COBAI : un modèle générique à base d'agents centré sur les contextes et les interactions pour la simulation de comportements

Maëlle Beuret
maelle.beuret@u-bourgogne.fr

Christian Gentil
christian.gentil@u-bourgogne.fr

Irène Foucherot
irene.foucherot@u-bourgogne.fr

Joël Savelli
joel.savelli@u-bourgogne.fr

Laboratoire d'Informatique de Bourgogne, Université de Bourgogne, 21000 Dijon, France

Résumé

Cet article présente un modèle générique à base d'agents pour la simulation de comportements. Le modèle COBAI (Context-Based Agent Interactions) est basé sur un modèle existant dont nous avons conservé les principes fondamentaux : des contextes donnent des comportements aux agents ; les agents peuvent être influencés par plusieurs contextes et choisissent les comportements à adopter en fonction de leurs attributs de personnage. Ce mécanisme permet de contrôler le réalisme à la fois aux niveaux individuel et collectif. Nous proposons un modèle plus complet avec une nouvelle architecture permettant l'exécution de plusieurs comportements simultanés issus d'une combinaison de contextes. Nous introduisons des notions telles que les ressources, outils, modalités et comportements incomplets et nous définissons des groupes d'agents avec distribution de tâches. Nous appliquons le modèle à une simulation de situation de crise développée sur Unity.

Mots-clés : *Architecture d'agent, architecture de comportements, simulation multi-agents, interactions d'agents*

Abstract

This paper presents a generic agent-based model for human behaviors in simulations. COBAI (Context-Based Agent Interactions) is based on a previous model of which we kept the fundamental principles : contexts give behaviors for agents to adopt. Agents can be influenced by several contexts and choose behaviors to adopt depending on their character attributes. This mechanism lets us control realism both at the individual and collective levels. We propose a more comprehensive model with a new behavior architecture, allowing the execution of several simultaneous behaviors resulting from a combination of contexts. We introduce resources, tools, modalities, and incomplete behaviors. We define groups of agents with task distribution. We apply the model to a case study of an emergency crisis developed in Unity.

Keywords: *Agent architecture, behavior architecture, multi-agent simulation, agent interactions*

1 Introduction

Le contexte joue un rôle essentiel dans les comportements humains (ou autres êtres vivants) et les interactions avec l'environnement. Ainsi, il est naturel que le contexte soit devenu un sujet de recherche en intelligence artificielle. Comme les systèmes à base d'agents sont devenus populaires pour la simulation de comportements humains au niveau microscopique, des chercheurs ont exploré l'utilisation de contextes pour influencer les comportements des agents.

Dans ce cadre, Soussi et Savelli ont introduit en 2009 un modèle générique d'animation comportementale basé sur des contextes [15] et des attributs de personnage. Ce modèle permet d'obtenir une variété de comportements cohérents au niveau des individus en utilisant les attributs de personnage (réalisme individuel), tout en contrôlant les comportements collectifs par la force des contextes (réalisme collectif). Ces éléments combinés permettent d'obtenir des comportements globaux réalistes (avec un niveau de granularité dépendant des choix des concepteurs). Ces mécaniques sont prometteuses car elles permettent de représenter une grande variété de situations. Cependant, le modèle présentait certaines limites. Alors que plusieurs contextes pouvaient influencer un agent, un comportement ne pouvait résulter que d'un seul contexte. Cela empêche la prise en compte d'éléments externes, comme la présence d'autres agents, dans un comportement. Des comportements plus élaborés pourraient résulter d'une combinaison

de contextes, notamment la coopération entre agents. Cet article présente un nouveau modèle basé sur les principes fondamentaux des travaux de Soussi et Savelli et traitant ces limites. Nous introduisons une nouvelle architecture de comportements. En utilisant cette architecture, nous présentons une définition de groupes avec distribution de tâches. Dans le cadre d'un partenariat avec le Centre d'Enseignement des Soins d'Urgence de Dijon (CESU21), nous visons à nous assurer que le modèle est adapté à la simulation de situation de crise pour un jeu sérieux visant à entraîner les décideurs. Nous appliquons le modèle conceptuel à un cas d'étude du personnel de la sécurité publique (pompiers, médecins, policiers, infirmiers) et civils (victimes et badauds) dans une situation de crise.

Cet article se présente comme suit. Après avoir introduit le sujet, la deuxième section présente une brève revue des travaux connexes. La troisième section introduit notre modèle COBAI (*Context-Based Agent Interactions*) et détaille chacun de ses éléments. La quatrième section présente une application de COBAI au cas d'étude précédemment mentionné. La cinquième section commente certains concepts utilisés. La dernière section conclut cet article et présente les futurs travaux.

2 Travaux connexes

Un comportement (humain ou animal) est fortement dépendant de circonstances apportées par son environnement. En intelligence artificielle, notamment dans les systèmes à base d'agents, cela a été traduit par certains auteurs par la notion de contexte. Le contexte donne des informations à propos des circonstances d'une action pour qu'elle soit plus pertinente. Dans les systèmes à base d'agents, il est également utilisé pour améliorer l'efficacité de la sélection de comportement, comme affirmé par Turner [16], nécessitant une architecture simple pour sélectionner un comportement pertinent dans une situation donnée. Ces observations ont mené à différentes approches de modélisation de comportements d'agents basés sur les contextes.

Il existe une grande variété de représentations du contexte dans les architectures de systèmes à base d'agents. Par exemple, les schémas contextuels dans le modèle CMB (*Context-Mediated Behaviors*) [16] ou le raisonnement par contexte appliqué aux systèmes à base d'agents [3, 6, 13, 12]. Dans les modèles EASI (*Environment as Active Support of Interaction*) [14] et, plus récemment utilisé, EASS (*Environment as Active Support for Simulation*) [1, 2], les contextes regroupent les entités externes à l'agent qui interviennent dans des filtres déterminant si un agent peut exécuter une action. Il n'existe ainsi aucune définition universelle du contexte dans ce domaine. L'idée principale est d'inclure des informations contextuelles, c'est-à-dire des informations qui dépendent de variables telles que la localisation, le temps, ou des points d'intérêt dans l'environnement.

Une autre manière plus indirecte de représenter les contextes, mais similaire à notre approche, est l'affordance telle que définie par Gibson [5]. L'affordance est la capacité d'un objet ou d'un environnement à transmettre des actions potentielles qui y sont liées. Bien qu'il fut initialement introduit en 1977, ce concept est toujours utilisé dans des travaux récents [9, 8, 7]. Une vision similaire à la nôtre a été présentée dans [17], où les actions sont fortement dépendantes de la position dans l'espace.

L'idée d'utiliser l'environnement pour stocker les interactions réalisables entre les agents a été introduite par Kubera *et al.* avec le modèle IODA [11, 10].

Notre approche peut être considérée comme une représentation de l'affordance utilisant des entités que nous appelons contextes. Cependant, l'architecture de comportements ainsi que d'autres concepts clefs du modèle tels que les attributs de personnage et les ressources, diffèrent des modèles existants basés sur l'affordance. Les autres modèles basés sur la notion de contexte ont une définition de celle-ci différente de la nôtre, que nous développerons dans la section suivante.

3 Context-Based Agent Interactions

COBAI est basé sur un modèle basé sur les contextes [15], qui fait intervenir trois notions principales : les notions d'agent, de contexte et de règle de comportement. Les agents n'ont pas les moyens d'agir par eux-mêmes, ce sont les contextes qui leur donnent des comportements qu'ils pourront potentiellement adopter. Les agents sélectionnent les comportements les plus appropriés avant de les exécuter. Le rôle de ce modèle était d'obtenir un réalisme au niveau des agents individuels tout en gardant le contrôle des comportements collectifs lorsque cela s'avère nécessaire. Ce contrôle permet d'assurer que les comportements globaux sont cohérents. Cependant, le modèle présentait un pro-

blème majeur que nous avons traité dans COBAI : un comportement ne pouvait résulter de plusieurs contextes simultanés, limitant ainsi la variété et le réalisme des comportements. Il était notamment impossible de faire coopérer des agents. Dans cette section, nous introduisons la nouvelle architecture de comportements.

Nous avons représenté les concepts du modèle et leurs relations dans un diagramme sur la Figure 1. Chaque concept sera détaillé dans cette section.

3.1 Agents

Suivant les mêmes principes que [15], les agents ne peuvent agir sans être soumis à des contextes. Leur rôle est de traiter l'influence et les comportements venant des contextes auxquels ils sont soumis. Chaque agent a des attributs de personnage qui peuvent représenter leurs caractéristiques, par exemple dans le cas de simulation de personnages humains, leurs capacités (comme la force ou l'endurance) et leur personnalité (comme l'exubérance ou encore l'agressivité). Un attribut de personnage a un nom, une tendance et une valeur dans l'intervalle $[0, 100]$, comme décrit dans [15]. Une tendance est la valeur que prend l'attribut lorsqu'aucun contexte ne l'influence. Les attributs de personnage, définis par le concepteur en fonction de l'application, permettent de s'assurer que plusieurs agents dans les mêmes conditions extérieures (soumis aux mêmes contextes) n'adopteront pas toujours les mêmes comportements.

Les contextes peuvent influencer la valeur des attributs de personnage d'un agent (mais pas sa tendance). Par exemple, un contexte entourant une enceinte jouant une musique pourrait augmenter l'enthousiasme d'un agent si celui-ci se trouve sous l'influence du contexte.

Pour des raisons de clarté, nous avons ajouté dans COBAI un espace de travail lié à l'agent. Il s'agit d'une représentation de la situation globale de l'agent : elle contient tous les contextes auxquels il est soumis. L'agent y traite ces contextes et les règles de comportements qui y sont liées. L'espace de travail est détaillé dans la section 3.4.

3.2 Contextes

Dans [15], les contextes sont des éléments fondamentaux qui gèrent toutes les interactions des agents avec leur environnement (y compris les autres agents). Ils servent de médiateurs entre l'environnement et les agents. Ainsi, toute information pertinente pour les agents (non exhaustivement : les obstacles, les événements, les autres agents, les objets) est associée à un contexte contenant les comportements appropriés. Dans une application, les contextes sont définis par le concepteur et peuvent ensuite être dynamiquement instanciés par les agents au cours de la simulation. Un contexte peut être localisé, comme un objet, ou non, comme une instruction donnée par un autre agent. Prenons par exemple le contexte *Incendie*. Les agents peuvent réagir à la perception du contexte, c'est pourquoi un contexte contient les comportements pertinents pour la situation qu'il représente. Ainsi, il est associé à des règles de comportement. Suivant l'exemple précédent, le contexte *Incendie* contient les comportements suivants : *Fuir*, *AppelerPompiers*, *Éteindre*. Les prémisses de la règle de comportement vont déterminer quel comportement l'agent adopte.

Un contexte possède une force dans l'intervalle $[0, 100]$. Les agents vont privilégier les comportements venant des contextes qui ont la plus grande force. Par exemple, considérons un agent soumis au contexte *Incendie* de force 90 et au contexte *Panneau* (représentant un panneau publicitaire) de force 30. Si l'agent satisfait les prémisses d'une règle de comportement portée par le contexte *Incendie*, il va exécuter ce comportement plutôt qu'un comportement concurrent venant du contexte *Panneau*. En effet, dans une situation réelle, le danger représenté par l'incendie est prioritaire sur l'information fournie par le panneau publicitaire. Le concepteur doit donc choisir la force des contextes de manière pertinente pour éviter les situations incohérentes telles qu'un agent observant un panneau publicitaire pendant un incendie. La force permet également de pondérer l'influence des contextes sur la valeur des attributs de personnage des agents. Par exemple, le contexte *Incendie* pourrait augmenter la valeur de l'attribut *Peur* des agents qui y sont soumis, en fonction des valeurs de la force du contexte et de la tendance de l'attribut pour chaque agent.

Un contexte peut être localisé ou non localisé. Un contexte localisé a une position dans l'espace et une zone d'influence qui représente les points de l'espace auxquels les agents sont sous son influence (sont soumis au contexte). Un contexte non localisé contient une liste révisée dynamiquement des agents sous son influence.

Un contexte localisé peut être associé à un agent.

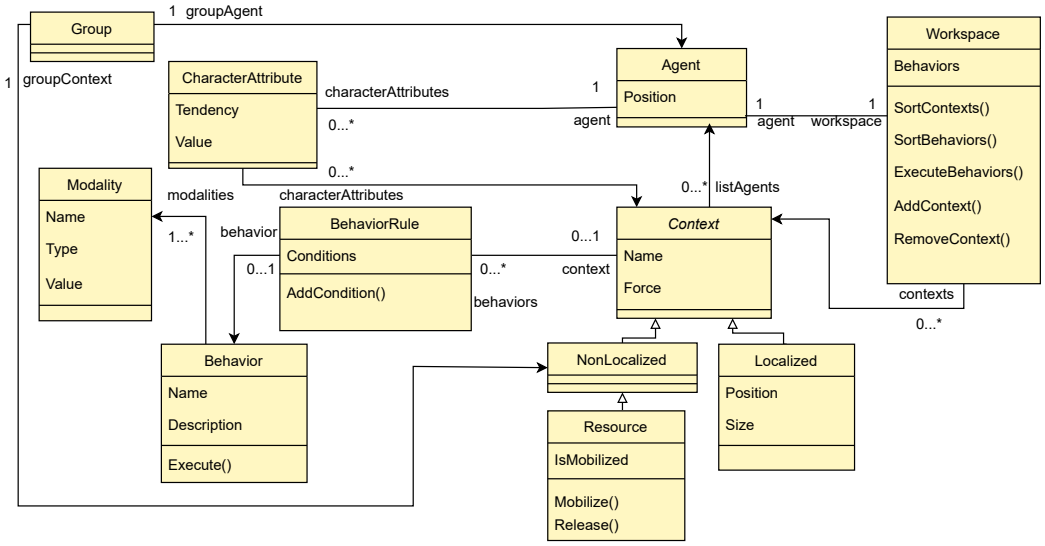

FIGURE 1 – Diagramme conceptuel de COBAI.

Sa position est alors à tout instant celle de l'agent. Un tel contexte peut représenter, par exemple, l'influence d'un agent sur les autres agents. Par exemple, dans [15], les collisions sont gérées par des contextes associés à chaque agent.

Dans COBAI, nous avons introduit un changement mineur dans la définition des contextes : la force d'un contexte peut désormais être une fonction mathématique. Cela permet de faire dépendre la force de la perception qu'un agent a du contexte. Elle peut dépendre par exemple des attributs de personnage de l'agent, ou de la distance à laquelle il se trouve de la source du contexte dans le cas d'un contexte localisé. Par exemple, un contexte peut être perçu comme plus fort lorsqu'un agent est proche de sa source, et plus faible au fur et à mesure qu'il s'éloigne.

3.3 Comportements

Dans [15], un comportement est toute action qu'un agent réalise dans la simulation. Il peut être visible pour l'utilisateur (animation graphique) ou non (modification des données de la simulation, par exemple de la force d'un contexte).

La principale contribution de COBAI est une nouvelle manière de gérer les comportements. Nous avons introduit les nouveaux concepts de règle de comportement incomplète, modalité, ressource et outil. Ces concepts seront détaillés dans cette section.

Le principe général de la nouvelle architecture de comportements est de pouvoir combiner des comportements issus de contextes différents. Cela permet de prendre en compte plusieurs éléments de la situation dans laquelle se trouve l'agent, au lieu de choisir uniquement l'élément le plus pertinent. Plusieurs comportements peuvent être exécutés simultanément. Les conflits sont alors réglés par la gestion des ressources de l'agent.

Comme les comportements dépendent des contextes, ils doivent y être attachés. Ainsi, tout comportement dans la simulation fait partie d'une règle de comportement. Une règle de comportement comporte des prémisses (comme dans une règle logique), un identifiant de comportement, des modalités (incluant le code exécutable du comportement), et au moins un contexte associé. Une modalité est tout élément (donnée, script) nécessaire à l'exécution du comportement. Par exemple, un comportement *SeDéplacer* nécessite un script, une position, et une vitesse. L'association entre les comportements et leurs modalités est réalisée au niveau de la conception par l'identifiant du comportement. Pour exécuter un comportement, les conditions suivantes doivent être respectées. Chaque modalité doit être instanciée, un agent doit être sous l'influence du contexte associé et les prémisses doivent être satisfaites pour cet agent. Une règle de comportement dans laquelle au moins une modalité n'est pas instanciée est une règle de comportement incomplète. Nous utilisons le formalisme suivant pour représenter les règles de comportement :

$$[premisses]IdComportement(modalités)$$

$\{VariationsComportement()\}$

Dans le cas le plus simple, un contexte contient une règle de comportement complète. Selon ses attributs de personnage, un agent soumis à ce contexte a toutes les modalités pour exécuter le comportement correspondant. Un exemple simple de cette situation est un contexte non localisé contenant la règle de comportement suivante : $[]SeDéplacer(destination = maison)\{Marcher()\}$, où $maison$ est une variable contenant la position de la maison de l'agent. Un agent uniquement soumis à ce contexte va simplement marcher jusqu'à sa maison. Cependant, dans de nombreux cas dans COBAI, plusieurs contextes contiennent des règles de comportements incomplètes liées au même comportement. Un agent doit être soumis à plusieurs contextes complémentaires pour exécuter le comportement. Les règles de comportement incomplètes vont alors être combinées dans l'espace de travail de l'agent pour former une règle de comportement complète.

Les contextes non localisés peuvent représenter des ressources. Une ressource est un moyen accessible à l'agent (par exemple, ses bras et ses jambes) pour agir, dont l'accès est concurrent, de manière similaire à Lamarche *et al.* [4]. L'utilisation des ressources résout partiellement la compatibilité entre les comportements. Un contexte représentant une ressource contient des règles de comportements incomplètes nécessitant cette ressource, avec le script correspondant mais sans l'intégralité des modalités. Cela représente les compétences de l'agent, des comportements que l'agent peut adopter si la situation est appropriée. Lorsqu'un agent exécute un comportement C utilisant une ressource R, R est mobilisée, empêchant d'autres comportements nécessitant R de s'exécuter. Lorsque C est interrompu, R est libérée, les autres comportements peuvent donc à nouveau l'utiliser. Reprenons l'exemple précédent d'un agent rentrant chez lui en marchant. Au lieu de rencontrer un seul contexte contenant le comportement avec le script pour marcher et la destination, l'agent dispose d'une ressource $Jambes$ avec une règle de comportement incomplète (règle 1) $[]SeDéplacer(destination)\{Marcher()\}$. Lorsqu'il rencontre un contexte contenant une règle de comportement correspondante (règle 2) $[]SeDéplacer(destination = maison)\{\}$, l'agent va combiner les deux règles de comportement dans son espace de travail pour obtenir une nouvelle règle de comportement complète : $[]SeDéplacer(destination = maison)\{Marcher()\}$. Maintenant, considérons que les agents ont un attribut de personnage $Sportif$. Dans leur ressource $Jambes$, ils pourraient avoir deux règles de comportement incomplètes : $[Sportif < 80]SeDéplacer(destination)\{Marcher()\}$ et $[Sportif >= 80]SeDéplacer(destination)\{Courir()\}$. Avec les prémisses que nous avons ajoutées, les agents vont adopter une version différente du comportement $SeDéplacer$ en fonction de la valeur de leur attribut de personnage $Sportif$ lorsqu'ils rencontrent le contexte complémentaire portant la règle de comportement correspondante $[]SeDéplacer(destination = maison)\{\}$. Certains vont courir, d'autres vont marcher jusqu'à leur maison.

Nous avons défini un type de ressource spécifique : les outils. Un outil représente un objet (par exemple, un téléphone) qu'un agent peut utiliser pour exécuter un comportement (par exemple, appeler quelqu'un sur le téléphone). Deux différences sont à noter entre les ressources ordinaires et les outils. Les outils sont généralement associés à une entité physique de la simulation et nécessitent de mobiliser d'autres ressources (suivant l'exemple précédent, un bras pour utiliser le téléphone).

Cette nouvelle architecture de comportements permet la représentation de situations plus complexes telles que nous pouvons en rencontrer dans la vie réelle. Un agent peut exécuter plusieurs comportements si ceux-ci sont compatibles, c'est-à-dire s'ils utilisent des ressources différentes. Des comportements peuvent provenir d'interactions indirectes entre plusieurs contextes au lieu de toujours provenir d'un seul.

3.4 Espace de travail de l'agent

L'architecture de comportements que nous avons introduite dans COBAI a rendu le procédé de sélection de comportement plus complexe en ajoutant des opérations (trouver des règles de comportement correspondantes, vérifier que chaque modalité est instanciée et combiner les règles de comportements). Ainsi, il est pertinent de le séparer de l'agent pour la clarté du processus de conception d'application. Nous avons ajouté un espace de travail attaché à chaque agent pour traiter les règles de comportement.

L'espace de travail stocke tous les contextes auxquels l'agent est soumis et les trie par force décroissante. À partir de la liste des règles de comportement associées à ces contextes ainsi

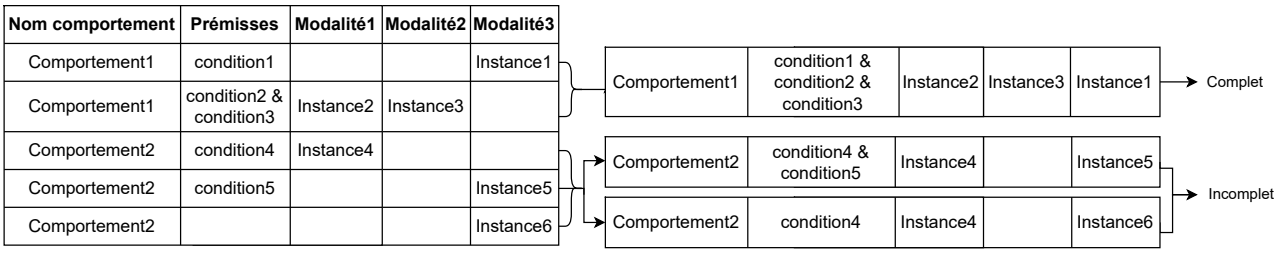

FIGURE 2 – Processus de combinaison de règles de comportement incomplètes permettant d'obtenir des règles de comportement complètes et incomplètes.

triés, il combine des règles de comportement incomplètes afin d'en obtenir des règles complètes qu'il pourra exécuter. Pour cela, il identifie les règles de comportement incomplètes portant les mêmes identifiants et vérifie qu'elles sont unifiables (chaque modalité ne doit être instanciée que dans une seule règle de comportement). Pour unifier les règles de comportement incomplètes, il doit d'abord combiner les prémisses de chaque règle en une unique expression logique. L'agent doit remplir chaque condition de chaque prémisse. Ainsi, les prémisses de chaque règle sont concaténées avec un opérateur logique ET entre elles. Il crée une nouvelle règle de comportement contenant toutes les instances de modalités des règles considérées. Après cette étape, certaines règles de comportement peuvent encore être incomplètes et seront alors ignorées pour l'étape suivante du traitement. Ce procédé est illustré par la figure 2.

En parcourant les règles de comportement complètes triées par ordre décroissant de la force de leurs contextes associés, l'agent exécute chaque comportement si les conditions suivantes sont remplies :

- L'agent satisfait les prémisses de la règle de comportement ;
- Les ressources associées à la règle de comportement sont libres (non mobilisées).

Soit l'exemple suivant : un médecin dans le poste médical avancé. En tant que médecin, l'agent possède une compétence donnée par une ressource partagée par tous les agents de type Médecin : $[TypeAgent = Médecin]TraiterVictime(victime, médicament)\{DonnerMédicament()\}$.
L'agent a également un outil, une trousse médicale, portant la règle de comportement incomplète suivante : $[]TraiterVictime(victime, médicament = TrousseMédicale)\{\}$. Dans le poste médical, plusieurs victimes attendent un traitement, chacune ayant un contexte associé portant la règle de comportement incomplète correspondante : $[]TraiterVictime(victime = celleVictime, médicament)\{\}$. L'état de santé de la victime pondère la force de chaque contexte. Plus leur état de santé est grave, plus la force devient élevée. L'agent doit choisir une victime à traiter en premier. Il va privilégier le contexte avec la plus grande force. Avec la trousse médicale, la ressource venant du groupe des Médecins et le contexte de la victime, la règle de comportement complète suivante peut être exécutée :
$[TypeAgent = Médecin]TraiterVictime(victime = celleVictime, médicament = TrousseMédicale)\{DonnerMédicament()\}$.
Comme l'agent remplit les prémisses, il peut exécuter le comportement.

3.5 Groupes

Les groupes peuvent représenter une variété de structures impliquant plusieurs agents. Un groupe peut décrire une structure sociale (par exemple, une famille ou un groupe d'amis ou de collègues), un rassemblement circonstanciel (par exemple, une file d'attente ou un événement), ou une structure dédiée à une tâche spécifique. Nous pouvons distinguer deux types principaux de groupes : organisés et non organisés. Dans les groupes organisés, les agents ont des rôles au sein du groupe.

Bien que les groupes soient mentionnés dans [15], il s'agit seulement d'ajouter une propriété aux agents contenant le nom du groupe. Ainsi, cette propriété peut être utilisée dans des prémisses de règles de comportement. Cependant, cela ne permet ni la coopération entre les agents, ni des comportements de groupe, seulement de limiter certains comportements à des catégories d'agents spécifiques.

Par nature, un contexte réunit les agents qui y sont soumis. Il est alors naturel d'utiliser les contextes comme base des groupes dans COBAI.

Un contexte seul peut représenter un groupe circonstanciel non organisé où tous les agents exécutent le même comportement. Cela vaut pour les rassemblements autour d'un point d'intérêt (par exemple, une oeuvre d'art dans un musée) ou une action (file d'attente, spectateurs d'un événement culturel), un événement (une alarme par exemple) ou une tâche ponctuelle (comme chercher des victimes autour d'un accident). Le contexte porte alors les comportements pertinents pour la situation. En suivant les exemples précédents, il s'agirait de : regarder une oeuvre d'art, suivre la file, applaudir lors d'un événement culturel et se déplacer autour de la zone d'accident pour chercher des victimes.

Cependant, un groupe nécessite d'être plus explicite lorsqu'il représente une structure conceptuelle telle qu'une équipe, un groupe organisé (une armée, par exemple, avec des sous-groupes) ou lorsqu'il est plus naturel de donner des comportements à un niveau collectif qu'à des agents individuels.

Dans COBAI, ces groupes explicites sont représentés par un agent et un contexte non localisé qui lui est associé (un agent-groupe et un contexte-groupe). L'agent-groupe permet au groupe d'être influencé par des contextes et traite les comportements pour distribuer des tâches aux agents qui sont rassemblés par le contexte-groupe. Chaque agent dans le groupe est soumis au contexte-groupe. Ce mécanisme permet au groupe de fournir des comportements à ses membres.

Les comportements de groupe peuvent être plus complexes que lorsqu'ils n'impliquent qu'un seul agent. Le cas le plus simple est la distribution égale de tâches à chaque membre du groupe. Cependant, particulièrement dans les groupes organisés, certaines tâches nécessitent d'être divisées en différentes sous-tâches qui doivent être accomplies par des rôles spécifiques dans le groupe. Le groupe doit avoir au moins une ressource associée pour gérer ces deux cas. Elle doit contenir des règles de comportements incomplètes avec les scripts de comportements pour la distribution de tâches aux différents membres du groupe. Nous appelons cette ressource le moteur du groupe.

La Figure 3 montre une représentation de la distribution égale de tâches à tous les membres du groupe sur un exemple de pompiers se déplaçant vers une zone d'accident. De manière plus générique, le moteur du groupe contient une compétence qui correspond à une règle de comportement incomplète fournie par un contexte localisé auquel le groupe est soumis. Le comportement complet obtenu en les combinant clone la règle de comportement issue du contexte localisé dans le contexte-groupe. Les membres du groupe ont accès à une ressource contenant une règle de comportement incomplète correspondante qui peut donc être combinée avec la règle de comportement clonée.

La Figure 4 montre une représentation de la distribution des tâches avec des rôles sur un exemple de division des pompiers en deux groupes ayant des rôles différents. Tout comme pour le cas précédent, le moteur du groupe et un contexte localisé contiennent des règles de comportements incomplètes qui se correspondent. Au lieu de cloner une règle de comportement, le comportement obtenu va créer de nouveaux groupes pour chaque rôle avec le moteur approprié et sélectionner des agents du groupe principal pour les ajouter aux nouveaux groupes.

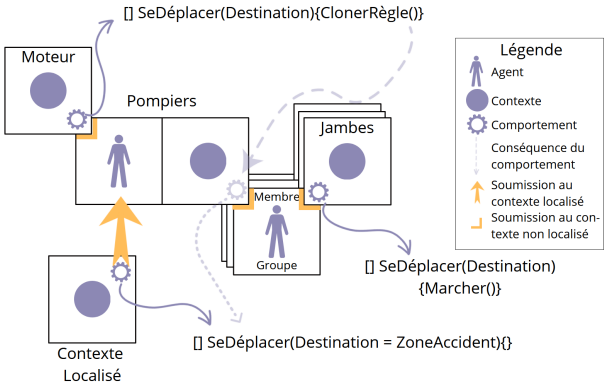

FIGURE 3 – Représentation schématique d'une distribution de tâches uniforme au sein d'un groupe sur l'exemple du déplacement d'un groupe de pompiers vers une zone d'accident.

4 Résultats

Nous appliquons COBAI à une étude de cas de situation de crise, une situation riche et complexe qui présente des interactions variées. En utilisant COBAI, nous représentons des pompiers, des policiers, des médecins, des infirmiers, des victimes et des badauds dans le cas d'une explosion de bâtiment.

Nous implémentons le modèle dans le moteur de jeux Unity. Le moteur de jeu fournit des outils pour créer des environnements tridimensionnels avec la physique et un système de navigation intégrés, qui ne figurent pas dans les points d'intérêt centraux de notre recherche.

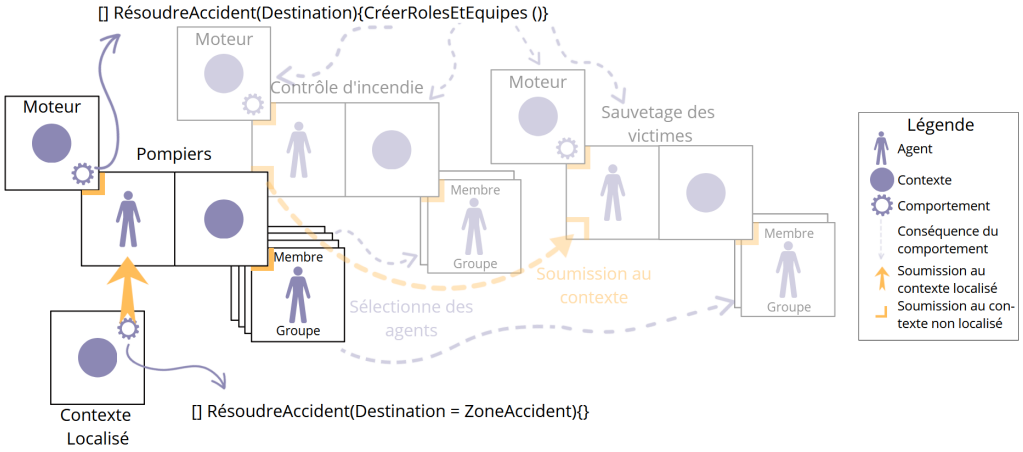

FIGURE 4 – Représentation schématique d'une distribution de tâches complexe au sein d'un groupe avec des rôles, sur l'exemple de pompiers se divisant en deux sous-groupes : contrôle d'incendie et sauvetage des victimes.

La simulation contient une zone d'accident représentée par un contexte localisé, où nous plaçons des victimes dont l'état de santé est variable. Chaque victime a un contexte associé contenant des règles de comportement pour que les pompiers, les infirmiers et les médecins puissent prendre soin d'elle.

Nous donnons à nos agents les ressources suivantes : jambes, bras, tronc et tête. Elles donnent les compétences nécessaires pour que les agents puissent accomplir les tâches. Ces compétences peuvent être génériques (non exhaustivement, marcher, courir) ou spécifiques à la fonction de l'agent (par exemple, donner des médicaments ou arrêter une personne). Outre ces ressources, nous ajoutons divers outils (par exemple des trousses médicales et des brancards) que les agents peuvent utiliser.

En calibrant soigneusement les forces des contextes, nous pouvons créer un séquençage dynamique de comportements. La ressource *Jambes* donne aux agents la règle de comportement incomplète $[]SeDéplacer(destination)\{Marcher()\}$. Les pompiers sont soumis à un contexte non localisé de force faible (30) pour leur permettre de se déplacer vers la zone d'accident. Ce contexte comporte donc la règle de comportement incomplète complémentaire $[]SeDéplacer(destination = ZoneAccident)\{\}$, où *ZoneAccident* contient les coordonnées de la zone où a eu lieu l'accident. La zone d'accident est un contexte localisé avec une force moyenne (60). Comme sa force est plus élevée que celle des contextes non localisés, lorsque les pompiers entrent dans sa zone d'influence, ils commencent à chercher des victimes aléatoirement dans la zone d'accident. Chaque victime a un contexte associé avec une force élevée (80). Lorsqu'un pompier entre dans la zone d'influence du contexte associé à une victime, comme la force est plus élevée que celle des autres contextes, le pompier va se diriger vers la victime. Lorsqu'il l'atteint, l'un de ses comportements sera de diminuer considérablement la force du contexte associé à la victime pour éviter d'attirer d'autres pompiers. Lorsque le pompier atteint la victime, il la récupère et la ramène au poste médical avancé. La Figure 5 présente une capture d'écran de la situation dans la zone d'accident.

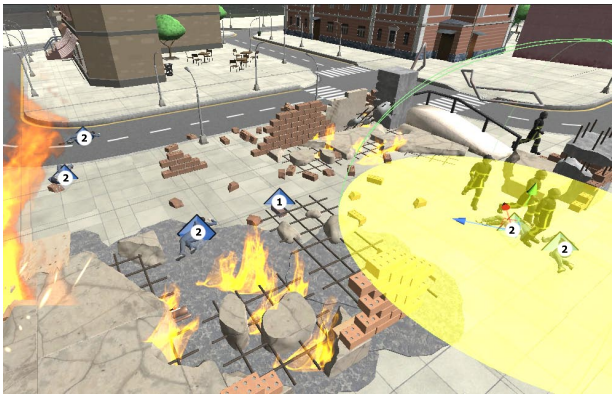

FIGURE 5 – Capture d'écran en mode éditeur de la zone d'accident dans la simulation. La zone d'accident est un contexte désigné par (1), et chaque victime a un contexte associé désigné par (2). Le cercle jaune représente la zone d'influence du contexte, associé à la victime, sélectionné sur la droite.

En utilisant COBAI dans Unity, nous obtenons des contextes dynamiques et des comportements avec séquençage naturel (les comportements s'enchaînent de manière cohérente par rapport à la situation). Les contextes peuvent être créés ou déplacés au cours de la simulation, auxquels les agents vont réagir en fonction de leurs priorités relatives modélisées par leurs forces respectives.

5 Discussions

Définir de multiples contextes complémentaires contenant des règles de comportement incomplètes peut paraître laborieux. Cette technique permet cependant de représenter différentes circonstances de comportements qui participent au réalisme de la simulation. De plus, elles sont simples à mettre en oeuvre pour le concepteur, car elles représentent naturellement des éléments conceptuels issus de l'analyse d'une situation réelle. Par exemple, nous pouvons distinguer un ou plusieurs contextes pour représenter :

- Un besoin qui ordonne d'adopter un comportement sans détailler comment le réaliser. Par exemple, un agent a besoin de se rendre au supermarché. Il peut y aller à pied, en voiture, ou en utilisant d'autres modes de transport. Ce contexte ne fournit aucune information concernant le moyen de transport.
- Des ressources, dont des outils, ajoutent du réalisme, permettant aux agents d'exécuter des comportements similaires différemment en fonction de leurs ressources. En suivant l'exemple précédent, cela fournit le moyen de transport : à pied, en voiture, en vélo ou en bus.
- Des circonstances : pour un comportement donné, elles donnent les informations nécessaires. Il peut s'agir, entre autres, d'une localisation spatiale ou temporelle, d'un objet ou une entité sur laquelle l'agent va appliquer le comportement. Dans l'exemple du supermarché, le contexte pourrait contenir l'emplacement du supermarché.

Concernant la compatibilité des comportements, comme mentionné par Larmarche *et al.* [4], n'autoriser qu'un comportement à la fois pour chaque ressource ne règle pas entièrement le problème. Les concepteurs doivent malgré tout être consciencieux lorsqu'ils associent des comportements à des ressources pour éviter des situations d'incohérence entre comportements simultanés. Cependant, notre solution permet d'exécuter plusieurs comportements simultanés avec un mécanisme simple. Nous envisageons l'exploration plus approfondie de ce problème dans nos travaux futurs pour améliorer cette solution.

6 Conclusion

Dans cet article, nous avons présenté un modèle générique à base d'agents pour la simulation de comportements. Nous avons utilisé un modèle précédemment développé par Soussi et Savelli, dans lequel des entités appelées contextes imposent des comportements aux agents. Nous avons amélioré ce modèle, en particulier l'architecture des comportements, lorsque les agents sont soumis à plusieurs contextes.

Nous avons intégré de nouveaux concepts au modèle, tels que les ressources, les outils et les règles de comportement incomplètes. Nous avons également défini le fonctionnement des groupes dans le modèle.

Nos prochains travaux seront consacrés à la mise en place d'une méthodologie de conception d'applications avec COBAI. Nous décrirons le procédé de conception associé au modèle et représenterons des situations classiques telles que l'échange d'information entre agents, la coopération entre agents avec des rôles définis pour une tâche commune, la combinaison de ressources et outils pour effectuer une tâche, ou différents types de groupes (par exemple des foules non structurées, des hiérarchies de groupes, ou des groupes dont les membres ont des rôles précis).

Nous visons à évaluer plus précisément les capacités et les limites du modèle au niveau conceptuel (quelles situations peuvent ou non être représentées par COBAI) et au niveau technique (évolution des performances en fonction de l'accroissement du nombre d'agents).

Nous travaillerons sur l'intégration de notre travail dans un jeu sérieux développé par notre partenaire, le CESU21, pour tester le modèle dans une situation concrète.

Remerciements. Le travail présenté dans cet article a été financé par la Région Bourgogne-Franche-Comté. Nous tenons également à remercier le Centre d'Enseignement des Soins d'Urgence (CESU21) et Medicalem pour leur partenariat sur ce projet de recherche.

Références

[1] Fabien Badeig and Flavien Balbo. Modèle pour l'activation contextuelle : le mo-

dèle eass. In *Proceedings Journées Francophones pour les Systèmes Multi-Agents (JFSMA'2006)*, pages 49–62, 2006.

[2] Fabien Badeig, Flavien Balbo, and Mahdi Zargayouna. Dynamically configurable multi-agent simulation for crisis management. In Gordan Jezic, Yun-Heh Jessica Chen-Burger, Mario Kusek, Roman Šperka, Robert J. Howlett, and Lakhmi C. Jain, editors, *Agents and Multi-agent Systems : Technologies and Applications 2019*, pages 343–352, Singapore, 01 2020. Springer Singapore.

[3] Oana Bucur, Philippe Beaune, and Olivier Boissier. Representing context in an agent architecture for context-based decision making. *CEUR Workshop Proceedings*, 136, 01 2005.

[4] Lamarche Fabrice and Stéphane Donikian. Automatic orchestration of behaviours through the management of resources and priority levels. In *Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems : Part 3*, AAMAS '02, pages 1309–1316, New York, NY, USA, 07 2002. Association for Computing Machinery.

[5] James J. Gibson. The theory of affordances. In John Bransford Robert E Shaw, editor, *Perceiving, acting, and knowing : toward an ecological psychology*, pages pp.67–82. Hillsdale, N.J. : Lawrence Erlbaum Associates, 1977.

[6] Avelino Gonzalez, Brian Stensrud, and Gilbert Barrett. Formalizing context-based reasoning : A modeling paradigm for representing tactical human behavior. *Int. J. Intell. Syst.*, 23 :822–847, 07 2008.

[7] Sajjad Hassanpour and Amir Rassafi. Agent-based simulation for pedestrian evacuation behaviour using the affordance concept. *KSCE Journal of Civil Engineering*, 25, 01 2021.

[8] Franziska Klügl and Sabine Timpf. Towards more explicit interaction modelling in agent-based simulation using affordance schemata. In Stefan Edelkamp, Ralf Möller, and Elmar Rueckert, editors, *KI 2021 : Advances in Artificial Intelligence*, pages 324–337, Cham, 2021. Springer International Publishing.

[9] Franziska Klügl. Using the affordance concept for model design in agent-based simulation. *Annals of Mathematics and Artificial Intelligence*, 78, 09 2016.

[10] Yoann Kubera, Philippe Mathieu, and Sébastien Picault. IODA : An interaction-oriented approach for Multi-Agent Based Simulations. *Journal of Autonomous Agents and Multi-Agent Systems*, 23(3) :303–343, 2011.

[11] Yoann Kubera, Philippe Mathieu, and Sébastien Picault. Interaction-oriented agent simulations : From theory to implementation. In Ghallab, Malik, Spyropoulos, Constantine, Fakotakis, Nikos, Avouris, and Nikos, editors, *18th European Conference on Artificial Intelligence (ECAI'08)*, pages 383–387, Patras, Greece, 01 2008. IOS Press.

[12] Rikke Amilde Løvlid, Solveig Bruvoll, Karsten Brathen, and Avelino Gonzalez. Modeling the behavior of a hierarchy of command agents with context-based reasoning. *The Journal of Defense Modeling and Simulation*, 15(4) :369–381, 2018.

[13] Francesco Riccio, Emanuele Borzi, Guglielmo Gemignani, and Daniele Nardi. Context-based coordination for a multi-robot soccer team. In Luis Almeida, Jianmin Ji, Gerald Steinbauer, and Sean Luke, editors, *RoboCup 2015 : Robot World Cup XIX*, pages 276–289, Cham, 2015. Springer International Publishing.

[14] Julien Saunier, Flavien Balbo, and Fabien Badeig. Environment as active support of interaction. In Danny Weyns, H. Van Dyke Parunak, and Fabien Michel, editors, *International Workshop on Environments for Multi-Agent Systems III*, volume 4389, pages 87–105, Berlin, Heidelberg, 05 2007. Springer Berlin Heidelberg.

[15] Hakim Soussi, Joël Savelli, and Marc Neveu. A platform for the behavioral animation of crowds. In *Proceedings of the 2009 Summer Computer Simulation Conference*, pages 328–336, Vista, CA, 07 2009. Society for Modeling & Simulation International.

[16] Roy M. Turner. Context-mediated behavior for intelligent agents. *International Journal of Human-Computer Studies*, 48(3) :307–330, 1998.

[17] Afoutni Zoubida. *Un modèle multi-agents pour la représentation de l'action située basé sur l'affordance et la stigmergie*. PhD thesis, 09 2015.