie des agents. Il réduit aussi le couplage fort entre agent, action et environnement.

7.1.1.2 Environnements contraints et environnements non contraints par le temps.

Cette nouvelle considération du temps comme un milieu d'interaction a également bouleversé le cycle d'activation classique des simulations multi-agent. En effet, la mise en place de l'environnement temporel nous a fait prendre conscience de l'existence d'au moins deux types d'environnements dans les simulations multi-agent :

— les environnements contraints par le temps ;
— les environnements non contraints par le temps.

Les environnements contraints par le temps ont besoin de calculer leur nouvel état en début du cycle d'activation de la simulation, avant l'activation des agents. Ces calculs doivent s'établir sur l'intervalle de temps qui s'est écoulé depuis la dernière activation. Ces environnements ont des états qui évoluent dans le temps, en fonction des lois inertielles qui leur sont propres, et des influences qu'exercent les agents. C'est le cas de l'environnement spatial. Contrairement à cela, l'état de l'environnement temporel détermine la dynamique d'écoulement du temps. Par conséquent, cet état évolue hors du temps. L'état dynamique de l'environnement temporel reste alors soumis à des lois inertielles et aux influences exercées par les agents. Cependant, cela se fait de façon indépendante de l'intervalle de temps écoulé depuis les derniers calculs. Seule la connaissance de la dernière date de déclenchement courante est nécessaire pour servir de point de référence aux nouvelles localisations temporelles exprimées par les agents. L'environnement temporel est alors un environnement non contraint par le temps. Auparavant, le cycle d'activité classique d'un simulateur utilisant un type d'ordonnancement à temps discret a été rythmé par deux phases :

— Une phase d'activation de l'environnement durant lequel le simulateur doit traiter les conflits de simultanéité des actions du cycle précédent, mettre à jour l'horloge virtuelle, initialiser le prochain cycle, etc. ;
— Une phase d'activation des agents durant laquelle l'agent applique le processus itératif : perception, mémorisation, décision.

L'introduction de la notion d'environnement temporel nous a donc menés aux conclusions suivantes :

— les environnements contraints par le temps doivent être activés en début de cycle pour leur permettre de calculer leur nouvel état sur l'intervalle de temps écoulé ;
— les environnements non contraints par le temps doivent être activés en fin de cycle pour leur permettre de calculer leur nouvel état en fonction de l'ensemble des informations temporelles courant.

Ainsi, aux deux phases du cycle d'activation classique de la simulation s'ajoute une troisième phase. Désormais, les trois phases du cycle d'activation d'une simulation multi-agent sont :

1. La phase d'activation des environnements contraints par le temps sur l'intervalle de temps $]t - dt, t]$;
2. La phase d'activation des agents sur l'instant courant t ;
3. La phase d'activation des environnements non contraints par le temps sur la considération que t vient d'être passé.

7.1.2 Un raisonnement temporel : un raisonnement anticipatif basé sur AGRET

L'étude du temps simulé revêt deux aspects : la représentation du temps et le raisonnement temporel. Notre première contribution dont le fonctionnement a été résumé précédemment concerne la représentation du temps comme un environnement. Cette nouvelle considération du temps dans les SMA a ouvert de nouveaux horizons. L'un d'entre eux a été l'optimisation d'un raisonnement temporel : le raisonnement anticipatif. Cette approche d'anticipation, basée sur le modèle AGRET constitue notre deuxième contribution. Son originalité repose sur le fait qu'elle permet de prendre en compte, dans le raisonnement anticipatif les trois positions du temps : le passé, le présent, et le futur. En effet, la prise en compte des informations sur le passé et sur le présent dans les mécanismes d'anticipation dans les SMA est classique. Cependant, la prise en compte du futur est une nouveauté. Ces informations futures concernent les activités que les agents projettent d'effectuer. En effet, dans la plupart des approches d'anticipation dans les SMA, les agents sont contraints d'essayer de deviner ce qui va se passer dans le futur en réalisant des microsimulations en interne. Cela est dû au fait qu'ils ne peuvent avoir aucune information sur ce que les autres agents prévoient de faire. La mise en place de notre nouveau support qui est l'environnement temporel offre aux agents une visibilité sur cette dimension future du temps. Cette nouvelle dimension d'expression et de partage qui est de nature temporelle permet aux agents de partager leurs projets individuels et de les diffuser sur le collectif. L'analyse prédictive réalisée par les agents se base alors sur la perception de ce nouvel environnement commun, plutôt que par des calculs de simulation reproduits par chacun d'eux.

Concrètement, nous considérons que les agents possèdent un emploi du temps plus ou moins défini comme dans la vie réelle. Ce emploi du temps rassemble les activités passées, présentes et futures que l'agent souhaite effectuer. Ce emploi du temps est partagé et rendu accessible par les agents au niveau de l'environnement temporel, en fonction des règles d'accessibilité et de l'étendue de l'horizon temporel de perception de chaque agent ou chaque type d'agent. Nous avons donc exploité ces informations que l'agent peut récolter sous forme de percepts au niveau des différents environnements du système afin de remettre en question cet emploi du temps. Les objectifs ont été :

— évaluer ou de réévaluer la possibilité d'exécution d'une activité ;
— choisir l'activité la plus pertinente à exécuter immédiatement ;
— enrichir et d'apporter plus de précision à l'emploi du temps de l'agent.

Ce raisonnement anticipatif a été mis en place au niveau du processus de décision de l'agent. Comme beaucoup de modèles d'anticipation, notre modèle intègre un modèle prédictif et un modèle de décision. Le modèle prédictif prend en compte le contexte d'activation temporel, spatial et social des agents et génère des instanciations de plans d'action et une estimation de leurs coûts. Le modèle de décision choisit l'instanciation de plan la plus pertinente sur la base d'une pondération des coûts associés à chaque instanciation de plan et d'une minimisation de ces coûts pondérés.

Notre modèle prédictif intègre un modèle d'action et un modèle d'environnement. Notre modèle d'action est classique. Il est caractérisé par :

— des préconditions qui doivent être remplies avant l'exécution de l'action. Ces préconditions permettent l'enchaînement des actions. Cet enchaînement donne naissance à un plan ;
— des influences qui en résultent. Ces influences s'appliquent sur l'agent ou/et sur l'environnement ;
— des coûts qui définissent son importance vis-à-vis de l'agent et de la simulation ;
— des actionneurs qui permettent son exécution.

Notre modèle d'environnement, en particulier notre modèle d'environnement temporel, diffère des modèles classiques dans le sens où il est construit à partir de la perception de l'environnement temporel.

L'architecture de notre modèle prédictif est composée de trois (3) briques qui prennent chacune en entrée un ensemble de percepts spatiaux, temporels ou sociaux :

— La détermination d'actions élémentaires qui permet lister un ensemble d'actions abstraites que l'agent prévoit d'effectuer ;
— L'instanciation de plans qui consiste en trois (3) phases :
  — la déclinaison des actions abstraites en actions situées. C'est-à-dire la prise en compte de la réalité environnementale au niveau de chaque action abstraite ;
  — la vérification des préconditions. C'est-à-dire la vérification des conditions préalables à remplir pour que l'action puisse être exécutée ;
  — la déclinaison en plusieurs instances de plans. Un plan est un ensemble d'actions concrètes.
— La qualification des actions concrètes. Cette étape consiste à renseigner les coûts relatifs à chaque instanciation de plan ;

Notre modèle de décision procède à la sélection du plan le plus pertinent. Cette sélection s'appuie sur une minimisation des coûts pondérés. En d'autres termes, le plan le plus pertinent est celui dont la somme des coûts pondérés des actions qui le composent est minimale.

7.1.3 Mise en oeuvre dans le cadre le la plateforme de simulation SimSKUAD et du modèle de simulation SkuadCityModel

Nous avons mis en oeuvre nos solutions à travers un modèle de simulation appelée SkuadCityModel implémenté sur SimSKUAD, le mode simulé de la plateforme bimodale SKUAD. SKUAD est un outil que nous développons au sein même de notre équipe de recherche. Il fonctionne en environnement réel, avec des objets connectés et en simulation. Cette thèse porte essentiellement sur le mode simulé de SKUAD. Cependant, comme perspective, nous envisageons de transposer nos propositions en environnement réel.

Nous avons mis en oeuvre nos solutions dans SkuadCityModel, dans le cadre de la gestion des bornes de recharge électrique sur le territoire. Les objectifs ont été :

— Permettre aux agents conducteurs de véhicules électriques d'effectuer l'ensemble des activités qu'ils ont prévu dans leur emploi du temps par le biais des déplacements en utilisant leurs voitures électriques ;
— Éviter les longues files d'attente au niveau des bornes de recharge électrique ;
— Répartir l'occupation des bornes de recharge électrique dans l'espace et dans le temps.

Dans SkuadCityModel, l'environnement temporel a été implémenté sous forme d'environnement physique, au même titre que l'environnement spatial. Cela est en accord avec notre objectif de prendre en compte le temps au même titre que l'espace.

Les mécaniques environnementales au niveau de l'environnement temporel ont été implémentées sous forme de librairie Java générique. Cette librairie est indépendante du modèle et la plateforme de simulation. Ainsi, bien que la manière d'implémenter un environnement varie d'une plateforme à une autre, l'implémentation des mécaniques de l'environnement temporel peut se faire par réutilisation des méthodes fournies par cette librairie générique.

L'approche d'anticipation que nous avons mise en place est une approche distribuée. Cela veut dire qu'aucun agent ne joue le rôle d'un chef d'orchestre au niveau collectif. Chaque agent du système participe par son comportement à la satisfaction de ses objectifs personnels et des objectifs collectifs. Cette approche est en accord avec les principes des villes et îles intelligentes dans le sens où elle préconise une participation proactive de chaque individu aux objectifs collectifs du système. Les expérimentations que nous avons menées ont révélé une optimisation sur le long terme de la répartition de l'occupation des bornes de recharge électrique dans l'espace et dans le temps.

Nous avons pu implémenté une majeure partie des approches que nous avons décrites théoriquement dans les chapitres 3 et 4. Cela démontre l'applicabilité de nos solutions, non seulement au niveau conceptuel, mais également au niveau technique. Nous avons également pu évaluer, théoriquement, l'efficacité de nos propositions. Cette évaluation a été présentée dans le chapitre 6. Une évaluation complète sera faite une fois l'implémentation complètement finalisée. Cela figure parmi les perspectives de la thèse.

7.2 Perspectives

7.2.1 Perspectives au niveau du modèle de simulation : implémentation sur Repast Simphony et SmartCityModel

Une des perspectives que nous aimerions explorer dans la continuité de nos travaux de recherche concerne la mise en oeuvre de notre solution dans le cadre de la plateforme de simulation Repast Simphony et du modèle de simulation SmartCityModel.

Il s'agit d'une piste sur laquelle nous avons déjà commencé à travailler au cours de cette thèse. Nous avons notamment un prototype fonctionnel qui a déjà donné de premiers résultats. Cela a fait l'objet de publications [76] [75]. Une implémentation complète des nos contributions et de l'approche d'ordonnancement de type modèle à temporalité au niveau de la plateforme Repast Simphony et de SmartCityModel figure parmi nos perspectives.

Dans Repast Simphony, l'environnement est structuré en contexte et en projection.

Definition 7.2.1. Contexte. Les contextes sont des conteneurs qui contiennent des composants du modèle. Ces composants peuvent être n'importe quel type d'objet Java, y compris un autre contexte, ou même des agents. L'environnement de l'agent est représenté par des contextes. Les entités peuvent adapter leur comportement au contexte actuel dans lequel elles se trouvent [47]. Chaque agent doit appartenir à au moins un contexte. Ce dernier leur permet de s'orienter en fonction d'un ensemble de données. Les agents peuvent alors adapter leur comportement au contexte dans lequel ils se trouvent. Un contexte peut être organisé de manière hiérarchique, c'est-à-dire qu'il peut contenir des sous-contextes.

Definition 7.2.2. Projection. Une projection est utilisée pour définir l'espace dans lequel se trouvent les agents, ainsi que leur relation. Elle spécifie l'environnement dans lequel sont les agents et impose une structure au contexte. La projection est aussi utilisée pour la visualisation et l'affichage de la simulation. Un contexte doit inclure au moins une projection, car sans ce dernier, les agents ne pourront ni interagir ni communiquer entre eux. Une projection peut être soit un espace continu, soit une grille (2D ou 3D), soit du Système d'Information Géographique (SIG), soit un réseau ou toute autre implémentation de l'interface Projection.

L'implémentation de l'environnement temporel dans Repast Simphony se résume alors en l'implémentation d'un nouveau type de contexte et de projection.

7.2.1.1 Un contexte personnalisé

L'implémentation d'un contexte personnalisé peut se faire de deux manières :
— en créant une implémentation complète de l'interface Context à partir de zéro ;
— en créant une classe qui hérite de la classe DefaultContext de Repast Simphony.
La création de classe héritant de DefaultContext est la méthode recommandée. Néanmoins, les implémentations personnalisées de contexte peuvent être utilisées exactement de la même manière que le contexte par défaut. Dans notre cas, l'utilisation d'une classe qui étend DefaultContext est amplement suffisante. Le processus de remplissage du contexte avec des agents, des projections et des sous-contextes se fait lors de l'initialisation de la simulation. Une fois le contexte créé, l'assemblage des composants du modèle se fait en utilisant une méthode de l'interface ContextBuilder de Repast. Un modèle doit inclure une et une seule implémentation de ContextBuilder.

Pour l'implémentation dans SmartCityModel, notre proposition consiste à utiliser le contexte qui regroupe tous les agents du système nommé AgentContext. Nous avons déjà une implémentation de ce contexte dans notre modèle actuel. Ce contexte est utilisable comme tel. Aucune modification supplémentaire n'est donc requise à ce niveau.

7.2.1.2 Une projection personnalisée

Comme nous l'avons expliqué, une projection peut être soit un espace continu, soit une grille (2D ou 3D), soit du Système d'Information Géographique (SIG), soit un réseau ou **toute autre classe qui implémente une interface Projection**. Dans notre cas, la mise en place de l'environnement temporel requiert un nouveau type de projection que nous appelons projection temporelle et qui implémente l'interface Projection. Cette nouvelle projection permet la structuration de l'environnement temporel. C'est donc à ce niveau que s'implémentent les mécaniques de l'environnement temporel qui se basent sur les mécaniques du modèle à temporalité.

Plusieurs autres améliorations peuvent également être proposées si l'on s'intéresse au modèle de simulation comme la mise en place d'un mécanisme d'anticipation au niveau des agents gestionnaires de bornes de recharge. Nous comptons également finaliser l'implémentation de nos propositions au niveau de SkuadCityModel. Cela permettrait une évaluation plus complète et une validation du modèle.

7.2.2 Perspectives au niveau conceptuel : la piste des réseaux sociaux

7.2.2.1 Des règles de visibilité et un système d'abonnement basés sur une approche de type réseau social

Comme nous l'avons mentionné à la fin du chapitre 3, dans l'approche que nous proposons, nous avons choisi de nous concentrer principalement sur l'espace physique et le temps. Néanmoins, la dimension sociale fait partie de notre proposition (dans le modèle AGRET) et nous l'utilisons sur plusieurs niveaux dans nos contributions. Nous sommes cependant conscients qu'elle pourrait être exploitée de manière encore plus poussée. En effet, les organisations fournissent un moyen de partitionner un SMA. Chaque partition constitue un contexte d'interaction pour les agents. Dans ce cadre, nous avons déjà mené un début d'étude, au cours de la thèse, concernant l'intérêt que pourraient avoir les approches comme celles utilisées dans les réseaux sociaux (Facebook, LinkedIn) dans l'optimisation des simulations multi-agent. La piste des réseaux sociaux est intéressante, car il s'agit d'outils qui sont fortement utilisés actuellement pour l'échange d'informations dans les villes. Une utilisation consiste à mettre en place des règles d'accessibilité des informations temporelles qui se basent sur les groupes et les rôles dans la dimension sociale. Une autre utilisation qui vient compléter la première est la une mise en place d'un système d'abonnement plus abouti. Une utilisation plus élaborée du système d'abonnement consiste à définir un groupe d'abonnés au niveau de l'environnement social. L'abonnement à un objet ou à une localisation temporelle consiste alors en une inscription au groupe d'abonnés correspondant. Une fois inscrit dans ce groupe, l'agent aura accès à un certain nombre d'informations relatives à l'objet ou à la localisation temporelle. De même pour les règles d'accessibilité, l'agent pourra gérer l'accès à un certain nombre de ses

localisations temporelles en fonction de groupes particuliers. Exemples : famille proche, groupe d'amis, cercle de connaissance, etc.

Dans le contexte multi-agent, afin de ne pas porter confusion au niveau de la définition, nous parlons plutôt de média social.

Les premiers résultats de cette étude nous ont permis de dresser la définition 7.2.4 illustrée par la figure 7.2.

Definition 7.2.3. Réseau social.

Amblard [5] définit un réseau social comme étant un ensemble de relations qui sont elles-mêmes définies comme un couple d'agents ou d'acteurs (cf figure 7.1)

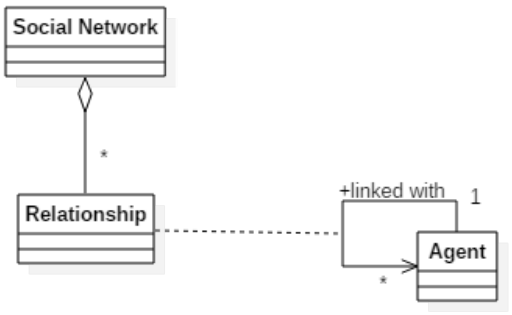


FIGURE 7.1 – Diagramme conceptuel d'un réseau social.

Definition 7.2.4. Média social.

Sur cette base, nous définissons un média social comme étant un ensemble d'espaces médias auquel les agents ont accès en fonction du type de relation qui existe entre eux (cf figure 7.2).

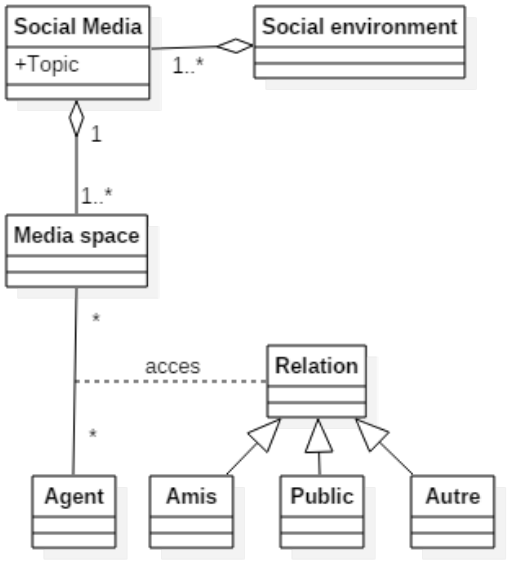


FIGURE 7.2 – Diagramme conceptuel d'un média social.

Definition 7.2.5. Espace média.

Il s'agit de l'espace sur lequel l'agent dépose des informations afin que d'autres agents puissent y accéder, en fonction du type de relation qui existe entre eux. Dans notre cas, il s'agit de l'agenda dans l'environnement temporel.

Il s'agit d'une piste très intéressante de par son intérêt pour les villes intelligentes et pour les SMA. De plus, la structuration de la dimension sociale au niveau de notre plateforme SKUAD fait

déjà transparaître quelques notions liées aux réseaux sociaux : avatar, participant, boîte mail, etc. Dans SKUAD, un space est un espace d'interaction social au sein duquel l'agent peut communiquer explicitement (par envoi de message) ou implicitement (par observation) avec les autres agents connectés à ce même space. Ces fonctionnalités peuvent par exemple être réutilisées dans le cadre de l'optimisation du système d'abonnement. Les agents pourraient s'abonner à des spaces particuliers afin d'être notifiés des changements au niveau de ces spaces. Une description plus détaillée du fonctionnement de la dimension sociale dans SKUAD est faite dans le chapitre 5. Pour des raisons d'organisation et de temps, nous avons choisi de nous concentrer uniquement sur la prise en compte du temps au même titre que la dimension spatiale. L'intégration des approches liées aux réseaux sociaux figure alors parmi les perspectives intéressantes pour la suite de la thèse. Que ce soit au niveau conceptuel ou au niveau de l'implémentation, les architectures déjà en place actuellement au niveau du modèle AGRET, de SimSKUAD et de SkuadCityModel devraient permettre d'intégrer facilement cette fonctionnalité. En effet, nous pensons que les approches que nous avons proposées et mises en place sont assez évolutives pour ne pas nécessiter de refonte ni modifications majeure au niveau du modèle et de l'algorithme.

7.2.2.2 Amélioration des horizons temporels et formalisation de la perception

L'ensemble des propositions que nous avons énoncées dans cette thèse reposent sur la perception de l'environnement temporel. Bien que le fonctionnement du processus de perception ait été expliqué à plusieurs reprises dans ce manuscrit, nous pensons qu'il serait intéressant de proposer un modèle formel de perception de l'environnement temporel.

Pour compléter cela, nous pensons également à l'optimisation de l'horizon temporel de perception au niveau de l'agent et de l'horizon temporel de stockage au niveau de l'environnement temporel. Comme nous l'avons mentionné dans le chapitre 3, il est très important de choisir une valeur optimale de ces horizons en fonction des objectifs et des particularités de la simulation et des agents. Selon les besoins, il est également possible de définir des horizons temporels "ciblés" comme un horizon temporel périodique par exemple. La valeur de l'horizon temporel de stockage peut également varier en fonction du type d'application. En effet, la pertinence et l'obsolescence des informations varient en fonction des objectifs. Par exemple : une simulation de l'évolution du développement d'une ville aurait peut-être besoin de stocker des informations sur plusieurs années tandis qu'une simulation de l'occupation des routes pourrait n'avoir besoin de stocker que des informations sur quelques heures, quelques semaines ou quelques mois uniquement.

D'autres améliorations peuvent également être faites comme l'optimisation de la fréquence de déclenchement du mécanisme d'anticipation ou de l'amélioration des critères de coûts.

7.2.2.3 Les environnements non contraints par le temps

Nous avons introduit dans le cadre de cette thèse la notion d'environnement non contraint par le temps. L'environnement temporel en est un exemple. Cette nouvelle catégorie d'environnement constitue une piste intéressante de recherche encore inexplorée. La question se pose alors concernant la possibilité d'existence d'autres environnements dont l'état évoluerait hors du cadre imposé par le temps. Notre réflexion s'oriente vers les méta-environnements d'observation. Ces derniers ont pour rôle de traiter les flux d'informations générés par la simulation. À partir de cela, ils génèrent à leur tour des informations utiles qu'ils affichent à l'utilisateur par exemple.

Cette piste viendrait en complément de travaux de thèses déjà effectuées auparavant au sein de notre équipe de recherche [21] [74].

7.2.3 Perspectives au niveau du contexte applicatif : vers un concept d'île intelligente

Une des problématiques sur lesquels nous avons travaillé en début de cette thèse, mais que nous n'avons pas pu continuer pour des raisons de temps et de priorité concerne le concept d'île intelligente. Nous partons d'un constat selon lequel beaucoup d'études sont faites autour de la ville intelligente, mais très peu traitent des îles intelligentes. Plusieurs termes sont utilisés, dans la littérature, pour désigner une île intelligente : intelligent island [19], cyberile [82], smart island, île à réseaux intelligents [41]. L'un des premiers projets mentionnant le terme "île intelligente" est celui de Singapour, dans les années 90 [19]. À l'époque, Singapour voulait devenir l'un des premiers pays

à avoir une infrastructure d'information sophistiquée, à l'échelle nationale, avec des ordinateurs interconnectés dans pratiquement chaque maison, bureau, école et usine. Cela afin d'améliorer la qualité de vie et la croissance économique du pays. En 2015, Gioda [41] a adopté une vision d'une "smart island", axée un peu plus sur le développement durable. Dans un projet concernant les Îles Canaries, il définit une "smart island" ou une île à réseaux intelligents comme une île avec une électricité 100% issue d'énergies renouvelables pour tous les usages : dessalement, énergie y compris la mobilité (pour les véhicules légers) avec un système réversible à terme ou "V2grid". Bien qu'il existe des différences assez marquantes entre les systèmes de ville et les systèmes d'île, la thématique de "smart island" ainsi que la mobilité dans les transports dans les îles reste actuellement sujette à très peu de projets de recherche. De plus, il n'existe pas encore de définition universelle ni de standard permettant de qualifier une île de "smart".

Les îles sont un type particulier de systèmes insulaires. Le terme insulaire fait référence aux zones possédant des limites spatiales plus ou moins marquées (ce qui est le cas des îles, des oasis ou des plates-formes pétrolières), qui permettent d'identifier, au moins théoriquement, un ensemble d'éléments d'origine diverse, participant aux mêmes dynamiques territoriales. Ainsi, l'insularité est propice à l'étude du fonctionnement et de la viabilité des systèmes logiques. Dans cette optique, les îles sont des exemples intéressants[59].

D'une manière plus générale, la piste des îles intelligentes nous semble intéressante dans l'étude et l'optimisation de la réplicabilité et la transférabilité d'un modèle de simulation multi-agent d'un territoire à un autre. Dans ce cadre, la ville et l'île peuvent être considérées comme deux extrêmes en termes de caractéristiques. En effet, beaucoup de modèles de simulation multi-agent sont conçus et testés dans le cadre d'un type particulier de territoire, avec les caractéristiques correspondantes. La plupart du temps, il s'agit de territoires urbains (villes). Le transfert de modèle de ce type dans des contextes de territoires insulaires (comme les îles) nous a permis de détecter un certain nombre de limites. En effet, en raison de leurs différences géographiques et socioculturelles, les villes et les îles réagissent différemment aux mêmes changements. Il s'agit d'une piste intéressante dans l'étude de la transférabilité des solutions de ville intelligente des villes aux îles et inversement. Tester un modèle de simulation conçu pour un contexte urbain dans un contexte à fortes contraintes peut aider à sa consolidation. Une première expérimentation a été menée dans le cadre de la ville de Londres et de l'île de La Réunion. Dans cette expérimentation, nous avons montré l'exemple de la conception d'un modèle de simulation pour un contexte d'île intelligente par enrichissement d'un modèle de simulation existant qui n'était auparavant appliqué qu'aux villes. Ces expériences ont été réalisées sur la plateforme de simulation Repast Simphony. L'objectif était de concevoir un modèle de simulation assez générique pour être facilement transférable d'un territoire à un autre doté de caractéristiques très différentes. Cela a fait l'objet de deux publications [75] [76].

Bibliographie

[1] The ecological sequestration trust (test). `https://resilience.io/`. Accessed : 2019-12-24.

[2] World urbanization prospects : The 2018 revision, 2018. URL `https://esa.un.org/unpd/wup/Publications`.

[3] N. Aky, T. Ralitera, D. Payet, and R. Courdier. Skuadcitymodel : Une simulation de déplacements urbains construite sur la plateforme SKUAD (démonstration). In *Distribution et Décentralisation - Vingt-sixièmes journées francophones sur les systèmes multi-agents, JFSMA 2018, Métabief, France, 10-12 Octobre 2018.*, pages 233–234, 2018.

[4] O. Alonso, J. Strötgen, R. Baeza-Yates, and M. Gertz. Temporal information retrieval : Challenges and opportunities.

[5] F. Amblard. Which ties to choose ? a survey of social networks models for agent-based social simulations. In *Proceedings of the 2002 SCS International Conference On Artificial Intelligence, Simulation and Planning in High Autonomy Systems*, pages 253–258, 2002.

[6] F. Badeig. *Un environnement actif pour la simulation multi-agents : application à la gestion de crise dans les transports.* PhD thesis, Paris 9, 2010.

[7] J. Banks. Introduction to simulation. In *Proceedings of the 31st conference on Winter simulation : Simulation - a bridge to the future, WSC 1999, Phoenix, AZ, USA, December 05-8, 1999, Volume 1*, pages 7–13, 1999. doi : 10.1109/WSC.1999.823046. URL `http://doi.ieeecomputersociety.org/10.1109/WSC.1999.823046`.

[8] G. Bustos-Turu and K. H. van Dam. Integrated planning of distribution networks : interactions between land use, transport and electric vehicle charging demand. 2015.

[9] G. Bustos-Turu, K. H. van Dam, S. Acha, and N. Shah. Estimating plug-in electric vehicle demand flexibility through an agent-based simulation model. In *IEEE PES Innovative Smart Grid Technologies, Europe*, pages 1–6. IEEE, 2014.

[10] G. Bustos-Turu, K. V. Dam, S. Acha, and N. Shah. Integrated planning of distribution networks : interactions between land use, transport and electric vehicle charging demand. In *23rd International Conference on Electricity Distribution, Lyon, France*, 2015.

[11] G. Bustos-Turu, K. H. van Dam, S. Acha, C. N. Markides, and N. Shah. Simulating residential electricity and heat demand in urban areas using an agent-based modelling approach. In *2016 IEEE International Energy Conference (ENERGYCON)*, pages 1–6. IEEE, 2016.

[12] M. V. Butz, O. Sigaud, and P. Gérard. Internal models and anticipations in adaptive learning systems. In *Anticipatory Behavior in Adaptive Learning Systems, Foundations, Theories, and Systems*, pages 86–109, 2003. doi : 10.1007/978-3-540-45002-3_6. URL `https://doi.org/10.1007/978-3-540-45002-3_6`.

[13] M. V. Butz, O. Sigaud, and P. Gérard, editors. *Anticipatory Behavior in Adaptive Learning Systems, Foundations, Theories, and Systems*, volume 2684 of *Lecture Notes in Computer Science*, 2003. Springer. ISBN 3-540-40429-5. doi : 10.1007/b11711. URL `https://doi.org/10.1007/b11711`.

[14] L. Calderoni, D. Maio, and P. Palmieri. Location-aware mobile services for a smart city : Design, implementation and deployment. *JTAER*, 7(3) :74–87, 2012. URL `http://www.jtaer.com/dec2012/Calderoni_p7.pdf`.

[15] C. Callender. *What makes time special ?* Oxford University Press, 2017.

[16] T. Carron. *Des Systèmes Multi-Agents temporels pour des systèmes industriels dynamiques*. PhD thesis, École nationale supérieure des mines de Saint-Étienne, France, 2001. URL https://tel.archives-ouvertes.fr/tel-00818319.

[17] C. Castelfranchi, R. Falcone, and M. Piunti. Agents with anticipatory behaviors : To be cautious in a risky environment. In *ECAI 2006, 17th European Conference on Artificial Intelligence, August 29 - September 1, 2006, Riva del Garda, Italy, Including Prestigious Applications of Intelligent Systems (PAIS 2006), Proceedings*, pages 693–694, 2006.

[18] A. C. Chaouche, A. E. Fallah-Seghrouchni, J.-M. Ilié, and D. E. Saïdouni. Smart Agent Foundations : From Planning to Spatio-temporal Guidance. In *Enablers for Smart Cities*, pages 33–63. John Wiley & Sons, Inc, Aug. 2016. doi : 10.1002/9781119329954.ch3. URL https://hal.archives-ouvertes.fr/hal-01366800.

[19] C. W. Choo. - it2000 : Singapore's vision of an intelligent island. In P. Droege, editor, *Intelligent Environments*, pages 49 – 65. North-Holland, Amsterdam, 1997. ISBN 978-0-444-82332-8. doi : https://doi.org/10.1016/B978-044482332-8/50006-8. URL http://www.sciencedirect.com/science/article/pii/B9780444823328500068.

[20] R. Courdier, F. Guerrin, F. Andriamasinoro, and J. Paillat. Agent-based simulation of complex systems : application to collective management of animal wastes. *J. Artificial Societies and Social Simulation*, 5(3), 2002. URL http://jasss.soc.surrey.ac.uk/5/3/4.html.

[21] D. David. *Prospective Territoriale par Simulation Orientée Agent.* PhD thesis, La Réunion, 2010.

[22] P. Davidsson. A framework for preventive state anticipation. In *Anticipatory Behavior in Adaptive Learning Systems, Foundations, Theories, and Systems*, pages 151–166, 2003. doi : 10.1007/978-3-540-45002-3_9. URL https://doi.org/10.1007/978-3-540-45002-3_9.

[23] J. Dávila and K. Tucci. Towards a logic-based, multi-agent simulation theory. In *International Conference on Modeling, Simulation and Neural Networks MSNN*, pages 22–24, 2000.

[24] J. Davila and K. Tucci. Towards a logic-based, multi-agent simulation theory. In *International Conference on Modeling, Simulation and Neural Networks MSNN*, pages 22–24, 2000.

[25] A. Doniec, R. Mandiau, S. Piechowiak, and S. Espié. Anticipation based on constraint processing in a multi-agent context. *Autonomous Agents and Multi-Agent Systems*, 17 (2) :339–361, 2008. doi : 10.1007/s10458-008-9048-7. URL https://doi.org/10.1007/s10458-008-9048-7.

[26] A. Drogoul. *De la simulation multi-agents à la résolution collective de problèmes*. PhD thesis, Thesis at University of Paris IV, 1993.

[27] M. Esteva, J. A. Padget, and C. Sierra. Formalizing a language for institutions and norms. In J. C. Meyer and M. Tambe, editors, *Intelligent Agents VIII, 8th International Workshop, ATAL 2001 Seattle, WA, USA, August 1-3, 2001, Revised Papers*, volume 2333 of *Lecture Notes in Computer Science*, pages 348–366. Springer, 2001. doi : 10.1007/3-540-45448-9_26. URL https://doi.org/10.1007/3-540-45448-9_26.

[28] J. Evans, A. Karvonen, A. Luque-Ayala, C. Martin, K. McCormick, R. Raven, and Y. V. Palgan. Smart and sustainable cities ? pipedreams, practicalities and possibilities. *Local Environment*, 24(7) :557–564, 2019. doi : 10.1080/13549839.2019.1624701. URL https://doi.org/10.1080/13549839.2019.1624701.

[29] J. Ferber. *Multi-agent systems : an introduction to distributed artificial intelligence*, volume 1. Addison-Wesley Reading, 1999.

[30] J. Ferber and O. Gutknecht. A meta-model for the analysis and design of organizations in multi-agent systems. In *Proceedings of the Third International Conference on Multiagent Systems, ICMAS 1998, Paris, France, July 3-7, 1998*, pages 128–135, 1998. doi : 10.1109/ICMAS.1998.699041. URL https://doi.org/10.1109/ICMAS.1998.699041.

[31] J. Ferber and J.-P. Müller. Influences and reaction : a model of situated multiagent systems. In *Proceedings of Second International Conference on Multi-Agent Systems (ICMAS-96)*, pages 72–79, 1996.

[32] J. Ferber, F. Michel, and J. Báez-Barranco. AGRE : integrating environments with organizations. In *Environments for Multi-Agent Systems, First International Workshop, E4MAS 2004, New York, NY, USA, July 19, 2004, Revised Selected Papers*, pages 48–56, 2004. doi : 10.1007/978-3-540-32259-7_2. URL https://doi.org/10.1007/978-3-540-32259-7_2.

[33] J. Ferber, T. Stratulat, and J. Tranier. Towards an integral approach of organizations in multi-agent systems. In *Handbook of Research on Multi-Agent Systems - Semantics and Dynamics of Organizational Models.*, pages 51–75. 2009. doi : 10.4018/978-1-60566-256-5.ch003. URL https://doi.org/10.4018/978-1-60566-256-5.ch003.

[34] J. Fleischer, S. Marsland, and J. Shapiro. Sensory anticipation for autonomous selection of robot landmarks. In *Anticipatory Behavior in Adaptive Learning Systems, Foundations, Theories, and Systems*, pages 201–221, 2003. doi : 10.1007/978-3-540-45002-3_12. URL https://doi.org/10.1007/978-3-540-45002-3_12.

[35] R. M. Fujimoto. Time management in the high level architecture. *Simulation*, 71(6) : 388–400, 1998. doi : 10.1177/003754979807100604. URL https://doi.org/10.1177/003754979807100604.

[36] H. P. Galler. *Discrete-time and continuous-time approaches to dynamic microsimulation reconsidered*. National Centre for Social and Economic Modelling, 1997.

[37] B. Gâteau. *Modélisation et Supervision d'Institutions Multi-Agents. (Multi-Agent based Institutions Modelling and Supervision)*. PhD thesis, École nationale supérieure des mines de Saint-Étienne, France, 2007. URL https://tel.archives-ouvertes.fr/tel-00777825.

[38] M. R. Genesereth and N. J. Nilsson. *Logical foundations of artificial intelligence*. Morgan Kaufmann, 1988. ISBN 978-0-934613-31-6.

[39] D. V. Gibson, G. Kozmetsky, and R. W. Smilor. *The technopolis phenomenon : Smart cities, fast systems, global networks*. Rowman & Littlefield, 1992.

[40] R. Giffinger. European smart cities : the need for a place related understanding. *Department of Spatial Development, Infrastructure and Environmental Planning of Vienna university of Technology*, 2011.

[41] A. Gioda. El hierro (canaries) : une île et le choix des transitions énergétique et écologique. *[VertigO] La revue électronique en sciences de l'environnement*, 14(3), 2014.

[42] A. Giordano, G. Spezzano, and A. Vinci. Smart agents and fog computing for smart city applications. In *Smart Cities - First International Conference, Smart-CT 2016, Málaga, Spain, June 15-17, 2016, Proceedings*, pages 137–146, 2016. doi : 10.1007/978-3-319-39595-1_14. URL https://doi.org/10.1007/978-3-319-39595-1_14.

[43] A. Greenfield. *Against the Smart City : A Pamphlet. This is Part I of" The City is Here to Use"*. Do projects, 2013.

[44] M. Hannoun, O. Boissier, J. S. Sichman, and C. Sayettat. Moïse : Un modèle organisationnel pour la conception de systèmes multi-agents. In M. Gleizes and P. Marcenac, editors, *Ingénierie des systèmes Multi-Agents - JFIADSMA 99 - septième journées francophones d'Intelligence Artificielle et systèmes multi-agents, Le Récif, Saint Gilles, Ile de la Réunion, France, November 8-10, 1999*, pages 105–118. Hermès Lavoisier Editions, 1999.

[45] A. J. Heppenstall, N. Malleson, and A. Crooks. "space, the final frontier" : How good are agent-based models at simulating individuals and space in cities ? *Systems*, 4(1) :9, 2016. doi : 10.3390/systems4010009. URL https://doi.org/10.3390/systems4010009.

[46] J. H. Holmes, P. L. Lanzi, W. Stolzmann, and S. W. Wilson. Learning classifier systems : New models, successful applications. *Inf. Process. Lett.*, 82(1) :23–30, 2002. doi : 10.1016/S0020-0190(01)00283-6. URL https://doi.org/10.1016/S0020-0190(01)00283-6.

[47] S. Holzhauer. *Developing a Social Network Analysis and Visualization Module for Repast Models*, volume 4. kassel university press GmbH, 2010.

[48] J. F. Hübner, J. S. Sichman, and O. Boissier. A model for the structural, functional, and deontic specification of organizations in multiagent systems. In G. Bittencourt and G. L. Ramalho, editors, *Advances in Artificial Intelligence, 16th Brazilian Symposium on Artificial Intelligence, SBIA 2002, Porto de Galinhas/Recife, Brazil, November 11-14, 2002, Proceedings*, volume 2507 of *Lecture Notes in Computer Science*, pages 118–128. Springer, 2002. doi : 10.1007/3-540-36127-8_12. URL https://doi.org/10.1007/3-540-36127-8_12.

[49] J. Innes and R. Kouhy. *The Activity-Based Approach*, pages 243–274. Palgrave Macmillan UK, London, 2011. ISBN 978-0-230-35327-5. doi : 10.1057/9780230353275_10. URL https://doi.org/10.1057/9780230353275_10.

[50] E. Ismagilova, D. L. Hughes, Y. K. Dwivedi, and K. R. Raman. Smart cities : Advances in research - an information systems perspective. *Int J. Information Management*, 47 :88–100, 2019. doi : 10.1016/j.ijinfomgt.2019.01.004. URL https://doi.org/10.1016/j.ijinfomgt.2019.01.004.

[51] J. Jiang. Opera+ : a model for context-aware organizational interactions in virtual organizations. In M. L. Gini, O. Shehory, T. Ito, and C. M. Jonker, editors, *International conference on Autonomous Agents and Multi-Agent Systems, AAMAS '13, Saint Paul, MN, USA, May 6-10, 2013*, pages 1435–1436. IFAAMAS, 2013. URL http://dl.acm.org/citation.cfm?id=2485263.

[52] R. Jordan, M. H. Birkin, and A. J. Evans. An agent-based model of residential mobility : Assessing the impacts of urban regeneration policy in the EASEL district. *Computers, Environment and Urban Systems*, 48 :49–63, 2014. doi : 10.1016/j.compenvurbsys.2014.06.006. URL https://doi.org/10.1016/j.compenvurbsys.2014.06.006.

[53] L. P. Kaelbling, M. L. Littman, and A. W. Moore. Reinforcement learning : A survey. *J. Artif. Intell. Res.*, 4 :237–285, 1996. doi : 10.1613/jair.301. URL https://doi.org/10.1613/jair.301.

[54] O. Klein. *Modélisation et représentations spatio-temporelles des déplacements quotidiens urbains : Application à l'aire urbaine Belfort-Montbéliard*. PhD thesis, Strasbourg 1, 2007.

[55] J. Lee and H. Lee. Developing and validating a citizen-centric typology for smart city services. *Government Information Quarterly*, 31(Supplement-1) :S93–S105, 2014. doi : 10.1016/j.giq.2014.01.010. URL https://doi.org/10.1016/j.giq.2014.01.010.

[56] S. Liao, X. Chen, Y. Qian, and L. Shen. *Comparative Analysis of the Indicator System for Guiding Smart City Development*, pages 575–594. Springer Singapore, Singapore, 2017. ISBN 978-981-10-0855-9. doi : 10.1007/978-981-10-0855-9_51. URL http://dx.doi.org/10.1007/978-981-10-0855-9_51.

[57] M. Longo, M. Roscia, and G. C. Lazaroiu. Innovating multi-agent systems applied to smart city. 2014.

[58] S. Luke, C. Cioffi-Revilla, L. Panait, and K. Sullivan. Mason : A new multi-agent simulation toolkit. In *Proceedings of the 2004 swarmfest workshop*, volume 8, pages 316–327. Michigan, USA, 2004.

[59] R. Maia, M. Silva, Araújo, and U. Nunes. Electric vehicle simulator for energy consumption studies in electric mobility systems. In *2011 IEEE Forum on Integrated and Sustainable Transportation Systems*, pages 227–232. IEEE, 2011.

[60] P. Mathieu, J. Routier, and Y. Secq. RIO : roles, interactions and organizations. In V. Marík, J. P. Müller, and M. Pechoucek, editors, *Multi-Agent Systems and Applications III, 3rd International Central and Eastern European Conference on Multi-Agent Systems, CEEMAS 2003, Prague, Czech Republic, June 16-18, 2003, Proceedings*, volume 2691 of *Lecture Notes in Computer Science*, pages 147–157. Springer, 2003. doi : 10.1007/3-540-45023-8_15. URL https://doi.org/10.1007/3-540-45023-8_15.

[61] F. Michel. *Formalisme, outils et éléments méthodologiques pour la modélisation et la simulation multi-agents. (Formalism, tools and methodological elements for the modeling and simulation of multi-agents systems).* PhD thesis, Montpellier 2 University, France, 2004. URL https://tel.archives-ouvertes.fr/tel-01610063.

[62] N. Minar, R. Burkhart, C. Langton, M. Askenazi, et al. The swarm simulation system : A toolkit for building multi-agent simulations. 1996.

[63] Q. T. Nguyen. *Plate-forme de simulation pour l'aide à la décision : application à la régulation des systèmes de transport urbain. (Simulation platform for decision support : application to the regulation of urban transportation systems).* PhD thesis, University of La Rochelle, France, 2015. URL https://tel.archives-ouvertes.fr/tel-01280109.

[64] J. Nigon, N. Verstaevel, J. Boes, F. Migeon, and M. Gleizes. Smart is a matter of context. In P. Brézillon, R. M. Turner, and C. Penco, editors, *Modeling and Using Context - 10th International and Interdisciplinary Conference, CONTEXT 2017, Paris, France, June 20-23, 2017, Proceedings*, volume 10257 of *Lecture Notes in Computer Science*, pages 189–202. Springer, 2017. doi : 10.1007/978-3-319-57837-8_15. URL https://doi.org/10.1007/978-3-319-57837-8_15.

[65] M. J. North, T. R. Howe, N. T. Collier, and J. Vos. The repast simphony runtime system. In *Agent 2005 Conference on Generative Social Processes, Models, and Mechanisms. Argonne, Illinois, USA : Argonne National Laboratory.* Citeseer, 2005.

[66] M. J. North, N. T. Collier, J. Ozik, E. R. Tatara, C. M. Macal, M. J. Bragen, and P. Sydelko. Complex adaptive systems modeling with repast simphony. *CASM*, 1 :3, 2013. doi : 10.1186/2194-3206-1-3. URL https://doi.org/10.1186/2194-3206-1-3.

[67] J. Odell, H. V. D. Parunak, and M. Fleischer. The role of roles in designing effective agent organizations. In *Software Engineering for Large-Scale Multi-Agent Systems, Research Issues and Practical Applications [the book is a result of SELMAS 2002]*, pages 27–38, 2002. doi : 10.1007/3-540-35828-5_2. URL https://doi.org/10.1007/3-540-35828-5_2.

[68] J. Odell, H. V. D. Parunak, M. Fleischer, and S. Brueckner. Modeling agents and their environment. In *Agent-Oriented Software Engineering III, Third International Workshop, AOSE 2002, Bologna, Italy, July 15, 2002, Revised Papers and Invited Contributions*, pages 16–31, 2002. doi : 10.1007/3-540-36540-0_2. URL https://doi.org/10.1007/3-540-36540-0_2.

[69] H. V. D. Parunak. "go to the ant" : Engineering principles from natural multi-agent systems. *Annals OR*, 75 :69–101, 1997. doi : 10.1023/A\%3A1018980001403. URL https://doi.org/10.1023/A%3A1018980001403.

[70] D. Payet, R. Courdier, T. Ralambondrainy, and N. Sébastien. Le modèle à temporalité : pour un équilibre entre adéquation et optimisation du temps dans les simulations agent. In *Systemes Multi-Agents, Articulation entre l'individuel et le collectif - JFSMA 2006 - Quatorzieme journees francophones sur les systemes multi-agents, Annecy, France, October 18-20, 2006*, pages 63–76, 2006.

[71] D. Payet, R. Courdier, T. Ralambondrainy, and N. Sébastien. Le modèle à temporalité : pour un équilibre entre adéquation et optimisation du temps dans les simulations agent. In *Systemes Multi-Agents, Articulation entre l'individuel et le collectif - JFSMA 2006 - Quatorzieme journees francophones sur les systemes multi-agents, Annecy, France, October 18-20, 2006*, pages 63–76, 2006.

[72] G. Pezzulo, M. V. Butz, O. Sigaud, and G. Baldassarre. From sensorimotor to higher-level cognitive processes : An introduction to anticipatory behavior systems. In *Anticipatory Behavior in Adaptive Learning Systems, From Psychological Theories to Artificial Cognitive Systems [4th Workshop on Anticipatory Behavior in Adaptive Learning Systems, ABiALS 2008, Munich, Germany, June 26-27, 2008]*, pages 1–9, 2008. doi : 10.1007/978-3-642-02565-5_1. URL https://doi.org/10.1007/978-3-642-02565-5_1.

[73] M. Piunti, C. Castelfranchi, and R. Falcone. Surprise as shortcut for anticipation : Clustering mental states in reasoning. In *IJCAI 2007, Proceedings of the 20th International Joint Conference on Artificial Intelligence, Hyderabad, India, January 6-12, 2007*, pages 507–512, 2007. URL `http://ijcai.org/Proceedings/07/Papers/080.pdf`.

[74] T. Ralambondrainy. *Observation de simulations multi-agents à grande échelle*. PhD thesis, La Réunion, 2009.

[75] T. Ralitera and R. Courdier. Toward smart island simulation application - A case study of reunion island. In *Highlights of Practical Applications of Cyber-Physical Multi-Agent Systems - International Workshops of PAAMS 2017, Porto, Portugal, June 21-23, 2017, Proceedings*, pages 457–469, 2017. doi : 10.1007/978-3-319-60285-1_39. URL `https://doi.org/10.1007/978-3-319-60285-1_39`.

[76] T. Ralitera, M. Ferard, G. Bustos-Turu, K. H. van Dam, and R. Courdier. Steps towards simulating smart cities and smart islands with a shared generic framework - A case study of london and reunion island. In *SMARTGREENS 2017 - Proceedings of the 6th International Conference on Smart Cities and Green ICT Systems, Porto, Portugal, April 22-24, 2017.*, pages 329–336, 2017. doi : 10.5220/0006371203290336. URL `https://doi.org/10.5220/0006371203290336`.

[77] T. Ralitera, N. Aky, D. Payet, and R. Courdier. Steps towards a balance between adequacy and time optimization in agent-based simulations - A practical application of the temporality model time scheduling approach. In *Proceedings of 8th International Conference on Simulation and Modeling Methodologies, Technologies and Applications, SIMULTECH 2018, Porto, Portugal, July 29-31, 2018.*, pages 160–166, 2018. doi : 10.5220/0006904601600166. URL `https://doi.org/10.5220/0006904601600166`.

[78] T. Ralitera, N. Aky, D. Payet, and R. Courdier. Steps towards scalable agent-based simulation model : Impact of the time scheduling approach. In *3rd International Workshop on Agent-Based Modelling of Urban Systems Systems (ABMUS2018)*, 2018.

[79] T. Ralitera, D. Payet, N. Aky, and R. Courdier. The temporality model time scheduling approach : A practical application. In *Multi-Agent-Based Simulation XIX - 19th International Workshop, MABS 2018, Stockholm, Sweden, July 14, 2018, Revised Selected Papers*, pages 115–125, 2018. doi : 10.1007/978-3-030-22270-3_9. URL `https://doi.org/10.1007/978-3-030-22270-3_9`.

[80] T. Ralitera, D. Payet, and R. Courdier. Toward a temporal environment for multi-agent simulation. *J. Commun.*, 14(7) :607–613, 2019. doi : 10.12720/jcm.14.7.607-613. URL `https://doi.org/10.12720/jcm.14.7.607-613`.

[81] Q. Reynaud. *Une architecture hybride et flexible pour agents virtuels en environnement urbain : problématiques de la composition de comportements et de l'anticipation. (A hybrid and flexible agent architecture for urban simulations : behavior composition and anticipation issues)*. PhD thesis, Pierre and Marie Curie University, Paris, France, 2014. URL `https://tel.archives-ouvertes.fr/tel-01142548`.

[82] J.-Y. Rochoux. Les technologies de l'information et de la communication dans l'océan indien, Oct 2013. URL `https://www.cairn.info/revue-hermes-la-revue-2002-1-page-471.htm?contenu=resume`.

[83] C. Rolland-May. *Evaluation des territoires : concepts, modèle, méthodes*. Hermès science publications, 2000.

[84] M. Roscia, M. Longo, and G. C. Lazaroiu. Smart city by multi-agent systems. In *2013 International Conference on Renewable Energy Research and Applications (ICRERA)*, pages 371–376. IEEE, 2013.

[85] S. Russell and P. Norvig. *Intelligence artificielle : Avec plus de 500 exercices*. Pearson Education France, 2010.

[86] T. Saunders, P. Baeck, et al. Rethinking smart cities from the ground up. *London : Nesta*, 2015.

[87] G. Schlesinger. The similarities between space and time. *Mind*, 84(334) :161–176, 1975.

[88] R. S. Sutton and A. G. Barto. *Reinforcement learning - an introduction*. Adaptive computation and machine learning. MIT Press, 1998. ISBN 978-0-262-19398-6. URL http://www.worldcat.org/oclc/37293240.

[89] T. Sweda and D. Klabjan. An agent-based decision support system for electric vehicle charging infrastructure deployment. In *2011 IEEE Vehicle Power and Propulsion Conference*, pages 1–5. IEEE, 2011.

[90] M. Tambe. Implementing agent teams in dynamic multiagent environments. *Appl. Artif. Intell.*, 12(2-3) :189–210, 1998. doi : 10.1080/088395198117820. URL https://doi.org/10.1080/088395198117820.

[91] T. Támos. Anticipatory systems : Robert rosen. *Automatica*, 23(1) :128–129, 1987. doi : 10.1016/0005-1098(87)90124-5. URL https://doi.org/10.1016/0005-1098(87)90124-5.

[92] M. van Steen and B. Leiba. Smart cities. *IEEE Internet Computing*, 23(1) :7–8, Jan 2019. ISSN 1941-0131. doi : 10.1109/MIC.2018.2887182.

[93] I. Velontrasina. *No English title available*. Theses, Université de la Réunion, Apr. 2019. URL https://tel.archives-ouvertes.fr/tel-02468232.

[94] R. Vuaridel. Le rôle du temps et de l'espace dans le comportement économique. *Revue économique*, 10(6) :809–837, 1959.

[95] G. Weiss. *Multiagent systems : a modern approach to distributed artificial intelligence*. MIT press, 1999.

[96] D. Weyns, A. Omicini, and J. Odell. Environment as a first class abstraction in multiagent systems. *Autonomous Agents and Multi-Agent Systems*, 14(1) :5–30, 2007. doi : 10.1007/s10458-006-0012-0. URL https://doi.org/10.1007/s10458-006-0012-0.

[97] C. Yin, Z. Xiong, H. Chen, J. Wang, D. Cooper, and B. David. A literature survey on smart cities. *Science China Information Sciences*, 58(10) :1–18, 2015. ISSN 1869-1919. doi : 10.1007/s11432-015-5397-4. URL http://dx.doi.org/10.1007/s11432-015-5397-4.

[98] C. Yin, Z. Xiong, H. Chen, J. Wang, D. Cooper, and B. David. A literature survey on smart cities. *SCIENCE CHINA Information Sciences*, 58(10) :1–18, 2015. doi : 10.1007/s11432-015-5397-4. URL https://doi.org/10.1007/s11432-015-5397-4.

[99] M. Zargayouna. Une représentation spatio-temporelle de l'environnement pour le transport a la demande. *Atelier : Représentation sur le temps et l'espace (RTE 2005), Plate-forme AFIA*, 2005.

[100] B. P. Zeigler. *Theory of Modeling and Simulation*. John Wiley, 1976.

[101] H. Zheng, Y.-J. Son, Y.-C. Chiu, L. Head, Y. Feng, H. Xi, S. Kim, M. Hickman, et al. A primer for agent-based simulation and modeling in transportation applications. Technical report, United States. Federal Highway Administration, 2013.

UNIVERSITÉ DE LA RÉUNION

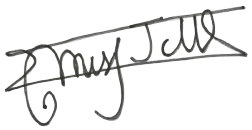
Pôle Recherche
Ecoles doctorales

LETTRE D'ENGAGEMENT DE NON-PLAGIAT

Je, soussigné(e) Ralitera Tahina en ma qualité de doctorant(e) de l'Université de La Réunion, déclare être conscient(e) que le plagiat est un acte délictueux passible de sanctions disciplinaires. Aussi, dans le respect de la propriété intellectuelle et du droit d'auteur, je m'engage à systématiquement citer mes sources, quelle qu'en soit la forme (textes, images, audiovisuel, internet), dans le cadre de la rédaction de ma thèse et de toute autre production scientifique, sachant que l'établissement est susceptible de soumettre le texte de ma thèse à un logiciel anti-plagiat.

Fait à Saint-Denis le : 01/09/2020

Signature :



Extrait du Règlement intérieur de l'Université de La Réunion
(validé par le Conseil d'Administration en date du 11 décembre 2014)

Article 9. Protection de la propriété intellectuelle – Faux et usage de faux, contrefaçon, plagiat

L'utilisation des ressources informatiques de l'Université implique le respect de ses droits de propriété intellectuelle ainsi que ceux de ses partenaires et plus généralement, de tous tiers titulaires de tes droits.
En conséquence, chaque utilisateur doit :
- utiliser les logiciels dans les conditions de licences souscrites ;
- ne pas reproduire, copier, diffuser, modifier ou utiliser des logiciels, bases de données, pages Web, textes, images, photographies ou autres créations protégées par le droit d'auteur ou un droit privatif, sans avoir obtenu préalablement l'autorisation des titulaires de ces droits.

La contrefaçon et le faux
Conformément aux dispositions du code de la propriété intellectuelle, toute représentation ou reproduction intégrale ou partielle d'une œuvre de l'esprit faite ans le consentement de son auteur est illicite et constitue un délit pénal.
L'article 444-1 du code pénal dispose : « Constitue un faux toute altération frauduleuse de la vérité, de nature à cause un préjudice et accomplie par quelque moyen que ce soit, dans un écrit ou tout autre support d'expression de la pensée qui a pour objet ou qui peut avoir pour effet d'établir la preuve d'un droit ou d'un fait ayant des conséquences juridiques ».
L'article L335_3 du code de la propriété intellectuelle précise que : « Est également un délit de contrefaçon toute reproduction, représentation ou diffusion, par quelque moyen que ce soit, d'une œuvre de l'esprit en violation des droits de l'auteur, tels qu'ils sont définis et réglementés par la loi. Est également un délit de contrefaçon la violation de l'un des droits de l'auteur d'un logiciel (…) ».

Le plagiat est constitué par la copie, totale ou partielle d'un travail réalisé par autrui, lorsque la source empruntée n'est pas citée, quel que soit le moyen utilisé. Le plagiat constitue une violation du droit d'auteur (au sens des articles L 335-2 et L 335-3 du code de la propriété intellectuelle). Il peut être assimilé à un délit de contrefaçon. C'est aussi une faute disciplinaire, susceptible d'entraîner une sanction.
Les sources et les références utilisées dans le cadre des travaux (préparations, devoirs, mémoires, thèses, rapports de stage…) doivent être clairement citées. Des citations intégrales peuvent figurer dans les documents rendus, si elles sont assorties de leur référence (nom d'auteur, publication, date, éditeur…) et identifiées comme telles par des guillemets ou des italiques.

Les délits de contrefaçon, de plagiat et d'usage de faux peuvent donner lieu à une sanction disciplinaire indépendante de la mise en œuvre de poursuites pénales.