Maëlle Beuret, Irène Foucherot, Christian Gentil, Joël Savelli

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

Volume 5, nº 4 (2024), p. 91-115.

https://doi.org/10.5802/roia.88

La Revue Ouverte d'Intelligence Artificielle est membre du Centre Mersenne pour l'édition scientifique ouverte
www.centre-mersenne.org
e-ISSN : 2967-9672

Revue Ouverte d'Intelligence Artificielle
Volume 5, n° 4, 2024, 91-115

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

Maëlle Beuret[a], Irène Foucherot[a], Christian Gentil[a], Joël Savelli[a]

[a] Laboratoire d'Informatique de Bourgogne, Université de Bourgogne, 21000 Dijon France
E-mail : maelle.beuret@u-bourgogne.fr, irene.foucherot@u-bourgogne.fr, christian.gentil@u-bourgogne.fr, joel.savelli@u-bourgogne.fr.

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, affordance.

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 du contexte pour influencer les comportements des agents. En parallèle, d'autres approches permettent de prendre en compte la situation des agents pour que leurs comportements soient pertinents, comme l'affordance ou l'approche par interactions.

Dans ce cadre, Soussi et Savelli ont introduit en 2009 un modèle générique d'animation comportementale basé sur des contextes [20] qui représentent chaque élément

de l'environnement pouvant induire des comportements des agents. Ce modèle permet d'obtenir une hétérogénéité de comportements cohérents au niveau des individus en utilisant des 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 cohérents – 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 présente certaines limites. Alors que plusieurs contextes peuvent influencer un agent, un comportement ne peut 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 dans le cadre de la coopération entre agents.

Cet article présente un nouveau modèle basé sur les principes fondamentaux des travaux de Soussi et Savelli. Notre objectif est de faire évoluer le modèle pour augmenter son expressivité tout en limitant l'ajout de nouvelles notions. Nous introduisons une nouvelle architecture de comportements. Les règles de comportements comprennent de nouveaux éléments, les modalités, qui peuvent ne pas être instanciées, résultant en des règles de comportement incomplètes. Ces règles peuvent être unifiées pour obtenir des règles de comportement complètes exécutables par les agents, permettant ainsi de prendre en compte plusieurs contextes pour un même comportement. En utilisant cette architecture, nous présentons une représentation de comportements collectifs 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 des 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 explique comment représenter des comportements collectifs avec ce modèle. La cinquième section présente une application de COBAI au cas d'étude d'une situation de crise. La sixième 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 proposé par Turner [21], 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*) [21] ou le raisonnement par contexte appliqué aux systèmes à base d'agents [5, 9, 16, 18]. Dans les modèles EASI (*Environment as Active Support of Interaction*) [19] et, plus récemment utilisé, EASS (*Environment as Active Support for Simulation*) [3, 4], 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 de la notion de 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.

Les circonstances des actions sont également représentées par les interactions entre les agents et l'environnement dans certains systèmes à base d'agents. Des modèles existants ont été étendus à l'aide de cette notion, comme le modèle AGR (*Agent-Group-Role*) avec son extension AGRE (*Agent-Group-Role-Environment*) [6]. Dans ce dernier, les environnements physiques et sociaux sont représentés par des espaces respectivement géométriques (*areas*) et sociaux (*groups*). L'importance de l'environnement a notamment été soulevée par Weyns *et. al* [23], qui propose d'en faire une abstraction de première classe dans les systèmes multi-agents. L'environnement peut également être utilisé pour stocker les interactions réalisables entre les agents, comme dans les travaux de Kubera *et al.* avec le modèle IODA [13, 14].

L'affordance, telle que définie par Gibson [8], est une manière de représenter les interactions entre les agents et leur environnement. L'affordance est la capacité d'un objet ou d'un environnement à transmettre des actions potentielles qui y sont liées. Initialement introduit en 1977, ce concept est toujours utilisé dans des travaux récents [10, 11, 12]. Une vision similaire à la nôtre a été présentée dans [24], où les actions sont fortement dépendantes de la position dans l'espace. Dans ces travaux, la vision est cependant inversée : les acteurs (les entités qui représentent les personnes réelles, capables d'agir sur l'environnement) qui exécutent les actions sont considérés comme des entités environnementales, c'est-à-dire qu'elles font partie de l'environnement, plutôt que des agents car ils sont dépourvus de capacités de calcul et d'autonomie. Ce sont les « agents-places », entités pilotant des « places » (zones spatiales localisées), qui calculent eux-mêmes les actions à exécuter par les entités environnementales de manière à ce qu'elles soient cohérentes par rapport à leur situation. Les agents sont donc les entités qui représentent la situation des acteurs et non les acteurs eux-mêmes comme généralement proposé dans les simulations de comportements humains (y compris dans notre modèle).

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. Nous

nous axons en particulier sur l'hétérogénéité des comportements et les comportements collectifs que nous pouvons obtenir avec notre représentation de l'affordance.

3. COBAI : Context-Based Agent Interactions

Le modèle COBAI a pour objectif de permettre de représenter une large variété de situations dans des simulations de comportements (principalement humains). Il est fondé sur un modèle créé par Soussi *et. al.*[20] qui présente une représentation de l'affordance, et vise à en approfondir les notions pour améliorer son expressivité.

COBAI est fondé sur un modèle basé sur les contextes [20], qui fait intervenir trois notions principales : les *agents*, les *contextes* et les *règles 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, selon le principe d'affordance. Les agents sélectionnent les comportements les plus « appropriés » avant de les exécuter. Le rôle de ce modèle était d'obtenir un réalisme au niveau individuel des agents – grâce aux attributs de personnage expliqués dans la sous-section suivante – tout en gardant le contrôle des comportements collectifs lorsque cela s'avère nécessaire – à l'aide de la force des contextes comme détaillé plus loin. Ce contrôle permet d'assurer que les comportements globaux sont cohérents. Cependant, le modèle présentait un problème majeur que nous avons traité dans COBAI : un comportement ne pouvait pas 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 3.1. Chaque concept sera détaillé dans cette section.

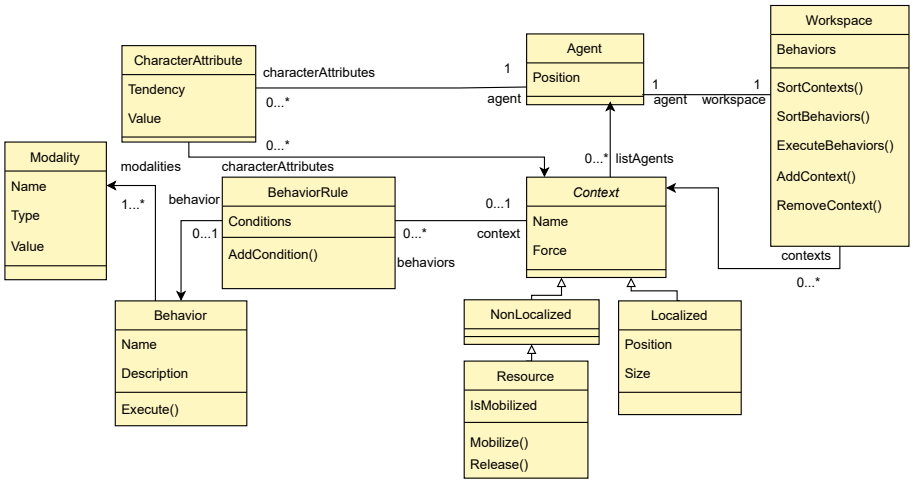

Figure 3.1. Diagramme conceptuel de COBAI.

3.1. Agents

Suivant les mêmes principes que [20], les *agents* ne peuvent agir sans être soumis à des contextes. Leur rôle est de traiter l'influence et les règles de comportement associées aux 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 [20]. 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 leur 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.

Dans COBAI, l'agent comporte un espace de travail qui contient les éléments représentant son état, et en particulier la liste des contextes auxquels il est soumis, qui est mise à jour à chaque instant. 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.

Définition 3.1. — *Un agent* A *est un triplet (*Id*,* attributs*,* attributsPersonnage*) où* Id *est l'identifiant de l'agent,* attributs $= \{\alpha_1, \ldots, \alpha_n\}$ *est un ensemble d'attributs et* attributsPersonnage $= \{\pi_1, \ldots, \pi_n\}$ *est un ensemble d'attributs de personnage. Un attribut* α_i *est un couple (*nom*,* valeur*). Un attribut de personnage* π_i *est un triplet (*nom*,* tendance*,* valeur*) avec* tendance*,* valeur $\in [0, 100]$ *où* tendance *est la valeur par défaut de l'attribut de personnage lorsque celui-ci n'est influencé par aucun contexte, et* valeur *est calculée en fonction de la tendance et des influences des contextes auxquel* A *est soumis sur* π_i.

3.2. Contextes

Dans [20], les *contextes* sont des éléments fondamentaux qui gèrent toutes les interactions des agents avec leur environnement – y compris les autres agents. Les contextes sont donc des entités de la simulation qui représentent notamment l'affordance des objets de l'environnement sur les 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 localisé, comme dans le cas d'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 déterminent quel comportement l'agent peut adopter.

Un contexte possède une force dans l'intervalle $[0, 100]$. Les agents privilégient 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 exécute 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). Cela permet de représenter l'environnement physique des agents, par exemple des objets ou des événements. Un contexte non localisé contient une liste révisée dynamiquement des agents sous son influence. Ces contextes permettent de représenter l'environnement social et interne des agents, par exemple des instructions données par un tiers ou un besoin ressenti par l'agent.

Un contexte localisé peut être associé à un agent. 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 [20], 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 le résultat d'un calcul prenant en compte la situation de chaque agent. Autrement dit, la force d'un contexte n'est plus une constante représentant sa prégnance sur l'ensemble des agents. Elle représente désormais la perception de l'importance d'un contexte pour un agent particulier à un moment donné. 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.

Définition 3.2. — *Un contexte* C *est un tuple (*Nom, Force, Règles, Agents, Influences*) où :*

- Force *: f(*attributs*) est la fonction qui définit sa force en fonction des attributs de l'agent qui le perçoit.* Force *définit la priorité entre les comportements issus de différents contextes et pondère l'effet des* Influences *sur les agents (elle est donc calculée avant celles-ci) ;*
- Règles = $\{r_1, \ldots, r_n\}$ *est un ensemble de règles de comportements associées au contexte (voir définition 3.4). Elles peuvent être divisées en k sous-ensembles* $\text{Règles}_1, \ldots, \text{Règles}_k$ *;*
- Agents = $\{a_1, \ldots, a_n\}$ *est un ensemble dynamique d'agents soumis au contexte. Il peut être divisé en k sous-ensembles* $\text{Agents}_1, \ldots, \text{Agents}_k$ *;*
- Influences = $\{i_1, \ldots, i_n\}$ *est un ensemble d'influences sur les attributs de personnage des agents. Chaque influence* i_i *est un couple* (π_i, v_i) *où* π_i *est un attribut de personnage et* v_i *la valeur associée à son influence.*

3.3. Comportements

Dans [20], 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). Une *règle de comportement* est constituée d'un comportement, de prémisses et d'un contexte associé. Un comportement est exécutable par un agent dès lors que celui-ci valide les prémisses de la règle de comportement. Les auteurs n'ont pas défini de gestion de la compatibilité entre comportements dans le cas où l'agent pourrait en exécuter plusieurs.

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 et ressource. Ces concepts seront détaillés dans cette section.

Le principe général de la nouvelle architecture de comportements est de permettre de combiner des comportements issus de contextes différents. Cela permet de prendre en compte plusieurs paramètres de la situation dans laquelle se trouve l'agent, au lieu de choisir uniquement le plus prégnant. 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 (dont l'une contient 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 des contextes associés 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.

Définition 3.3. — *Un comportement* B *est un triplet (*Id*,* Nom*,* ModalitésAttendues*) où* Id *est l'identifiant de règle de comportement dans lequel il peut être utilisé, nom est le nom du comportement et* ModalitésAttendues = $\{m_{a1}, \ldots, m_{an}\}$ *est un ensemble de modalités nécessaires pour l'exécution du comportement. Chaque modalité attendue* m_{ai} *est un couple (*nom*,* type*).*

Définition 3.4. — *Une règle de comportement* R *est un triplet (*Id*,* Prémisses*,* Modalités*) où* Id *est l'identifiant de la règle,* Prémisses = $\{p_1, \ldots, p_n\}$ *est un ensemble de prémisses et* Modalités = $\{m_1, \ldots, m_n\}$ *est un ensemble de modalités instanciées. Chaque modalité* m_i *est un triplet (*nom*,* type*,* valeur*).* R *est complète si à chaque modalité attendue* m_{ai} *du comportement associé correspond une modalité* m_i *de* R.

Par la suite, nous utilisons la représentation suivante pour les règles de comportement :

$$[premisses]IdComportement(modalités)\{Script()\}. \tag{3.1}$$

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 :

$$[]SeDéplacer(destination = maison)\{Marcher()\}. \tag{3.2}$$

Dans cette règle, $maison$ est une variable contenant la position de la maison de l'agent. Un agent uniquement soumis à ce contexte marche simplement 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 sont alors combinées par unification dans l'espace de travail de l'agent pour former une règle de comportement complète. Un exemple est donné dans le paragraphe suivant.

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.* [15]. 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. Une ressource ne nécessite pas forcément de définir sa force ; en effet, selon l'application, il n'est pas toujours naturel qu'un agent privilégie une ressource plutôt qu'une autre. 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 :

$$[]SeDéplacer(destination)\{Marcher()\}. \tag{3.3}$$

Lorsqu'il rencontre un contexte C contenant une règle de comportement correspondante :

$$[]SeDéplacer(destination = maison)\{\}. \tag{3.4}$$

L'agent unifie les deux règles de comportement dans son espace de travail pour obtenir une nouvelle règle de comportement complète :

$$[]SeDéplacer(destination = maison)\{Marcher()\}. \tag{3.5}$$

Maintenant, considérons que les agents ont un attribut de personnage *Sportif*. Dans leur ressource $Jambes$, ils pourraient avoir deux règles de comportement incomplètes :

$$[Sportif < 80]SeDéplacer(destination)\{Marcher()\}\ ; \tag{3.6}$$

$$[Sportif >= 80]SeDéplacer(destination)\{Courir()\}. \tag{3.7}$$

Avec les prémisses que nous avons ajoutées, les agents adoptent 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 (3.4). Certains courent, d'autres marchent jusqu'à leur maison. Dans les deux cas, la ressource *Jambes* est mobilisée. Elle ne peut donc pas être utilisée par d'autres comportements, à moins que ceux-ci ne proviennent de contextes dont la force est supérieure à celle de C. Dans ce cas, le comportement *SeDéplacer* est interrompu et c'est un autre comportement qui est exécuté, mobilisant la ressource *Jambes*.

L'expressivité du modèle peut être facilement enrichie en introduisant le concept d'outils, des objets utilisables par les agents. Dans la réalité, différents outils peuvent être utilisés pour réaliser la même action de manière différente. Dans COBAI, cela se traduit par l'utilisation de ressources associées à des entités physiques de la simulation, et qui peuvent mobiliser d'autres ressources. Par exemple, un pompier disposant d'une clef peut ouvrir une porte fermée grâce à cet outil, d'une manière différente d'un pompier possédant une hache[(1)].

Cette nouvelle architecture de comportements permet la représentation de situations plus complexes que le modèle initial, 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. Lors de la conception d'applications avec COBAI, il n'est pas

[(1)] L'utilisation des outils a été illustrée par un projet étudiant. La vidéo de ce projet est disponible : `https://youtu.be/5Ld9Y-irUys`

nécessaire de prévoir toutes les combinaisons de règles de comportement possibles : il suffit de les prévoir localement, et les règles s'unifieront en fonction des circonstances émergentes au cours de la simulation.

3.4. Interactions agent-contexte

L'architecture de comportements que nous avons introduite dans COBAI a rendu le procédé de sélection de comportement plus complexe que dans le modèle initial en ajoutant des opérations (trouver des règles de comportement correspondantes, vérifier que chaque modalité est instanciée et unifier les règles de comportements). Nous avons ajouté un espace de travail attaché à chaque agent pour traiter les règles de comportement. Conceptuellement, l'espace de travail est une partie de l'agent. Mais techniquement, on peut représenter l'agent et son espace de travail par deux objets différents pour la clarté du processus de conception d'application.

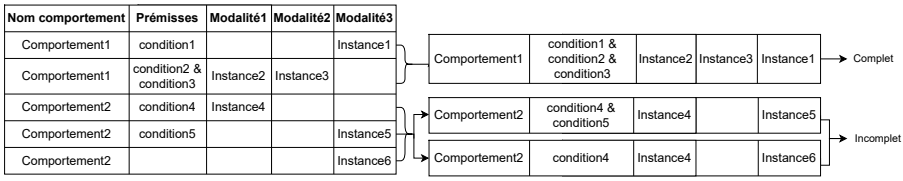

Figure 3.2. Processus d'unification de règles de comportement incomplètes permettant d'obtenir des règles de comportement complètes et de nouvelles règles incomplètes.

Lorsqu'un agent rencontre un contexte, celui-ci ajoute l'agent à l'ensemble des agents qui lui sont soumis. L'agent a alors « conscience » du contexte et l'ajoute dans son espace de travail à la liste des contextes auxquels il est soumis.

L'espace de travail contient à chaque instant la liste des contextes auxquels l'agent est soumis, ordonnée par force décroissante. À partir de la liste des règles de comportement associées à ces contextes ainsi 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 incomplètes portant les mêmes identifiants de comportement 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 conjoindre 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 conjointes. Il crée une nouvelle règle de comportement contenant toutes les instances de modalités des règles considérées (voir algorithme 3. 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 3.2 et décrit par les algorithmes 1 et 2.

Algorithme 1 : ComplèteTout : procédure récursive d'unification des règles de comportements.

Données : R : liste des règles de comportement classées par ordre décroissant de force des contextes, C : liste de comportements à exécuter
Résultat : C : liste de comportements à exécuter complétée

```
si !R.estVide() alors
    complèteTout(R.queueListe, C) ;
    complète(R.têteListe, R.queueListe, C) ;
fin
```

Algorithme 2 : Complète : tente de compléter une règle de comportement courante avec les règles de la liste.

Données : r : règle de comportement courante à compléter, R : liste des règles de comportement classées par ordre décroissant de force des contextes, C : liste de comportements à exécuter
Résultat : C : liste de comportements à exécuter

```
si !R.estVide() alors
    Règle u = unifie(r, R.premier) ;
    si u != null alors
        si u.estComplète() alors
            C.ajouter(u) ;
        fin
        sinon
            complète(u, R.queueListe, C)
        fin
    fin
    sinon
        complète(r, R.queueListe, C);
    fin
fin
```

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

Algorithme 3 : Unifie : unifie deux règles de comportement

```
   Données : R1, R2 : règles de comportement incomplètes
   Résultat : U : règle de comportement unifiée
 1 si R1.identifiant != R2.identifiant alors
 2 |   Renvoyer null
 3 fin
 4 U = nouvelle Règle() ;
 5 U.setModalitésAttendues(R1.modalitésAttendues) ;
 6 pour tous les modalité dans Règle1.modalitésAttendues() faire
 7 |   si R1.modalité != null alors
 8 |   |   si R2.modalité != null alors
 9 |   |   |   Renvoyer null;
10 |   |   fin
11 |   |   sinon
12 |   |   |   U.modalité = R1.modalité ;
13 |   |   fin
14 |   fin
15 |   sinon si R2.modalité != null alors
16 |   |   U.modalité = R2.modalité ;
17 |   fin
18 fin
```

les agents de type Médecin :

$$[TypeAgent = Médecin]TraiterVictime(victime, médicament) \{DonnerMédicament()\}. \quad (3.8)$$

L'agent a également un outil, une trousse médicale, portant la règle de comportement incomplète suivante :

$$[]TraiterVictime(victime, médicament = TrousseMédicale)\{\}. \quad (3.9)$$

Dans le poste médical, plusieurs victimes attendent un traitement, chacune ayant un contexte associé portant la règle de comportement incomplète correspondante :

$$[]TraiterVictime(victime = cetteVictime, médicament)\{\}. \quad (3.10)$$

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 privilégie 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 :

$$[TypeAgent = Médecin]TraiterVictime(victime = cetteVictime, médicament = TrousseMédicale)\{DonnerMédicament()\} \quad (3.11)$$

Comme l'agent remplit les prémisses, il peut exécuter le comportement.

4. Comportements collectifs et groupes

La capacité des agents à réaliser des comportements collectifs est une propriété importante des modèles à base d'agents. On distingue deux approches principales : les approches centrées agent (ascendantes) et les approches centrées organisation (descendantes). Weyns *et al.* [22] soulignent le problème posée par la première approche : les agents ont une double responsabilité (leur fonction dans l'application et l'organisation du système). Des travaux se sont donc centrés sur une approche descendante dans laquelle l'organisation des agents est représentée de manière explicite : les organisations multi-agents [2]. C'est sur ce principe que nous représentons des comportements collectifs avec COBAI.

Bien que les groupes soient mentionnés dans [20], leur gestion se limite seulement à 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 collectifs, seulement de limiter certains comportements à des catégories d'agents spécifiques.

L'objet de cette section est de montrer comment utiliser les éléments du modèle pour représenter des comportements collectifs de groupes d'agents. Plus précisément, il s'agit de représenter avec COBAI un comportement collectif et son articulation avec les comportements des agents qui le constituent, selon une approche descendante, sans ajouter d'élément au modèle.

4.1. Comportements collectifs simples et groupes implicites : les contextes-groupes

Dans COBAI, l'approche consiste à modéliser une dynamique de comportements plutôt que les agents qui les effectuent. Les agents sont avant tout des entités capables d'exécuter des comportements, uniquement distingués par leurs attributs de personnage. Des comportements potentiels, représentés par des règles de comportement, leur sont affectés dynamiquement en fonction de l'environnement, médiatisé dans notre modèle par des contextes.

À un instant donné, un contexte influe sur un ensemble d'agents. Techniquement, il a pour attribut la liste des agents qu'il a sous son influence, cette liste étant remise à jour régulièrement. Il transmet à tous les agents de cette liste l'ensemble des règles de comportements qu'il contient. Sans enrichissement du modèle, cette architecture permet de représenter des comportements collectifs simples, où des agents ont conjointement le même comportement potentiel. C'est par exemple le cas des *boids* de Reynolds [17] (hardes, nuées d'oiseaux, bancs de poissons), ou d'individus qui participent à une manifestation ou qui assistent à un spectacle.

Lorsque l'on considère l'aspect collectif d'un ensemble d'agents soumis à un même contexte, on l'appelle contexte-groupe. Cette notion est uniquement conceptuelle : tout contexte est potentiellement un contexte-groupe. Les groupes d'agents ainsi créés ne

sont pas explicitement identifiés : les agents regroupés sont simplement référencés ensemble par la liste d'agents influencés par le contexte-groupe (qui peut éventuellement fluctuer), comme illustré par la figure 4.1.

FIGURE 4.1. Schéma d'une collection d'agents regroupés par un contexte-groupe.

Notons que les agents influencés par un contexte ont accès à celui-ci et donc potentiellement à ses attributs. Cela permet aux agents d'un même groupe d'avoir connaissance les uns des autres — via la liste des agents sous influence — et de partager des informations. Il est notamment possible de créer un attribut du contexte-groupe pour représenter explicitement un environnement commun aux agents du groupe, auquel ils peuvent accéder en lecture et potentiellement (selon les besoins du concepteur) en écriture.

Avec cette même notion de contexte-groupe, il est possible de constituer des sous-groupes au sein du groupe global : par exemple un groupe d'amis (comportement de cohésion) dans une manifestation (comportement grégaire). Notons que la relation d'inclusion entre groupes n'est pas plus explicite que les groupes eux-mêmes : les agents sont influencés, avec des forces spécifiques, par les différents groupes et sous-groupes auxquels ils appartiennent.

Notons que, dans COBAI, il n'existe pas d'autre moyen de regrouper des agents que de les mettre sous influence d'un même contexte. Tout regroupement d'agents passera donc nécessairement par la mise en place d'au moins un contexte-groupe.

4.2. Comportements et groupes complexes : les contextes-rôles

Pour obtenir un comportement collaboratif au sein d'un groupe, les agents n'ont pas nécessairement tous les mêmes comportements : ils peuvent avoir des comportements différenciés complémentaires – ou rôles – qui concourent au comportement global du groupe.

Par exemple, considérons la construction d'un mur. On peut mobiliser des individus pour préparer un enduit, des manœuvres pour convoyer les matériaux et des maçons pour façonner le mur en enduisant le sommet de la partie déjà réalisée, puis en y disposant un alignement de parpaings (ou autres matériaux de construction comme des briques ou des pierres). De même, en général, l'utilisation d'une machine ou même d'un outil (par exemple un simple brancard) nécessite des rôles particuliers à remplir par des groupes spécifiques d'agents – parfois limités à un simple individu.

Chaque rôle regroupe des agents qui ont conjointement les mêmes comportements, ce qui nous place dans la situation de comportement collectif simple (c'est-à-dire sans collaboration) vu précédemment. En plus du contexte-groupe global, il y a donc un contexte-groupe pour chaque rôle, que nous appellerons contexte-rôle. Tout comme les contextes-groupes, la notion de contexte-rôle n'est que conceptuelle : aucun élément technique n'est ajouté au modèle. Conceptuellement, cette notion de rôle ressemble à ceux définis dans le modèle AGRE [6]. Elle se traduit cependant dans COBAI par des sous-groupes, qui respectent donc les propriétés des groupes, plutôt que par des entités distinctes intervenant dans la définition des groupes.

La configuration d'un comportement collectif peut donc être constituée d'un contexte-groupe global qui regroupe tous les agents concernés par le comportement et des contextes-rôles qui regroupent les agents affectés aux différents rôles (voir figure 4.2). Le contexte-groupe global permet de contenir :

- des règles de comportements éventuelles correspondant à des comportements communs à tous les agents (en plus de leurs rôles propres) ;
- des règles de comportements qui contiennent des modalités de déclenchement des rôles : par exemple, un objet mobilisé, une destination ou une valeur numérique ;
- des informations sur l'avancée du comportement collectif, qui correspond à l'état du groupe, ce qui peut permettre de synchroniser les rôles (assurer une exécution synchronisée ou un ordonnancement cohérent des comportements assurés par chaque rôle).

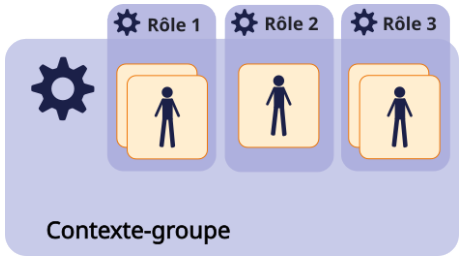

FIGURE 4.2. Schéma du fonctionnement de comportements collectifs avec des contextes-rôles. Les agents, regroupés par un contexte-groupe, sont répartis et soumis à différents contextes-rôles.

Notons qu'un même agent peut être affecté à différents rôles. Dans l'exemple de la construction d'un mur, par exemple, un maçon peut jouer le rôle d'un manœuvre ; il est référencé à la fois dans les deux rôles. Ce sont les forces relatives des contextes-rôles ainsi que les prémisses des règles de comportement de ces contextes qui permettent à l'agent dans cette situation de privilégier dynamiquement un rôle plutôt qu'un autre.

4.3. Gestion des comportements collectifs et matérialisation des groupes : les agents superviseurs

La configuration précédente permet de représenter de nombreuses situations mais comporte des limites. La synchronisation entre les rôles nécessite souvent une supervision globale que ne peuvent pas assurer les agents du groupe. Les groupes doivent également pouvoir eux-mêmes être soumis à des contextes qui portent des comportements à l'échelle du groupe, par exemple se déplacer jusqu'à un emplacement donné. Un tel comportement global nécessite une distribution des rôles dynamique. Par exemple, pour le déplacement d'une armée, les éclaireurs doivent effectuer une reconnaissance préalablement au déplacement d'une troupe. Les sous-groupes correspondant aux rôles ne sont pas forcément constitués au départ, ils peuvent être dynamiquement créés et modifiés. Il faut alors soumettre des agents du groupe global aux contextes-rôles pour remplir chacun des rôles. Les comportements associés aux rôles doivent pouvoir eux-mêmes être décomposés en sous-comportements distincts complémentaires.

Dans l'inventaire précédent, les comportements de gestion du groupe doivent être effectués au niveau global, celui du groupe lui-même. Or, dans le modèle, les comportements ne peuvent être exécutés que par des agents. Pour assurer ces fonctions, il faut donc disposer d'un agent pour représenter le groupe et agir en son nom : c'est l'agent superviseur. Cet agent est virtuel, il n'est pas représenté par une entité physique de la simulation.

Le contexte-groupe global est associé à cet agent, qui est lui-même soumis à un ou plusieurs contextes, appelés moteurs, qui contiennent ses compétences de gestion sous forme de règles de comportement incomplètes. Il peut par exemple y avoir un moteur de distribution des rôles, un moteur de synchronisation des rôles, un moteur de recrutement, ou un moteur qui prend en charge conjointement plusieurs de ces activités si elles ne peuvent pas être distinguées les unes des autres. Il doit notamment y avoir au minimum un moteur de distribution des rôles, contenant la distribution pour chaque comportement de groupe potentiel. Nous généralisons ces différents cas sous le terme de moteur de supervision. Le fonctionnement de comportements collectifs supervisé par un agent est illustré par la figure 4.3.

Cette manière de représenter les comportements collectifs respecte certaines propriétés des agents holoniques tels que présentés dans [1, 7]. Les objectifs des agents sont représentés par des contextes. Les agents individuels sont toujours soumis à des contextes, donc poursuivent leurs objectifs propres, mais également au contexte qui influence les membres du groupe, donc un objectif commun de l'ensemble d'agents qui constituent le groupe. Ils ont donc des comportements communs basés sur des objectifs. Les agents disposent de capacités de groupe améliorées (des actions globales au niveau du groupe constituées des actions des agents individuels membres du groupe). Les membres du groupe peuvent communiquer entre eux via le contexte qui les regroupe.

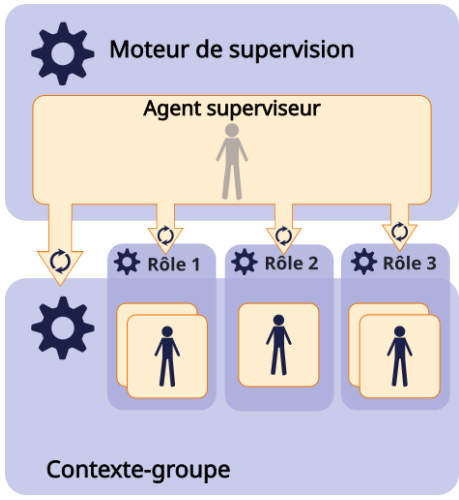

FIGURE 4.3. Schéma du fonctionnement de comportements collectifs supervisés par un agent. Les agents, regroupés par un contexte-groupe, sont répartis et soumis à différents contextes-rôles. Le contexte-groupe et les contextes-rôles sont supervisés et gérés par un agent superviseur doté d'au moins un moteur de supervision.

4.4. MÉTHODOLOGIE DES COMPORTEMENTS COLLECTIFS

Les différentes notions permettant d'obtenir des comportements collectifs ayant été introduites, cette section présente une approche méthodologique de la conception de groupes en fonction des mécanismes nécessaires pour obtenir le fonctionnement attendu.

Dans le cas où l'on souhaite modéliser des comportements collectifs très simples, on peut regrouper les agents simplement à l'aide d'un contexte-groupe. Ainsi, on peut donner à tous les agents soumis à ce contexte des comportements identiques, permettant ainsi d'obtenir des effets de groupe. Par exemple, on peut ainsi représenter un attroupement autour d'un jongleur de rue. Les spectateurs pourront tous adopter des comportements similaires, comme observer ou applaudir. Ils n'ont pas besoin de se répartir des tâches, il n'y a donc pas besoin de contexte-rôle, et il n'est pas nécessaire de gérer des interactions complexes, de la synchronisation ou de la répartition en sous-groupes, donc aucun agent-groupe n'est nécessaire dans ce cas. On parle alors de groupes contextuels, qui sont implicitement formés par un contexte.

DÉFINITION 4.1. — *Un groupe contextuel* G_c *est un contexte* C *(voir définition 3.2).*

Lorsque l'on souhaite représenter un comportement collectif B, qui nécessite l'exécution de n comportements $b_1 \dots b_n$, chacun effectué par un sous-ensemble des membres du groupe, il est nécessaire d'ajouter des contextes-rôles. Chaque contexte-rôle contient un comportement b_i. On peut ainsi représenter, par exemple, un groupe

d'enfants jouant à chat : on distingue le contexte-rôle du chat et celui des souris, chacun portant des comportements différents, pour un comportement collectif global du jeu. Si le groupe ne nécessite pas de synchronisation, de distribution supervisée des rôles (les agents peuvent se les attribuer eux-mêmes) ni de répartition dynamique de tâches dans les rôles, il n'est pas nécessaire d'ajouter d'agent-groupe. Le groupe est donc constitué d'un contexte-groupe et de contextes-rôles. On parle alors de groupes structurés.

Définition 4.2. — *Un groupe structuré* G_{struct} *est un couple (*C, rôles*) où* C *est un contexte (voir définition 3.2) et* rôles = $\{rôle_1, \ldots, rôle_n\}$ *est un ensemble de contextes permettant de répartir les agents soumis à* C *et leur donner des comportements différents pour accomplir un comportement collectif commun.*

Dès lors qu'au moins l'un des mécanismes cités ci-dessus (synchronisation, distribution supervisée des rôles, répartition dynamique de tâches dans les rôles) est essentiel au fonctionnement du groupe, il est nécessaire d'ajouter un agent superviseur pour assurer ces fonctions. Cela permet d'obtenir une grande variété de comportements collectifs complexes. Par exemple, on peut représenter une équipe de pompiers portant un brancard (ce qui nécessite de synchroniser les rôles), une équipe de secours (qui va être amenée à effectuer plusieurs comportements collectifs, qui nécessite donc une répartition dynamique de tâches dans les rôles) ou encore un jeu avec des équipes dont les membres doivent se synchroniser et dans lequel on a une gestion des tours. On parle alors de groupes supervisés.

Définition 4.3. — *Un groupe supervisé* G_{sup} *est un triplet (*A, Moteur, G*) où* A *est un agent superviseur,* Moteur *est un contexte (ou un ensemble de contextes) auquel* A *est soumis contenant les comportements de répartition des tâches, et* G *est un groupe structuré (voir définition 4.2).*

Par ailleurs, l'agent superviseur représente le groupe et peut lui-même être membre d'un autre groupe. Cela permet notamment de décomposer un rôle en sous-rôles. En effet, le contexte rôle correspondant à un comportement qui doit être décomposé en sous-rôles n'aura qu'un agent-groupe sous son influence, celui-ci ayant pour charge de décomposer son rôle en sous-rôles pris en charge par d'autres agents. Cette construction récursive de groupe est illustrée dans la figure 4.4.

Le type d'un groupe n'a pas d'incidence sur les éventuels sous-groupes qu'il peut contenir : un groupe contextuel peut par exemple compter parmi ses membres des agents superviseurs, qui sont donc rattachés à des groupes supervisés. Il en va de même pour tous les types de groupes. La figure 4.5 illustre ce fonctionnement sur un exemple de manifestation à laquelle participent – outre les manifestants individuels – différents groupes d'agents : un groupe contextuel représentant un groupe d'amis, un groupe structuré représentant une famille avec les rôles de parents et enfants et un groupe supervisé représentant les organisateurs de la manifestation séparés en deux rôles (service d'ordre et communication).

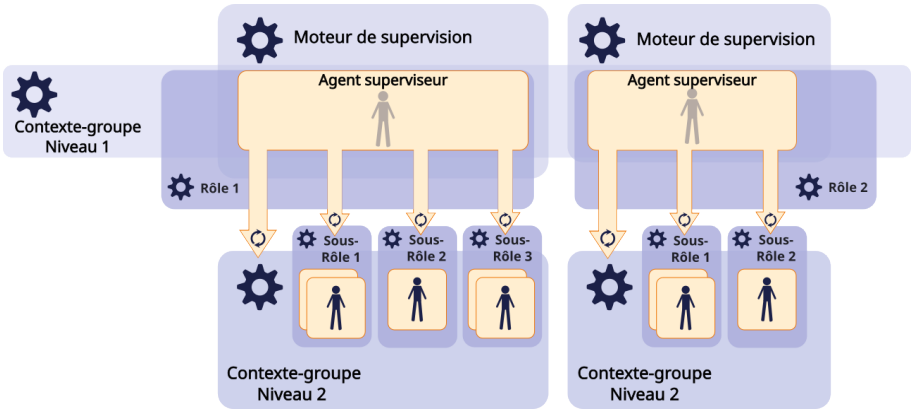

FIGURE 4.4. Illustration de la décomposition d'un rôle en sous-rôles supervisés en mettant comme membres du groupe de niveau 1 (le groupe qui contient les sous-groupes) les agents superviseurs des sous-groupes.

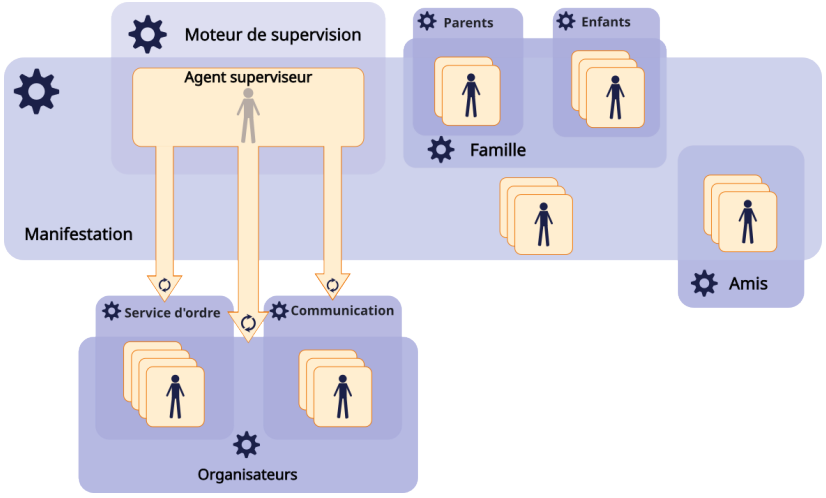

FIGURE 4.5. Exemple de représentation d'une manifestation avec COBAI. Le groupe contextuel *Manifestation* contient des agents (manifestants sans sous-groupe particulier) et différents types de sous-groupes : contextuel (*Amis*), structuré (*Famille*) et supervisé (*Organisateurs*).

5. 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.

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 gauche, bras droit. 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. Dans notre simulation, la ressource *Jambes* donne aux agents la règle de comportement incomplète :

$$[]\mathit{MoveTo}(\mathit{destination})\{\mathit{Walk}()\}. \tag{5.1}$$

Les pompiers, organisés en groupe supervisé contenant des sous-groupes correspondant aux équipes de pompiers, sont soumis à un contexte non localisé de force faible (0) pour leur permettre de se déplacer vers la zone d'accident. Ce contexte comporte donc la règle de comportement incomplète complémentaire :

$$[]\mathit{MoveTo}(\mathit{destination} = \mathit{AccidentZone})\{\}. \tag{5.2}$$

AccidentZone contient les coordonnées de la zone où a eu lieu l'accident. Cette règle de comportement est transmise par l'agent superviseur du groupe *Pompiers* aux sous-groupes *Equipes*, qui la transmettent eux-mêmes aux agents membres du groupe par le biais de leur contexte-groupe. La zone d'accident est un contexte localisé avec une force moyenne (30). Comme sa force est plus élevée que celle des contextes non localisés, lorsque les pompiers entrent dans sa zone d'influence, ils cherchent 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 se dirige vers la victime. Lorsqu'il l'atteint, l'un de ses comportements est 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 (PMA). Une fois la victime déposée au PMA, le pompier n'est plus sous l'influence du contexte de la victime, il cesse donc le comportement qu'il avait adopté. Il exécute le comportement prioritaire (issu du contexte ayant la force la plus élevée) dans la liste des comportements qu'il peut adopter – dans notre simulation, il se rend à nouveau sur la zone d'accident. Une vidéo du fonctionnement de la simulation

est disponible[2]. La figure 5.1 illustre un exemple de fonctionnement de groupe à trois niveaux avec deux équipes de pompiers se dirigeant vers deux zones d'accident. Elle offre une visualisation de la manière dont sont transférées les règles de comportement d'un niveau à un autre dans un groupe supervisé hiérarchique.

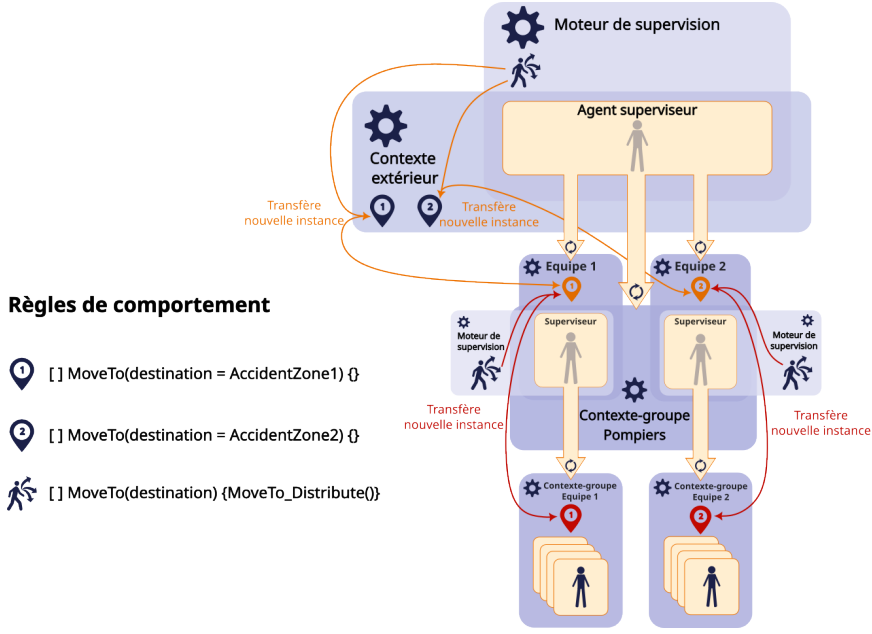

FIGURE 5.1. Schéma du fonctionnement des groupes de pompiers, illustré avec le comportement de déplacement vers deux zones d'accident différentes. Les règles de comportements portant les modalités de destination s'unifient chacune avec la règle de comportement contenant la modalité de script. Le script *MoveTo_Distribute*() transmet par exemple la règle de comportement portant la modalité de destination à la première équipe disponible. Dans la simulation, seule une zone d'accident est actuellement implémentée, la seconde zone étant ici utilisée pour illustrer l'intérêt de la différenciation des équipes de pompiers. Les pompiers membres des équipes pourront par la suite être eux-mêmes répartis en rôles afin de coopérer, par exemple pour porter des brancards.

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 réagissent en fonction de leurs priorités relatives modélisées par leurs forces respectives.

[2]Les animations visibles sont temporaires car nous travaillons actuellement sur l'utilisation de brancards par les équipes de pompiers pour porter les victimes. Elles ne sont donc pas réalistes et visent simplement à montrer le comportement adopté par les agents.
Lien de la vidéo : `https://youtu.be/ROQ__8bBtOY`

6. 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, ces mécanismes sont simples à mettre en oeuvre pour le concepteur, car ils représentent naturellement des éléments conceptuels issus de l'analyse d'une situation réelle. Les situations que nous pouvons représenter avec COBAI sont conceptuellement très riches, ce qui nécessite une certaine complexité de représentation quel que soit le modèle utilisé. 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, qui ajoutent de l'hétérogénéité, 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 qui, pour un comportement donné, 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 applique le comportement. Dans l'exemple du supermarché, le contexte pourrait contenir l'emplacement du supermarché.

Notre approche présente également l'avantage d'obtenir un ordonnancement naturel des comportements en fonction de la force des contextes dont ils sont issus. Ainsi, si un contexte est supprimé ou modifié, cela impacte directement la simulation sans avoir à effectuer d'autres modifications : les agents s'adaptent naturellement à la nouvelle configuration. Le concepteur n'a pas besoin de considérer chaque comportement dans la globalité de la simulation, mais uniquement dans leur contexte local – dans le temps comme dans l'espace. L'ordonnancement naturel des comportements assure alors une cohérence des comportements de tous les agents au niveau global. Cependant, pour que cet ordonnancement se déroule comme souhaité, il est nécessaire de calibrer les forces des contextes de manière cohérente. Les contextes donnant des comportements « par défaut » aux agents ont une force très faible (par exemple entre 0 et 10), ceux qui représentent une situation urgente (danger imminent ou mission cruciale) ont une force très élevée (par exemple entre 80 et 100). Les concepteurs peuvent concevoir des paliers intermédiaires correspondant aux besoins de la simulation.

Comme dans la plupart des systèmes à base d'agents, il existe des situations bloquantes. Dans COBAI, il ne peut pas y avoir de blocage dans le choix des comportements à adopter dans une situation donnée (un ensemble de contextes auquel l'agent est soumis) en raison de l'ordonnancement des comportements décrit par les algorithmes 1, 2 et 3. En revanche, des comportements oscillatoires ou des situations de blocage peuvent apparaître, par exemple sur un carrefour routier avec quatre voitures

qui tentent de s'engager simultanément avec une priorité à droite. Si ces situations bloquantes ne peuvent pas être détectées automatiquement, il est tout de même possible de résoudre ces problèmes dans COBAI en ajoutant des contextes ou encore des groupes supervisés pour gérer les priorités entre les agents ou les comportements une fois le problème identifié.

La possibilité de créer plusieurs instances de règles de comportement portant différentes modalités, avec toutes les unifications possibles de celles-ci, peut générer un nombre important de règles dans le cas d'une simulation complexe avec de nombreux contextes. Cependant, dans la plupart des cas, il n'est pas nécessaire de créer de nombreuses instances, le nombre de règles générées par unification reste donc faible dans ces cas-là.

Concernant la compatibilité des comportements, comme mentionné par Lamarche *et al.* [15], 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.

7. 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 [20], 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 mis en avant deux contributions principales pour améliorer l'expressivité du modèle :

- Une augmentation de l'hétérogénéité des comportements à travers les règles de comportements incomplètes, les ressources – permettant également d'exécuter plusieurs comportements simultanés – et les outils ;
- L'introduction de comportements collectifs dans le modèle, avec les différentes manières de les représenter.

Le modèle COBAI a l'avantage d'être évolutif : la simulation est entièrement dynamique. Grâce au nouveau mécanisme de règles de comportements incomplètes, chaque comportement est modulaire : on peut obtenir des variations dynamiquement au cours de la simulation en combinant des règles de comportements incomplètes différentes avec le même identifiant, en fonction des contextes auquel l'agent est soumis. Des contextes peuvent être ajoutés ou modifiés en cours de simulation, tout comme des règles de comportements, des ressources, des outils ou des groupes d'agents.

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. Nous détaillerons également les manières dont nous pouvons représenter différentes notions qui pourraient s'avérer nécessaires pour le concepteur, comme la perception ou la communication entre agents.

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.

Bibliographie

[1] E. Adam, R. Mandiau & C. Kolski, « Une methode de modelisation et de conception d'organisations organisations multi-agents holoniques », p. 41-75, Hermes, Paris, 2002.

[2] H. Ahmed Abbas, « Organization of Multi-Agent Systems: An Overview », *International Journal of Intelligent Information Systems* **4** (2015), n° 3, p. 46.

[3] F. Badeig & F. Balbo, « Modèle pour l'activation contextuelle : le modèle EASS », in *Proceedings Journées Francophones pour les Systèmes Multi-Agents (JFSMA'2006)*, 2006, p. 49-62.

[4] F. Badeig, F. Balbo & M. Zargayouna, « Dynamically Configurable Multi-agent Simulation for Crisis Management », in *Agents and Multi-agent Systems: Technologies and Applications 2019* (G. Jezic, Y.-H. J. Chen-Burger, M. Kusek, R. Šperka, R. J. Howlett & L. C. Jain, éds.), Springer Singapore, Singapore, 2020, p. 343-352.

[5] O. Bucur, P. Beaune & O. Boissier, « Representing Context in an Agent Architecture for Context-Based Decision Making », in *Proceedings of the CRR'05 Workshop on Context Representation and Reasoning*, vol. 136, CEUR Workshop Proceedings, 2005.

[6] J. Ferber, F. Michel & J. Baez, « AGRE: Integrating Environments with Organizations », in *Environments for Multi-Agent Systems* (D. Weyns, H. Van Dyke Parunak & F. Michel, éds.), Springer Berlin Heidelberg, Berlin, Heidelberg, 2005, p. 48-56.

[7] C. Gerber, J. Siekmann & G. Vierke, « Holonic multi-agent systems », Tech. report, German Research Center for Artificial Intelligence, 1999, `https://www.dfki.de/en/web/research/projects-and-publications/publication/6158`.

[8] J. J. Gibson, « The theory of affordances », in *Perceiving, acting, and knowing: toward an ecological psychology* (R. E. Shaw & J. Bransford, éds.), Lawrence Erlbaum Associates, Hillsdale, N.J., 1977, p. 67-82.

[9] A. Gonzalez, B. Stensrud & G. Barrett, « Formalizing context-based reasoning: A modeling paradigm for representing tactical human behavior », *Int. J. Intell. Syst.* **23** (2008), n° 7, p. 822-847.

[10] S. Hassanpour & A. Rassafi, « Agent-Based Simulation for Pedestrian Evacuation Behaviour Using the Affordance Concept », *KSCE Journal of Civil Engineering* **25** (2021), p. 1433–1445.

[11] F. Klügl & S. Timpf, « Towards More Explicit Interaction Modelling in Agent-Based Simulation Using Affordance Schemata », in *KI 2021: Advances in Artificial Intelligence* (S. Edelkamp, R. Möller & E. Rueckert, éds.), Lecture Notes in Computer Science, vol. 12873, Springer International Publishing, Cham, 2021, p. 324-337.

[12] F. Klügl, « Using the affordance concept for model design in agent-based simulation », *Annals of Mathematics and Artificial Intelligence* **78** (2016), p. 21-44.

[13] Y. Kubera, P. Mathieu & S. Picault, « IODA: An interaction-oriented approach for Multi-Agent Based Simulations », *Journal of Autonomous Agents and Multi-Agent Systems* **23** (2011), n° 3, p. 303-343.

[14] Y. Kubera, P. Mathieu & S. Picault, « Interaction-Oriented Agent Simulations: From Theory to Implementation », in *18th European Conference on Artificial Intelligence (ECAI'08)* (M. Ghallab, C. Spyropoulos, N. Fakotakis & N. Avouris, éds.), Frontiers in Artificial Intelligence and Applications, vol. 178, IOS Press, Patras, Greece, 2008, p. 383-387.

[15] F. Lamarche & S. 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, Association for Computing Machinery, New York, NY, USA, 2002, p. 1309-1316.

[16] R. A. Løvlid, S. Bruvoll, K. Brathen & A. Gonzalez, « Modeling the behavior of a hierarchy of command agents with context-based reasoning », *The Journal of Defense Modeling and Simulation* **15** (2018), nº 4, p. 369-381.

[17] C. W. Reynolds, « Flocks, Herds and Schools: A Distributed Behavioral Model », in *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '87, Association for Computing Machinery, New York, NY, USA, 1987, p. 25–34.

[18] F. Riccio, E. Borzi, G. Gemignani & D. Nardi, « Context-Based Coordination for a Multi-Robot Soccer Team », in *RoboCup 2015: Robot World Cup XIX* (L. Almeida, J. Ji, G. Steinbauer & S. Luke, éds.), Springer International Publishing, Cham, 2015, p. 276-289.

[19] J. Saunier, F. Balbo & F. Badeig, « Environment as Active Support of Interaction », in *International Workshop on Environments for Multi-Agent Systems III* (D. Weyns, H. V. D. Parunak & F. Michel, éds.), vol. 4389, Springer Berlin Heidelberg, Berlin, Heidelberg, 2007, p. 87-105.

[20] H. Soussi, J. Savelli & M. Neveu, « A platform for the behavioral animation of crowds », in *Proceedings of the 2009 Summer Computer Simulation Conference*, Society for Modeling & Simulation International, Vista, CA, 2009, p. 328-336.

[21] R. M. Turner, « Context-mediated behavior for intelligent agents », *International Journal of Human-Computer Studies* **48** (1998), nº 3, p. 307-330.

[22] D. Weyns, R. Haesevoets & A. Helleboogh, « The MACODO Organization Model for Context-Driven Dynamic Agent Organizations », *TAAS* **5** (2010), p. 16.

[23] D. Weyns, A. Omicini & J. Odell, « Environent as a First-Class Abstraction in Multiagent Systems », vol. 14, 2006, p. 5-30.

[24] A. Zoubida, « Un modèle multi-agents pour la représentation de l'action située basé sur l'affordance et la stigmergie », Thèse de doctorat, École doctorale Sciences, Technologies et Santé (Saint-Denis, La Réunion), 09 2015.

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, affordance.
