Contribution à la formation en réalité virtuelle : scénarios collaboratifs et intégration d'humains virtuels collaborant avec des utilisateurs réels

Stéphanie Gerbaud

N° d'ordre: D 08-10

THÈSE

présentée

devant l'INSA de Rennes

pour obtenir

le grade de : DOCTEUR DE L'INSA DE RENNES
Mention INFORMATIQUE

par

Stéphanie GERBAUD

Équipe d'accueil : Bunraku - IRISA
École Doctorale : Matisse
Composante universitaire : INSA

Titre de la thèse :

Contribution à la formation en réalité virtuelle : scénarios collaboratifs et intégration d'humains virtuels collaborant avec des utilisateurs réels

soutenue le 1er octobre 2008 devant la commission d'examen

M. :	Jean-Louis	PAZAT	Président
MM. :	Pascal	GUITTON	Rapporteurs
	Jacques	TISSEAU	
MM. :	Jean-Pierre	JESSEL	Examinateurs
	Guillaume	MOREAU	
M. :	Charles	SAINT-ROMAS	Invité
M. :	Bruno	ARNALDI	Directeur

“*Je pense qu’il est nécessaire de trouver rapidement une solution technique pour libérer le monde de GVT.*”

Charles Saint-Romas, extrait d’un mail d’avril 2006

Remerciements

Je tiens tout d'abord à remercier Bruno Arnaldi, mon directeur de thèse, pour son encadrement exemplaire, son soutien et ses conseils et pour avoir, malgré un emploi du temps surchargé, toujours su se rendre disponible lorsque j'en avais besoin. Je remercie également les membres de mon jury pour avoir accepté de juger mon travail : Jean-Louis Pazat en tant que président, Pascal Guitton et Jacques Tisseau en tant que rapporteurs, Jean-Pierre Jessel et Guillaume Moreau comme examinateurs et Charles Saint-Romas comme membre invité. Merci à vous pour vos remarques constructives.

Je tiens ensuite à remercier toutes les personnes impliquées dans le projet GVT, un projet passionnant et très enrichissant que je ne vais pas lâcher tout de suite ! Merci donc tout particulièrement à la GVT-team rennaise, Nicolas Mollet et Xavier Larrodé. Xavier, ça a été un plaisir de travailler avec toi durant ces 3 années et de partager le même bureau (et aussi le même cobural insupportable ;-)), je tiens à te remercier tout particulièrement pour ton implication dans la préparation de ma démo finale et pour ta patience lorsqu'il s'agissait de répondre à toutes mes (innombrables) questions techniques. Merci également aux brestois : Eric Cazeaux, Frédéric Devillers et aussi Franck Ganier. Franck, merci pour tes encouragements et pour m'avoir initiée à "la courbe d'apprentissage". Merci également aux personnes de Nexter, Charles pour m'avoir laissé une grande liberté dans mes travaux de thèse, Bernard pour sa réactivité (particulièrement lorsqu'il s'agit de prendre des photos dans des situations insolites :-)), Christophe pour avoir appris à harmoniser un char Leclerc avec moi et Camille pour son aide sur ma démo finale.

Je voudrais remercier également tous les zukaris qui font régner une bonne humeur légendaire au sein de l'équipe, contribuant ainsi à rendre le travail à l'IRISA si agréable (et si difficile à quitter, d'ailleurs je n'ai pas réussi...). Merci à tous les non permanents qui se sont succédé, peu de ceux qui étaient là à mon arrivée sont encore là, des nouveaux sont arrivés mais la bonne ambiance qui règne dans le couloir, elle, persiste. Merci donc pour commencer à celui qui emplit régulièrement le bureau de "toï toï" inattendus : Vincent. Au final c'est toi qui a cédé et qui va quitter le bureau avant moi :-P mais sache que ça n'a finalement pas été si déplaisant que ça de t'avoir en face de moi ;-) et je suis forcée de reconnaître que le bureau va être bien triste sans toi... Merci ensuite au bureau d'en face avec leur avion dont la trajectoire n'est pas toujours bien maîtrisée ;-) : Michaël le gourou de l'équipe, fervent adorateur de Chuck Norris, Julien et ses chants joyeux autour d'un feu de camp, Alex et son franc parler ; merci également au bureau du fond : l'éminent Yann R., toujours prêt à nous donner le signal du départ, Yann J., organisateur des très prisés *picnic & ultimate* et Fabien, l'éternel première année. Fabien je tiens à t'adresser un merci tout particulier car c'est toi qui as été à l'origine de tout ça en me

transmettant l'offre pour ma thèse. J'ai été très heureuse de partager avec toi ces 3 années de thèse au sein de l'équipe Bunraku. Merci également au bureau MKM : Yann P., joyeux festayre, Nico C. et ses annonces de marabout, Barth et ses promenades en espace ; merci aussi à Loïc le magicien, gardien du Champomy, à Xavier, notre zukari d'honneur, à Benoît et sa culture ; merci aux anciens, Marwan pour sa gentillesse, Seb pour son humour et Nico P. pour sa bonne humeur ; merci également à Jean, Jonathan, Laurent, Charles, Ludo, Delphine, Kévin et tous les autres zukaris (la liste est encore bien longue). Merci à tous pour ces débats sans fin lancés en pause café (merci au passage à Pierrette et Laurence qui nous ont toujours servi avec le sourire), j'aurais appris des tas de choses, pas toujours très utiles, certes, mais qu'importe... Merci aussi à tous les permanents de l'équipe : Stéphane notre chef d'équipe emblématique, Thierry pour son décryptage de la vie de l'équipe, Valérie pour sa bonne humeur et notre discussion sur l'après-thèse, Marc pour ses conseils lors de mes présentations et plus particulièrement pour m'avoir appris des subtilités sur la langue anglaise qu'on ne trouve pas dans le dictionnaire, Anatole et Julien pour leurs conseils notamment sur ma soutenance, Angélique et Véro pour leur efficacité en tant qu'assistantes, Fabrice, Alain et les autres.

Parce que pour réussir une thèse il faut également savoir se changer les idées, merci à tous mes amis, plus particulièrement à la bande du CSOne pour les discussions, les soirées et les sorties en tout genre, merci à Fabien (encore une fois, parce qu'avant d'être un collègue il était déjà un ami), Stéphane, Grégoire, Fabrice, Olivier, Nico, Solenn, Patricia, Peggy, Morgane, Pascale, Jeff, Fred, Aurélie, Gaël, Laurent et Baptiste. Je sais que je peux compter sur vous et je vous en remercie ! Merci aussi à Marie-Anne pour nos sorties ciné, à Cb pour nos discussions Msn, à Cathy, amie de longue date et puis aux membres du GSR pour m'avoir permis de m'évader sous terre. Sans oublier une pensée pour Marie qui nous a quittés bien trop tôt...

Enfin je tiens à remercier ma famille pour leur soutien même s'ils n'ont pas toujours compris en quoi consistait exactement ma thèse, merci donc à ma maman d'être revenue du sud pour assister à ma soutenance, à mon papa et à mon frère Sylvain, qui le jour même où je soutenais ma thèse commençait lui aussi une nouvelle aventure[1] (je n'ai pas résisté à l'envie de faire un peu de pub au passage ;-)).

Pour finir, merci à tous ceux qui ont contribué, de près ou de loin, à l'aboutissement de ce travail de thèse, merci aux personnes que j'ai sans doute oubliées (toutes mes excuses !) et merci également au(x) lecteur(s) de cette thèse :-)

[1] http ://www.auberge-aravis.fr/

Table des matières

Table des figures

Introduction

Cette thèse s'inscrit dans le cadre de l'utilisation de la réalité virtuelle pour la formation industrielle. Elle se situe dans le prolongement de la thèse effectuée par Nicolas Mollet [Mol05] et fait également partie du projet GVT (Generic Virtual Training) qui implique Nexter, l'INRIA et l'ENIB. Après avoir exposé les besoins en matière de formation industrielle de notre partenaire industriel : Nexter, nous allons donc présenter le projet GVT, notre problématique, puis décrire l'organisation de ce manuscrit.

La formation industrielle, l'exemple de Nexter

Nexter est une société française spécialisée dans l'équipement militaire comme par exemple le char Leclerc. Ce type de matériel, coûteux et dangereux, nécessite des opérations de maintenance régulières qui requièrent un savoir-faire spécifique. Les besoins de Nexter en matière de formation des personnels sont donc évidents.

Les procédures de maintenance nécessitent de travailler sur des objets complexes, qu'il faut alors modéliser pour pouvoir les intégrer à un environnement virtuel. La figure 1 illustre une centrale de parage réelle et sa modélisation 3D.

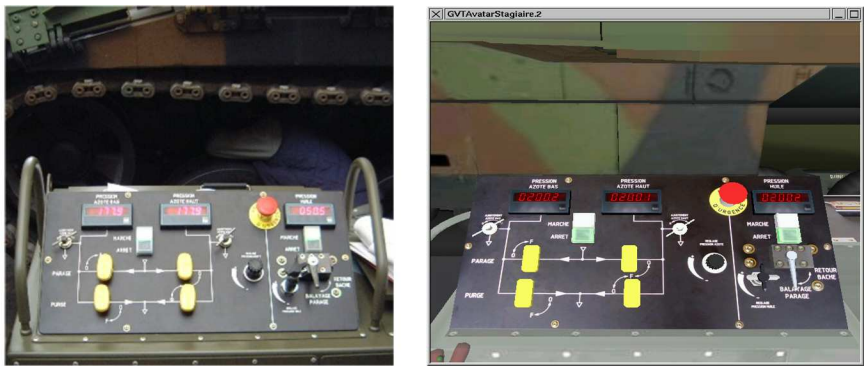


FIG. 1 – Le même équipement (une centrale de parage) en réel et en virtuel

Les procédures de maintenance sont décrites dans des cartes de travail (voir un exemple en annexe A). Pour donner un ordre de grandeur de la quantité de procédures disponibles, Nexter possède une armoire remplie uniquement de classeurs contenant les cartes de travail pour les procédures de maintenance du char Leclerc (figure 2).

FIG. 2 – Les procédures de maintenance sur le char Leclerc

Certaines procédures sont collaboratives et font intervenir plusieurs personnes, comme on peut le voir sur la figure 3. A gauche, trois personnes sont nécessaires pour positionner le char : un guide (au premier plan), une personne qui réplique les ordres du guide et le pilote du char. A droite, deux personnes sont nécessaires pour mettre en place le triangle de remorquage.

FIG. 3 – Extraits d'une procédure de remorquage

L'existant : le projet GVT

Le projet GVT[2] (Generic Virtual Training) a débuté en 2001 et rassemble deux partenaires académiques, l'INRIA et l'ENIB et un partenaire industriel, Nexter Training. Ce projet a abouti en 2005 à la création d'une plateforme de création d'environnements virtuels permettant une formation individuelle à des procédures industrielles de type maintenance.

GVT a été conçu pour une formation à la procédure et non pas aux gestes techniques. Nous considérons que les apprenants sont déjà familiers de l'activité de maintenance (par exemple comment appuyer sur un bouton ou visser une vis) et nous ne leur demandons donc pas de simuler de telles actions élémentaires. A la place, nous utilisons des métaphores visuelles : quand l'utilisateur veut interagir avec un objet, il le sélectionne et un menu contextuel apparaît (voir figure 4). Dans ce menu les icônes représentent les interactions possibles entre cet objet et l'utilisateur.

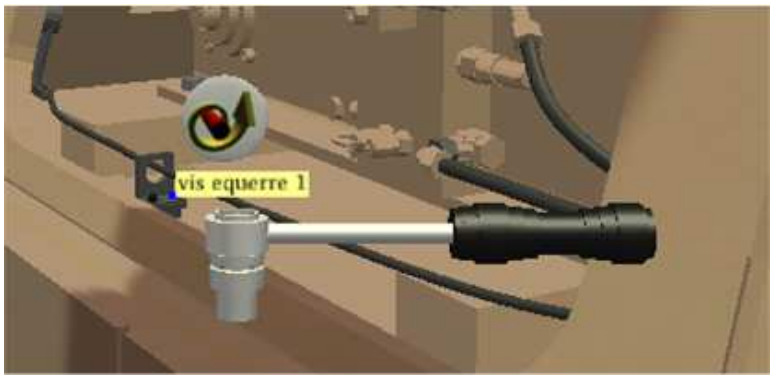


FIG. 4 – Sélection d'une action dans un menu

GVT est désormais une application industrielle opérationnelle. Dans une session de formation typique un formateur supervise plusieurs apprenants, comme l'illustre la figure 5. Chaque apprenant dispose d'un ordinateur et se forme seul sur l'une des procédures disponibles. Différentes configurations matérielles sont disponibles : du portable à l'environnement immersif, en passant par un poste de travail fixe équipé de deux écrans (un pour l'environnement virtuel, un pour l'affichage de l'interface 2D et des documents pédagogiques). Les périphériques utilisés peuvent variés en fonction de la configuration matérielle, mais le logiciel reste le même.

Problématique

L'objectif de cette thèse est de faire évoluer l'outil GVT, dédié à des procédures individuelles, pour proposer une formation à des procédures collaboratives où des utilisateurs réels collaborent entre eux et avec des humains virtuels. Une contrainte industrielle forte consiste à partir des modèles sur lesquels repose GVT pour les modifier. Les humains virtuels doivent avoir un comportement paramétrable, il faut également qu'ils puissent remplacer dynamiquement un utilisateur (ou être remplacés par un utilisateur). Le scénario doit décrire la procédure

[2]http ://www.gvt-nexter.fr/

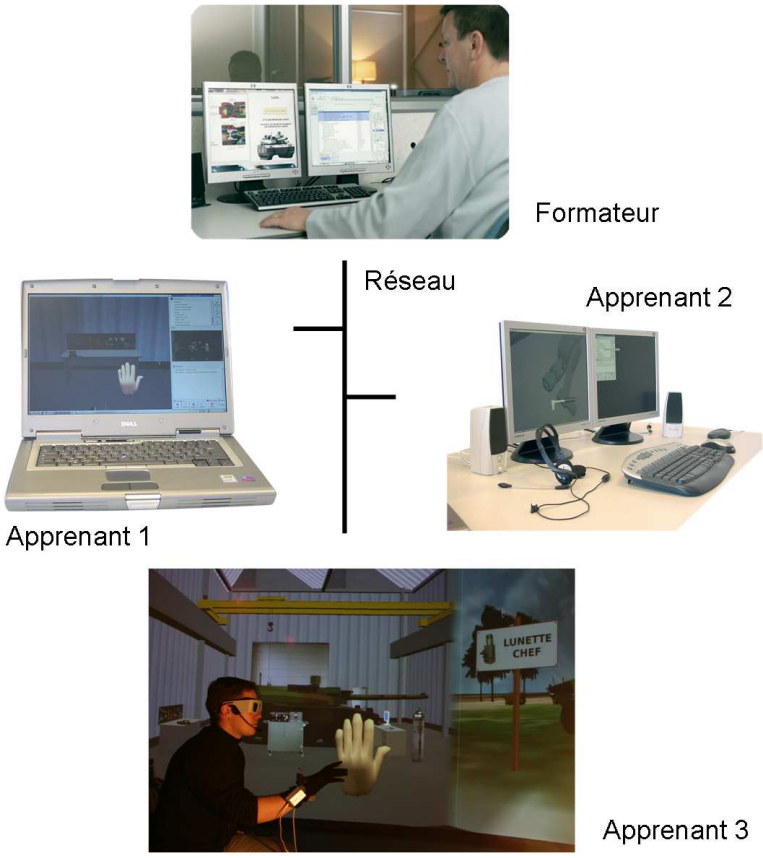


FIG. 5 – Utilisation en réseau de GVT

collaborative, en s'approchant des spécifications réelles des procédures et en permettant de modéliser un éventail le plus large possible des types de procédures collaboratives existantes. Nos contributions vont ainsi se porter sur trois éléments : la modélisation de l'activité d'un acteur, la modélisation du scénario collaboratif ainsi que la mise en place d'un mécanisme de sélection d'action.

Organisation de ce mémoire

Ce manuscrit de thèse est constitué de trois parties : le contexte, nos contributions et l'application qui valide nos modèles.

Le chapitre 1 constitue un état de l'art. Après avoir donné un aperçu des environnements virtuels de formation, nous présentons différents environnements virtuels de formation collaboratifs, puis les analysons. Cette analyse va suivre trois axes : la collaboration, le scénario et les humains virtuels. Nous déduisons de cette analyse les caractéristiques principales pour un environnement virtuel de formation collaboratif ainsi que les limitations des applications existantes. Dans le chapitre 2 nous présentons le contexte dans lequel s'inscrivent nos travaux ainsi que les objectifs que nous nous somme fixés.

La suite du manuscrit constitue nos contributions. Dans le chapitre 3 nous présentons un modèle pour représenter l'activité des acteurs, qu'ils soient réels ou virtuels. Puis, dans le chapitre 4, nous détaillons les extensions que nous avons proposées au langage de scénario LORA afin de pouvoir modéliser des procédures collaboratives. Enfin, dans le chapitre 5, nous expliquons le fonctionnement du mécanisme de sélection d'actions que nous avons mis en place afin de permettre aux humains virtuels de choisir une action à réaliser et d'offrir des conseils pédagogiques aux apprenants réels.

Nous présentons ensuite, dans le chapitre 6, l'intégration dans l'application GVT des modèles présentés. Le prototype ainsi créé permet de faire de la formation à des procédures collaboratives. Un exemple de scénario est présenté pour valider nos modèles.

Enfin nous terminons ce manuscrit par un bilan des travaux effectués puis nous discutons des perspectives envisageables.

Première partie

Contexte et état de l’art

Chapitre 1

Etat de l'art

L'objectif de ce chapitre est de dresser un état de l'art sur les environnements virtuels de formation collaboratifs, afin d'en déduire leurs composants essentiels, leurs points forts et leur manques. Nous allons commencer par décrire, dans la section 1.1, les environnements virtuels de formation sans l'aspect collaboratif. Puis nous présenterons dans la section 1.2 quelques environnements virtuels de formation collaboratifs sur lesquels va s'appuyer notre analyse qui sera détaillée dans la section 1.3. Enfin, dans la section 1.4, nous dresserons un bilan sur les caractéristiques des environnements virtuels de formation collaboratifs actuels.

1.1 Environnements Virtuels pour la Formation

Après avoir abordé, dans la section 1.1.1, quelques-uns des intérêts qu'offre la réalité virtuelle appliquée à un contexte de formation nous présenterons dans la section 1.1.2 les composants requis pour la création d'un environnement virtuel de formation. Nous décrirons enfin, dans la section 1.1.3, l'objectif de cette thèse qui consiste à introduire une dimension collaborative à un environnement virtuel de formation existant.

1.1.1 Réalité virtuelle et formation

Les besoins en formation dans l'industrie sont nombreux et concernent des types de formation variés. Un premier besoin concerne la formation pour acquérir des compétences pratiques, comme par exemple la formation à un geste technique ou à la conduite de véhicule. Le geste est ici au centre de la formation, souvent associé à un matériel spécifique. Un autre type de formation concerne des compétences relationnelles comme par exemple la formation au management, à la prise de décision ou encore à la gestion de personnes. La formation est alors axée autour des interactions entre les personnes. Enfin on trouve la formation à des procédures (maintenance, diagnostic ou utilisation par exemple) où l'accent est mis sur l'apprentissage de la suite d'actions à réaliser. Pour tous ces types de formation, une formation théorique, à base de supports papiers ou de vidéos par exemple, est insuffisante et la pratique est indispensable. Néanmoins, la mise en situation réelle implique parfois de fortes contraintes. Elle impose de réunir le formateur et les apprenants au même endroit, parfois de monopoliser du matériel (en

plusieurs exemplaires si l'on souhaite que les apprenants puissent pratiquer en même temps) ; en cas d'erreur il peut y avoir des risques pour les personnes ou pour le matériel, les situations d'apprentissage ne sont pas toujours simples à reproduire et à contrôler, etc. L'introduction de la réalité virtuelle dans le processus de formation a pour objectif de s'affranchir de ces contraintes. Mais avant de détailler les atouts de la réalité virtuelle en matière de formation, nous allons commencer par définir ce qu'est la réalité virtuelle. Dans le traité de la réalité virtuelle, la définition technique suivante est donnée [AFG06] :

"*La réalité virtuelle est un domaine scientifique et technique exploitant* ***l'informatique*** *et des* ***interfaces comportementales*** *en vue de* ***simuler*** *dans un* ***monde virtuel*** *le* ***comportement d'entités 3D****, qui sont en* ***interaction en temps réel*** *entre elles et avec* ***un ou des utilisateurs*** *en* ***immersion pseudo-naturelle*** *par l'intermédiaire de canaux sensori-moteurs.*"

Nous allons situer chacun de ces termes par rapport à la formation industrielle. L'**informatique** va nous permettre de modéliser des éléments essentiels de la formation. Par exemple pour une formation procédurale il sera possible de décrire la procédure à réaliser puis de suivre son exécution. Les **interfaces comportementales** vont permettre à l'apprenant d'agir sur le système et/ou de recevoir des informations du système. Ces interfaces pourront être adaptées au type de formation visée. Pour la formation aux gestes par exemple le périphérique utilisé est un élément important ; d'ailleurs, dans un souci de réalisme sensoriel, c'est souvent un élément issu de l'environnement réel qui est adapté pour piloter l'environnement virtuel. On peut par exemple citer Cs-Wave [MdMS+05], un environnement de formation au geste de soudage qui utilise une cabine de soudage ou encore certains simulateurs de vols qui utilisent un cockpit d'avion [RMP04]. Les éléments à **simuler** dans un contexte de formation peuvent être de différentes natures : des géométries, de la physique, des fonctions, etc. Il est également important de simuler une situation d'apprentissage précise puisqu'elle peut avoir un impact important sur les choix effectués par l'apprenant. Pour la formation aux gestes, il faut en plus simuler de manière réaliste les effets du geste sur l'environnement. Le fait d'utiliser un **monde virtuel** va permettre de s'affranchir de certaines contraintes qu'impose le monde réel pour se concentrer sur une situation d'apprentissage idéale. La réalité virtuelle permet de simuler le **comportement d'entités 3D**. On peut ainsi modéliser le comportement des objets qui vont composer le monde virtuel dans lequel l'apprenant va devoir réaliser une procédure par exemple. Il est aussi possible de simuler le comportement d'humains virtuels. Ils sont particulièrement importants pour une formation relationnelle puisque la plupart de ces environnements de formation proposent à l'apprenant de se former avec des humains virtuels. Par exemple Franck et al. [FGH+02] proposent un environnement pour former des policiers à gérer des personnes avec des troubles mentaux ; ces dernières sont représentées par des humains virtuels et l'accent est donc mis sur la modélisation de leur comportement et de leurs capacités d'interaction avec l'apprenant, notamment grâce à la communication. Un des éléments les plus importants de cette définition est la notion d'**interaction en temps réel**. Cette notion est primordiale en ce qui concerne le domaine de la formation car elle va permettre à l'apprenant d'apprendre activement : c'est le concept d'*apprendre en faisant* basé sur la théorie constructiviste [Pia76] (l'apprenant agit pour construire ses connaissances). La réalité virtuelle est accessible à **un ou des utilisateurs**, ce qui peut permettre une formation collaborative. Enfin la sensation d'**immersion pseudo-naturelle** peut augmenter les motivations de l'apprenant. Nous reviendrons sur cette notion un peu plus en avant dans cette section.

Les Environnements Virtuels de Formation (EVF) ont pour objectif de faire acquérir ou de renforcer les compétences ou le savoir faire d'apprenants, en vue de leur application dans des situations réelles. Dès lors se pose la problématique du transfert de ce savoir-faire acquis en environnement virtuel vers son utilisation en situation réelle. Cette question constitue un axe de recherche important dans le but de valider ces EVF ; nous y reviendrons un peu plus tard.

Les EVF offrent de multiples atouts par rapport à une formation en situation réelle (ces atouts ont été de nombreuses fois répertoriés, comme par exemple dans [Sto01][BLMd06][CDD+08]) et permettent de pallier aux contraintes évoquées précédemment. Grâce à l'usage de la réalité virtuelle, les risques disparaissent puisque même en cas de mauvaise manipulation, l'apprenant ne pourra pas se blesser ni endommager du matériel pouvant s'avérer très coûteux. Certains EVF permettent également de faire de la formation à distance : le formateur et les apprenants n'ont plus besoin d'être regroupés géographiquement, ce qui permet d'apporter plus de flexibilité dans la formation et de réduire les coûts. Le matériel n'a plus besoin d'être disponible, ainsi il est aisé de se former sur un matériel rare voire même inexistant (en cours de conception par exemple). Un autre avantage concerne la pédagogie : en environnement virtuel il est possible de reproduire une situation rare (par exemple le dysfonctionnement d'un composant), de reproduire plusieurs fois exactement la même situation ou de faire varier uniquement des paramètres précis impossibles à contrôler dans le monde réel (exemple : la pression dans une bonbonne de gaz). Les EVF facilitent aussi la surveillance des apprenants : les traces de l'activité de ces derniers peuvent être enregistrées et ré-exploitées ensuite, des mesures de performance peuvent ainsi être effectuées automatiquement (nombre d'erreurs, temps passé à réaliser la tâche, etc). De plus, les EVF offrent de nouvelles possibilités pédagogiques, comme visualiser des phénomènes qui ne sont pas visibles dans le monde réel (par exemple la température d'une pièce mécanique) ou encore ajouter des informations à l'environnement ou au contraire simplifier cet environnement afin de guider l'apprenant. Les EVF présentent donc de nombreux avantages. Ces avantages se traduisent par des retours sur investissement et c'est pourquoi le domaine de la formation est un des domaines d'application émergent pour la réalité virtuelle.

Parmi les EVF existants, certains exploitent les périphériques offerts par la réalité virtuelle afin de mieux immerger l'apprenant dans le monde virtuel. Le concept d'immersion a été défini par Pimentel et Teixeira [PT94] comme l'état (perceptif, mental, émotionnel) d'un sujet lorsque un ou plusieurs sens sont isolés du monde extérieur et sont alimentés uniquement par des informations issues de l'ordinateur. Les EVF immersifs permettent un sentiment de réalisme plus fort, ils permettent aussi des interactions sensori-motrices plus riches. Les EVF non immersifs, eux, sont plus simples à utiliser, moins coûteux, plus facilement diffusables [WL04]. Les deux systèmes ont ainsi leurs avantages et leur inconvénients, et le fait d'aller vers l'un ou vers l'autre peut dépendre du type de formation visé. La pertinence d'utiliser des EVF immersifs fait néanmoins l'objet de débats et il n'a pas été prouvé que l'immersion améliore les performances en matière d'apprentissage. Les dernières tendances concernant la création d'EVF sont de ne pas privilégier le réalisme à tout prix pour se focaliser davantage sur les fonctionalités pédagogiques et également de simplifier ces EVF pour qu'ils fonctionnent sur de simples ordinateurs de bureau afin de les rendre accessibles au plus grand nombre. Cependant, beaucoup des EVF développés restent à l'état de prototype et peu sont réellement utilisés dans le cadre d'une réelle formation industrielle. C'est pourquoi peu d'études existent concernant le transfert des connaissances apprises dans ces EVF vers le monde réel. Néanmoins, les premiers résultats

semblent prometteurs : la motivation des apprenants face à un EVF peut rendre la formation plus rapide par exemple. Une étude est intéressante à citer, il s'agit de l'évaluation menée sur le robinet virtuel d'EDF (citée dans [Md04]). Cet environnement vise à former des techniciens de maintenance en centrale nucléaire à une démarche de diagnostic de panne et est réellement utilisé pour faire de la formation. Il a été évalué dans le cadre d'une comparaison avec la situation précédente de formation (formation dans le monde réel), ce qui a permis de montrer une diminution d'environ 30% du temps requis pour la formation ainsi qu'une amélioration des compétences des apprenants pour la préparation de l'intervention de diagnostic à l'issue de la formation.

Parmi les différents types de formation abordés dans notre mémoire, nous nous intéressons à la formation à la procédure, et plus particulièrement à la procédure de maintenance. La suite de notre état de l'art se focalisera donc sur les EVF dans lesquels il y a une procédure à suivre (que cette procédure soit directement l'objet de la formation ou non).

1.1.2 Besoins pour un EVF

Dans un environnement virtuel de formation, on retrouve trois éléments constitutifs importants : l'environnement virtuel composé d'objets qui proposent des interactions, le scénario qui permet de traduire une procédure de référence et des composants pédagogiques. Ces différents éléments et la façon de les modéliser sont détaillés dans [Mol05], nous ne faisons ici que les rappeler brièvement.

– Le premier point important est de décrire la géométrie des objets qui composent l'environnement virtuel, de leur donner un comportement et de décrire les interactions possibles entre eux. Concernant la géométrie, il est possible de récupérer des modèles provenant de la CAO puis de les simplifier ou alors de les modéliser directement à partir d'un modeleur 3D. Pour décrire le comportement des objets, trois approches principales existent : les systèmes stimulus-réponse, les systèmes à base de règles et les systèmes à base d'automates. Les approches basées sur les systèmes stimulus-réponse proposent d'utiliser des réseaux composés de nœuds entre lesquels des informations vont pouvoir transiter. On peut par exemple citer les travaux de Van De Panne [vdP93] qui utilise des réseaux SAN (Sensor Actuator Networks) pour le contrôle du déplacement d'entités. De tels systèmes sont intéressants pour obtenir des comportements réactifs, adaptables et limités mais ils proposent un niveau d'abstraction très bas et ne permettent pas de contrôler finement le comportement obtenu. Les systèmes à base de règles offrent un plus haut niveau d'abstraction. Ce principe a été utilisé notamment par Reynolds [Rey87] et par Tu et Terzopoulos [TT94]. Par contre lorsque que le nombre de règles augmente les déductions peuvent prendre du temps ce qui est un inconvénient majeur pour des applications temps réel. Les systèmes à base d'automates peuvent paraître plus difficile à utiliser car ils ne permettent pas une description déclarative du comportement tandis que les systèmes à bases de règles le permettent. En revanche l'utilisation de systèmes d'automates avancés permet un contrôle fin et favorise la réutilisation d'éléments de comportement. Dans ce domaine, les automates parallèles hiérarchiques sont des bons candidats comme par exemple HPTS (Hierarchical Parallel Transitions System) [DA94] ou HCSM (Hierarchical Concurrent State Machines) [CKP95]. Pour décrire des interactions complexes,

il est nécessaire de fournir un haut niveau d'abstraction. Cette problématique se retrouve principalement dans les travaux portant sur des humais virtuels. Il est intéressant de citer ici deux approches très différentes. La première approche repose sur des techniques d'intelligence artificielle. Toutes les informations sur les interactions possibles sont centralisées, par exemple dans un système expert qui va gérer les interactions possibles entre un humain virtuel et les objets de l'environnement [HEHV03]. La deuxième approche consiste à créer un environnement informé. Les informations sur les interactions sont alors localisées dans chaque objet. Les smart-objects [KT98, Kal01] de Kallman, par exemple, contiennent toutes les informations sur les interactions qu'ils offrent ainsi que sur la manière dont un humain virtuel va pouvoir les utiliser.

– Le deuxième élément important est la description du scénario. En effet, puisque nous nous intéressons aux environnements de formation permettant d'apprendre une procédure, il faut donc formaliser cette procédure au sein de l'environnement virtuel et mettre en place un mécanisme capable de dérouler le scénario au fur et à mesure que l'apprenant réalise des actions. Certains outils permettant de modéliser des comportements complexes peuvent aussi être utilisés pour décrire des scénarios (par exemple le modèle HCSM [CKP95]). Mais ce mode de description reste trop bas niveau et ne convient donc pas pour décrire une procédure pour un EVF. Les langages de scénario sont alors mieux adaptés. On peut citer notamment trois grandes familles, les fictions interactives, la description complète des séquences d'actions et une description grâce à des contraintes et à des buts. Les fictions interactives (exemple [PHM+03]) sont plus adaptées pour représenter un scénario avec plusieurs déroulements possibles plutôt que pour représenter une procédure. Les langages permettant une description complète de la succession des actions comme le langage G [Ish02] ou le Grafcet [LSF99] offrent des propriétés intéressantes comme la paramétrisation, la hiérarchisation, la notion de parallélisme, etc. De plus le Grafcet offre une représentation graphique, ce qui le rend plus facilement accessible à des non informaticiens. En effet, le scénario d'un EVF devrait pouvoir être décrit par une personne ayant peu ou pas de connaissance en informatique : la personne la plus à même de décrire la procédure est le formateur ou un expert de la procédure et on gagnerait en productivité en évitant le transfert de connaissance entre cet expert et un informaticien. La représentation grâce à des contraintes et des buts (exemples [JR97][QC01][BEL02]) est une représentation déclarative, plus flexible que la description complète. Elle permet par exemple l'adaptation du scénario à des événements non prévus. Mais cette flexibilité n'est pas toujours souhaitable lorsque l'on veut imposer à l'apprenant de respecter strictement une procédure de référence. De plus, la causalité n'est pas toujours explicite dans les scénarios industriels, ce qui impose à l'auteur un travail supplémentaire de transcription.
– Le dernier élément nécessaire pour un EVF est un cœur pédagogique : il faut par exemple fournir au formateur un ensemble d'outils pour qu'il puisse superviser des apprenants. L'outil peut également proposer des actions pédagogiques que le formateur pourra déclencher. Domitile Lourdeaux [Lou01] parle d'interventions pédagogiques qu'elle définit comme des interactions formateur-formé via l'outil. Elle classe ces interventions pédagogiques en plusieurs types :
  – **Enrichissement** : ajout de symboles visuels, sonores ou de films d'animation.

- **Dégradation** : détérioration du réalisme (repères effacés, feed-back proprioceptifs dégradés, couleurs atténuées, flous à l'arrière-plan et/ou sur les cotés, réduction d'objets, iconisation, etc.).
- **Rehaussement** : exagération de la réalité (représentation d'objets à plus grande échelle, objets surréalistes, plus lumineux, plus brillants, etc.).
- **Simplification** : allégement de la scène virtuelle (une foule peut être représentée par des personnes aux mouvements simplifiés, objets simplifiés, systèmes kinesthésiques simplifiés, représentation en fil de fer, etc.), représentation schématique de certains appareils.
- **Restriction** : limitation de certains déplacements ou manipulations (limitations du périmètre dans lequel l'utilisateur peut se déplacer, etc.).
- **Animation** : séquence animée (positionnement automatique, clé qui tourne automatiquement une fois mise en place, etc.).
- **Décentrage** : changement du point de vue habituellement attaché à l'œil du formé immergé (vue de derrière, au-dessus, etc.).
- **Modification** : modification d'aspect, de texture (changement de couleurs, clignotement d'objets, etc.).
- **Modélisation** : Représentation de concepts abstraits, de phénomènes physiques invisibles à l'œil nu, de type de pannes, etc.
- **Visualisation** de mécanismes cachés (intérieur d'un moteur, engrenages, etc.).

Dans un EVF, il est également intéressant de proposer des aides automatiques qui pourront être données à l'apprenant. L'ajout d'ITS (Intelligent Tutoring System) à des EVF est une des solutions possibles [Buc05]. D'autres EVF proposent plutôt un agent pédagogique qui est un personnage virtuel ayant pour but de remplacer le formateur (exemple : Steve [JR97, RJ99a]).

Nous venons de décrire les composants principaux pour un EVF. D'autres éléments peuvent néanmoins être requis en fonction du type de formation proposé par l'EVF.

1.1.3 Introduction de la collaboration

L'objectif de cette thèse est d'ajouter des fonctionnalités à un EVF permettant initialement de former à des procédures individuelles, afin de proposer des procédures collaboratives où les apprenants pourront collaborer entre eux et avec des humains virtuels. La réalité virtuelle ajoute là encore de nouveaux atouts en ce qui concerne la collaboration : la possibilité de collaborer avec des personnes distantes, la possibilité de s'affranchir de la disponibilité des coéquipiers en proposant des coéquipiers virtuels et le contrôle du comportement des humains virtuels afin de mieux maîtriser la situation pédagogique.

Notre objectif est de partir d'un EVF existant : les composants principaux d'un EVF rappelés dans la section précédente sont donc déjà présents et doivent nous servir de base. Il faut par contre modifier certains de ces composants et en ajouter de nouveaux. Concernant l'environnement et le comportement des objets qui le constituent, il faut qu'il puisse prendre en compte plusieurs utilisateurs et doit donc permettre de modéliser un acteur, qu'il soit réel ou virtuel. Concernant le scénario, il doit permettre de décrire un scénario multi-utilisateur ; il va donc falloir décrire l'assignation des personnes aux actions. La pédagogie va elle aussi être modi-

fiée puisqu'on va pouvoir créer des humains virtuels avec lesquels les apprenants vont pouvoir s'entraîner. On pourra ainsi fournir aux apprenants un partenaire avec un comportement adapté. Dans la suite de cet état de l'art, nous allons donc étudier différents environnements virtuels de formation collaboratifs existants et voir quels sont les modèles qu'ils utilisent et leurs fonctions.

1.2 Description des EVFC étudiés

Parmi les environnements virtuels de formation, certains proposent à un apprenant de se former non pas seul mais en compagnie d'autres utilisateurs ou d'humains virtuels. Une forme de collaboration apparaît alors, ce qui leur vaut le qualificatif d'Environnements Virtuels de Formation Collaboratifs (EVFC). Nous allons étudier différents EVFC ayant trait à une procédure (à des séquences d'opérations), que l'apprentissage porte sur la procédure en elle même (exemple : une procédure de maintenance) ou que l'apprentissage porte sur la gestion d'une situation particulière, en s'appuyant sur une ou plusieurs procédures standards. Nous avons sélectionné cinq projets à la fois parce qu'ils nous semblaient représentatifs de la diversité qui existe au sein des applications de formation collaborative en RV, mais aussi car ils permettent d'identifier les éléments qui sont communs à tous les EVFC existants. Avant d'analyser leurs atouts et leurs manques (dans la section 1.3), nous allons commencer par présenter ces projets dans leur ensemble, en expliquant le contexte dans lequel ils ont été développés et leur finalité.

1.2.1 COVET

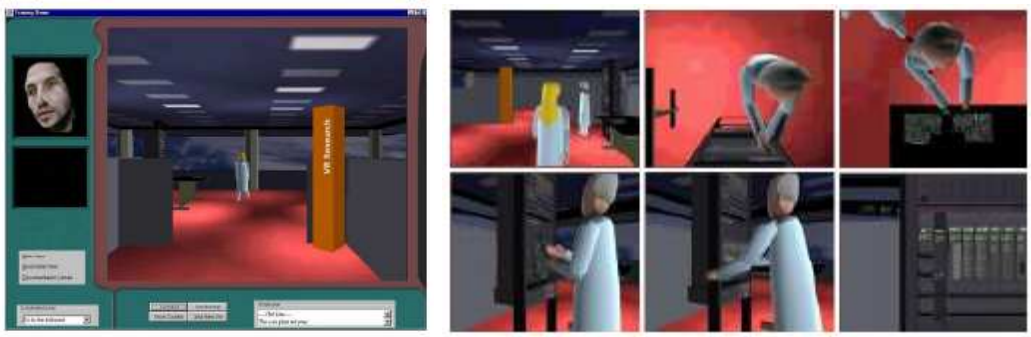


FIG. 1.1 – Illustrations de COVET

COVET [OHS+00, OSG00a, OSG00b, HG01] (COllaborative Virtual Environment for Training) est un prototype mis au point pour NewBridge Networks (maintenant Alcatel) permettant à des utilisateurs d'effectuer un exercice de formation simple. Afin d'apprendre à remplacer une carte défectueuse sur un commutateur ATM, plusieurs apprenants et un formateur se partagent l'environnement virtuel. Cependant, seulement l'un d'eux peut interagir avec les objets qui s'y trouvent et les autres participants sont seulement spectateurs : ils se contentent de regarder, de discuter entre eux de la procédure en train d'être accomplie ; ils peuvent également se déplacer ou changer leurs points de vue pour mieux voir ce qui se passe. Dans un premier

temps, c'est donc le formateur qui va interagir pour montrer aux apprenants la procédure à réaliser. Ensuite, un apprenant pourra à son tour réaliser la procédure. Pour sélectionner les actions que l'utilisateur souhaite réaliser dans l'environnement, deux moyens sont possibles : il peut naviguer dans la scène et sélectionner à la souris l'objet avec lequel il veut interagir, ou alors il peut passer par un menu.

Il est intéressant de noter que le projet COVET a ensuite évolué pour proposer un environnement de formation qui utilise désormais la réalité augmentée, toujours dans le but de faire de la formation sur des équipements ATM [ZLGB03].

1.2.2 EVFC de Dugdale

FIG. 1.2 – L'EVFC de Dugdale

Un autre EVFC intéressant est présenté dans l'article [DPPeJ04]. Il s'agit ici de former le chef d'une équipe de pompiers à bien superviser une opération de secours. Plusieurs autres participants sont nécessaires dans ce scénario et les concepteurs ont choisi de les faire tous jouer par des utilisateurs réels, afin d'économiser la ressource informatique nécessaire pour créer une forme d'intelligence virtuelle et pour simplifier les modèles. Les personnages virtuels que contrôlent les utilisateurs sont tout de même dotés d'une certaine autonomie concernant la réalisation d'actions non intentionnelles (par exemple hocher la tête dans une conversation) et sont aussi capables de s'adapter au contexte (par exemple s'éloigner plus rapidement lors d'une explosion). Le comportement du personnage virtuel dépend de sa perception de la situation, de son état émotionnel, de sa personnalité et de son humeur.

1.2.3 MRE

MRE [RMG+02, TRGM03, JGM+03, SGH+05, SGH+06] (Mission Rehearsal Exercise) est un EVFC qui permet à des chefs de petites unités de l'armée américaine d'acquérir des compétences concernant la prise de décision en situation critique. Dans cette application, il y a un seul formé qui doit collaborer avec des humains virtuels capables de raisonner, de tenir une

conversation et de montrer des émotions. Ici l'apprenant ne doit pas apprendre une procédure stricte ; la trame de l'action est scénarisée à la manière d'un film, le but est de mettre l'apprenant dans une situation stressante et de lui apprendre à gérer son stress pour réagir néanmoins de manière adéquate. Cet EVFC a été créé à partir des travaux sur Steve [JR97, RJ99a] qui ont été modifiés pour s'axer davantage sur les aspects dialogue naturel et émotions.

FIG. 1.3 – L'application MRE

1.2.4 SECUREVI

SECUREVI [QRC01, QBMC03, QBMC04] (SECUrité et REalité VIrtuelle) est un environnement virtuel pour entraîner des officiers sapeurs-pompiers à la gestion opérationnelle et au commandement lors d'interventions sur des sites à risques classés SEVESO. L'application permet une formation à un travail procédural et collaboratif. Elle a pour but d'aider les apprenants à prendre des décisions dans des situations opérationnelles et non pas de leur enseigner des gestes techniques. Cet EVFC repose sur le modèle MASCARET [BQLC03]. MASCARET utilise des systèmes multi-agents pour simuler des environnements réalistes, collaboratifs et adaptatifs pour l'entraînement. Il s'agit d'un modèle permettant de structurer les interactions entre les agents et de fournir aux agents des capacités réactives, cognitives et sociales. SECUREVI est une application développée pour la sécurité civile du Finistère. Les sites d'intervention, les phénomènes physiques ainsi que les intervenants humains sont simulés. Les apprenants participent à la simulation, via leurs avatars, en jouant le rôle qui leur est affecté. Leur objectif est de réaliser les actions qui sont de leur ressort tout en adaptant la procédure en fonction de la situation. Le formateur est lui aussi immergé dans l'environnement qu'il peut modifier en temps réel pour créer des incidents par exemple. L'apprenant joue généralement le rôle d'un chef de groupe et il doit alors recueillir des informations nécessaires pour choisir un plan à exécuter, donner des ordres aux pompiers qu'il dirige et effectuer un rapport à son supérieur hiérarchique (joué par le formateur). Les pompiers sont des agents autonomes et le formateur peut décider de prendre le contrôle de l'un d'eux s'il le désire.

Dans SECUREVI, trois types d'agents différents sont implémentés : les agents réactifs (éléments physiques du système, phénomènes naturels), les agents cognitifs (personnels de la sécurité civile) et les avatars (représentation des utilisateurs dans le monde virtuel). Les agents cognitifs et les avatars doivent coordonner leurs actions afin de réaliser en équipe des missions préétablies. Ces interactions sont effectuées grâce à des échanges de messages. Les missions sont décrites dans des procédures qui servent comme référence et qui sont considérées comme

parfaitement connues par tous les membres de l'équipe. Les agents cognitifs vont toujours réaliser les actions de la procédure qu'ils sont censés faire en fonction de leur rôle et ne feront donc jamais d'erreur.

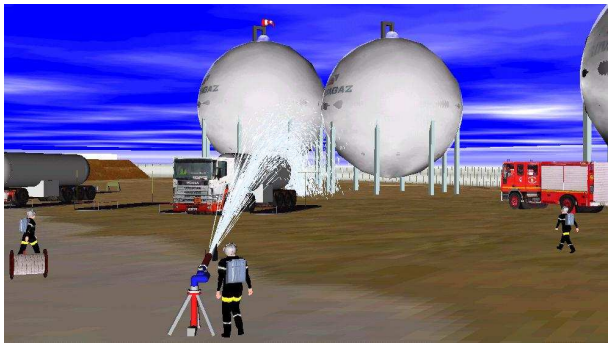


FIG. 1.4 – L'application SECUREVI

1.2.5 Version collaborative de Steve

Le dernier EVFC que nous allons présenter est celui qui se rapproche le plus de ce que nous voulons faire puisqu'il permet de faire de la formation à des procédures de maintenance collaboratives. Les agents Steve (Soar Training Expert for Virtual Environments) [JR97, RJ99a] ont été étendus pour gérer le travail en équipe [RJ99b, RJ00, RJ03]. Steve était au départ un agent virtuel animé capable de fournir à l'apprenant des aides pédagogiques automatisées. Désormais les agents Steve peuvent jouer deux rôles pédagogiques distincts : instructeur ou coéquipier. Dans cette application des humains réels (apprenants, formateurs) et des humains virtuels (des agents Steve) doivent réaliser une procédure collaborative, avec plusieurs rôles déterminés, dans le domaine de la maintenance navale. La procédure consiste en un certain nombre de buts à atteindre, eux même décomposés en sous-buts jusqu'à atteindre des actions primitives qui correspondent à des actions dans l'environnement (par exemple presser un bouton) devant être réalisées par un rôle donné. Chaque rôle peut être joué indistinctement par un utilisateur réel ou un humain virtuel. Chaque apprenant est assisté par un agent Steve ayant un rôle pédagogique d'instructeur et capable de répondre à ses questions et de réaliser des actions à sa place. L'apprenant et son instructeur joue donc le même rôle dans l'équipe afin de pouvoir réaliser l'un comme l'autre les tâches correspondant à ce rôle.

Rappelons que les agents Steve ont également servi de base pour modéliser les humains virtuels dans l'application MRE (présentée dans la section 1.2.3). Mais alors que dans MRE les fonctionnalités de communication des agents ont été étendues afin de transmettre les émotions, ici l'accent a été mis sur la description des tâches collaboratives et leur interprétation par les humains virtuels.

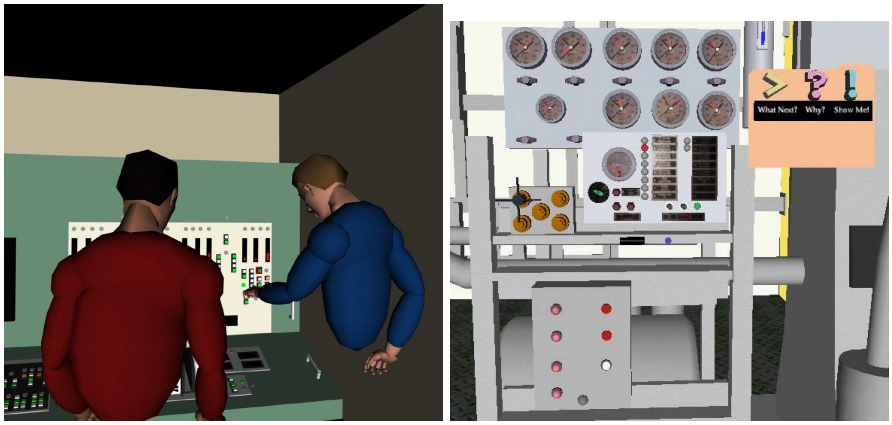


FIG. 1.5 – La version de Steve pour gérer le travail en équipe

1.2.6 Autres EVFC existants

D'autres projets ayant pour but la création d'EVFC existent et ont été présentés dans des publications ; nous pouvons en citer quelques uns ici. Nous n'allons pas y faire référence de manière systématique dans notre analyse des EVFC car ils nous semblaient redondants par rapport aux EVFC sélectionnés ou trop éloignés de nos objectifs. Néanmoins, ils comportent certains points d'intérêt particuliers et nous y ferons donc référence ponctuellement dans la suite de cet état de l'art.

- Le robinet virtuel industriel d'EDF (présenté dans [Md04]) a pour but de former les techniciens de maintenance en centrale nucléaire à une démarche de diagnostic de panne. Le système permet de visualiser interactivement un robinet industriel sur un écran. Cet environnement est piloté par le formateur devant un groupe d'apprenants, il repose donc sur un apprentissage collectif. L'environnement lui-même n'est pas multi-utilisateur puisque le formateur et les apprenants sont réunis dans une même salle et seul le formateur le contrôle. Cette approche de formation collective ressemble à celle de COVET.
- L'application GASPAR (Gestion Aviation Sur Porte-Avions par la Réalité virtuelle) [Buc05, Mar06, MSBQ07] permet de visualiser les activités aériennes sur un porte-avions. Ce projet a pour but de simuler différentes configurations pour un nouveau porte-avions que la marine française envisage de construire et de tester leur viabilité via la réalisation de diverses procédures (ex : procédure de catapultage). Même si, dans un cadre de validation, les agents sont tous joués par des humains virtuels autonomes, un utilisateur peut prendre la main sur un agent et réaliser les actions à sa place. Il est envisagé que l'application soit utilisée par la suite pour former le personnel à ces procédures. Cet EVFC repose sur le modèle MASCARET, tout comme SECUREVI, ce qui leur confère beaucoup de caractéristiques communes.
- Beaucoup d'applications militaires existent et utilisent la réalité virtuelle pour former des soldats dans diverses situations collaboratives (ex : commandement, gestion de crise). Dans l'article [LSMC04] par exemple, les auteurs présentent une application pour for-

mer un garde militaire devant garder un check-point. Cette application a beaucoup de points communs avec MRE : les autres participants sont joués par des humains virtuels, les interactions ont surtout lieu entre agents et il y a peu d'interaction entre l'apprenant et des objets de l'environnement virtuel. Un autre exemple d'application militaire que nous pouvons citer est Twilight City [ZT06, TZ07], une plateforme pour la simulation militaire sur terrain urbain où formés et humains virtuels peuvent être mélangés au sein d'une même équipe et doivent collaborer pour accomplir leur mission. Ce type d'EVFC est assez proche des jeux vidéos militaires. Il est difficile de parler de procédure à accomplir, car généralement l'objectif est d'atteindre un but donné et le moyen d'y arriver n'a souvent que peu d'importance.

1.3 Analyse d'EVFC : leurs points forts et leurs manques

Nous allons maintenant analyser ces différents EVFC suivant 3 grands axes : les formes de collaboration proposées, la description du scénario collaboratif et les caractéristiques des humains virtuels autonomes. Pour chacun de ces axes nous commencerons par faire un point théorique sur ce qui se fait dans un contexte plus général avant de nous restreindre à l'analyse spécifique des EVFC.

1.3.1 Différentes formes de collaboration

Par rapport aux EVF individuels, les EVFC intègrent en plus la notion de collaboration. Certains auteurs font une différence entre les termes collaboration et coopération, mais au sein de la communauté RV les nuances pouvant exister entre ces deux termes font encore l'objet de débats et leurs définitions ne sont pas figées. Nous avons donc décidé, dans un souci de cohérence, d'utiliser uniquement le terme de collaboration même si nous ne faisons pas de distinction sémantique avec le terme de coopération. Dans le projet ANR/RNTL Part@ge la collaboration est considérée comme "*la possibilité de contribuer à une tâche globale grâce à des actions complémentaires de plusieurs utilisateurs. Ces actions peuvent se faire de différentes façons, en alternance, ou en simultané, sur un même objet ou sur des objets différents*". Cette définition est très orientée action et nous considérons pour notre part la collaboration dans un cadre plus vaste. La collaboration sera pour nous le fait que plusieurs acteurs, réels ou virtuels, agissent avec un objectif commun ; cet objectif peut être une tâche concrête à accomplir (les actions d'une procédure par exemple) ou une tâche plus abstraite comme le fait d'apprendre ou de comprendre quelque chose. La collaboration pourra se traduire par des actions de communication, une coordination des actions ou des actions pouvant être simultanées sur un même objet.

Dans les EVFC la collaboration peut avoir des finalités variées et prendre des formes bien différentes. Nous allons donc donner dans un premier temps différentes problématiques liées à la collaboration au sein d'un environnement virtuel (et les différentes typologie de collaboration qui en découlent le cas échéant), puis nous décrirons quel peut être l'usage de la collaboration lorsqu'elle est utilisée dans un contexte de formation.

1.3.1.1 Prendre conscience des autres et de leur activité

Dans un environnement virtuel multi-utilisateur, plusieurs personnes se côtoient et partagent le même monde virtuel, il faut alors que chacun ait conscience de la présence des autres et de leur activité. Cette nécessité est d'autant plus importante dans un environnement virtuel collaboratif où les différents utilisateurs et/ou humains virtuels doivent collaborer. Cette notion de conscience[1] est donc un besoin essentiel dans un environnement virtuel collaboratif.

La solution la plus largement répandue pour incarner les utilisateurs est de les représenter dans le monde virtuel au travers d'un avatar. Cet avatar, qui est en fait une représentation 3D de l'utilisateur, peut avoir divers aspects, du plus symbolique ou plus réaliste (voir figure 1.6). Benford et al. [BBF+95] ont identifié les problèmes que permet de résoudre l'utilisation d'un avatar au sein d'un environnement virtuel collaboratif, les plus fondamentaux étant la présence, la localisation et l'identité. L'avatar est désormais adopté dans les environnements virtuels car il apporte de nombreuses informations, de manière intuitive ; d'ailleurs Benford [BBF+95] identifie très bien le problème qui se poserait si on omettait d'utiliser un avatar : "*without sufficient embodiment, users only become known to one another through their (disembodied) actions ; one might draw an analogy between such users and poltergeists, only visible through paranormal activity*".

FIG. 1.6 – Différents avatars. A gauche : une main, à droite : un avatar anthropomorphe

En revanche concernant la représentation de l'activité, les recherches sont toujours nombreuses afin de proposer les moyens les plus intuitifs ou porteurs du plus de sens possible pour représenter l'activité de chacun. Certaines métaphores permettent d'avoir une information en permanence sur les autres utilisateurs, pour cela Fraser et al. [FBHH99, FGV+00] proposent par exemple d'utiliser une pyramide de vision en fil de fer ou transparente pour représenter le champ de vision d'un utilisateur (voir figure 1.7). Il est possible d'apporter des informations plus ponctuelles, par exemple sur l'objet sélectionné par un utilisateur. Pour indiquer cette sélection il est possible par exemple de changer la texture ou la couleur de cet objet. En revanche avec cette méthode il peut s'avérer difficile de savoir quel utilisateur interagit avec un objet donné. Fraser et al. proposent également des techniques pour remédier à cela, comme par exemple d'étendre le bras d'un utilisateur jusqu'à le faire toucher l'objet qu'il va vouloir at-

[1] le terme anglais correspondant ici est *awareness*

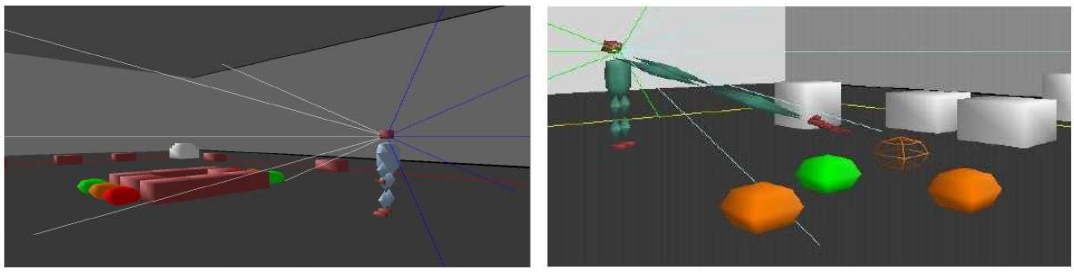


FIG. 1.7 – Métaphores. A gauche : représentation du champ de vision. A droite : manipulation d'un objet. D'après [FBHH99]

traper ou déplacer (voir figure 1.7). D'autres recherches concernent davantage les interactions collaboratives : comment faire comprendre aux utilisateurs qu'ils ne sont pas seul à agir sur un objet et que les modifications de l'objet ne sont donc pas le fruit de leur seule interaction mais bien d'une collaboration. Duval et al dans [DF02] proposent plusieurs types de rayons virtuels pour représenter la manipulation collaborative sur un objet. Par exemple le rayon coudé rattache l'utilisateur à l'objet manipulé mais il se courbe en fonction des forces appliquées par les autres utilisateurs (voir figure 1.8).

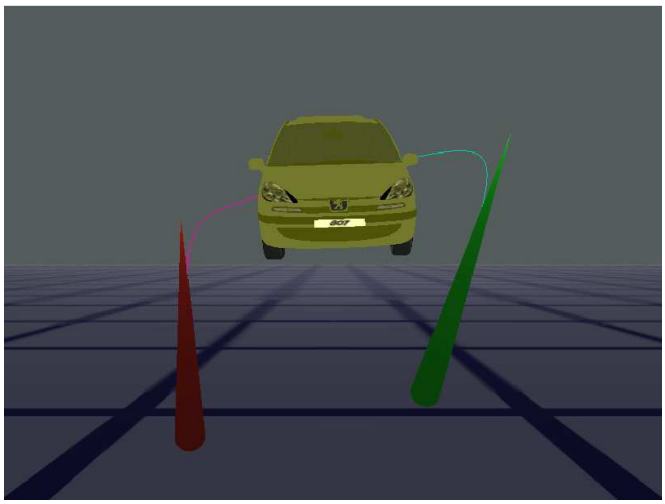


FIG. 1.8 – Rayons courbés pour représenter les contraintes

Beaucoup de métaphores ont été proposées pour représenter l'activité des utilisateurs et nous n'avons fait qu'en citer quelques unes afin d'illustrer certains des problèmes à résoudre. De plus, au delà de la représentation de l'activité des différents utilisateurs, il faut aussi pouvoir représenter de manière compréhensible par les utilisateurs l'activité des humains virtuels.

1.3.1.2 Complexité de la collaboration au sein d'un environnement virtuel

Dans un environnement virtuel, la collaboration peut prendre des formes variées qui induisent des niveaux de complexité différents du point de vue de la gestion de la scène. Margery [MAP99] a défini trois niveaux de collaboration pouvant apparaître dans un environnement virtuel collaboratif que nous allons détailler tout en nous intéressant aux difficultés techniques qu'ils impliquent.

- Le niveau 1 est le niveau basique de la collaboration. C'est à ce niveau qu'intervient la conscience des autres que nous avons décrit de manière plus détaillée dans la section précédente. Ici les utilisateurs se perçoivent les uns les autres dans le monde virtuel grâce aux avatars et disposent de moyens pour communiquer entre eux. Cette collaboration est commune à tous les systèmes multi-utilisateur. Néanmoins, les contraintes sur le système ne sont pas les mêmes entre une communication textuelle, par gestes ou audio. Pourtant, tous ces systèmes partagent un modèle simple de cohérence de la scène.
  Le niveau 1 de la collaboration suppose une distribution de l'application afin que celle-ci soit disponible sur plusieurs machines distantes. Elle implique également la mise en place de moyens de communication (exemple : transmission du son) qui doivent être transmis avec le moins de latence possible.
- Du point de vue de la gestion de scène, l'étape suivante est d'autoriser l'utilisateur à agir dans la scène. Quand chaque utilisateur peut changer la scène individuellement, on dit que le système supporte une collaboration de niveau 2. Ce niveau peut être divisé en 2 sous niveaux :
  - Au niveau 2.1 toutes les modifications possibles de l'environnement sont prédéfinies dans la conception de la scène. Par exemple les utilisateurs peuvent lancer des animations ou changer l'état de certains objets de la scène (ex : ouvrir un tiroir), la combinatoire des interactions est prévue à l'avance.
  - Au niveau 2.2 les modifications faites par un utilisateur unique ne sont pas contraintes. L'exemple le plus évident est la possibilité de déplacer un objet n'importe où dans la scène. La principale limitation de la collaboration de niveau 2 est que les utilisateurs ne peuvent pas collaborer sur les même objets de la scène. Par exemple si deux utilisateurs veulent déplacer simultanément le même objet, un mécanisme de verrou va assurer qu'un seul utilisateur puisse effectuer l'action. Une autre limitation est le problème de l'interaction fortuite : par exemple si deux utilisateurs déplacent chacun un objet différent, il se peut que ces objets entrent en collision, dans ce cas on devrait passer au niveau supérieur de collaboration. Si on veut se cantonner à de la collaboration de niveau 2, on va aboutir à un état incohérent (par exemple interpénétration des deux objets).

  Le niveau 2 implique la mise en place de verrous : lorsqu'un utilisateur manipule un objet, celui-ci est verrouillé et aucun autre utilisateur ne peut le manipuler tant que le verrou n'a pas été relâché. Il se pose alors le problème de comment informer le deuxième utilisateur qu'il ne peut pas agir sur l'objet pour le moment. Par ailleurs, il faut déterminer la stratégie à adopter dans ces cas là. Il est possible de mettre l'utilisateur en attente (il sera donc bloqué) et dès que le verrou sera relâché il l'acquerra automatiquement. Il est aussi possible de l'empêcher d'agir sur l'objet tant qu'il est verrouillé mais sans le bloquer ;

l'utilisateur pourra alors faire autre chose en attendant et réessayer ensuite de manipuler l'objet.

- La collaboration de niveau 3 lève les limitations du niveau 2 et autorise donc plusieurs utilisateurs à manipuler de manière simultanée le même objet. Ce niveau de collaboration est divisé en 2 sous-niveaux, en fonction du type de collaboration possible sur l'objet.
  - Au niveau 3.1 deux utilisateurs ou plus peuvent agir ensemble sur un même objet mais de manière indépendante : par exemple un utilisateur bouge un meuble pendant qu'un autre ouvre change sa couleur. Ainsi deux utilisateurs peuvent modifier de manière concurrentielle des propriétés indépendantes du même objet (dans l'exemple l'emplacement et la couleur).
  - Au niveau 3.2 deux utilisateurs peuvent agir sur l'objet de manière codépendante : la réaction résultante de l'objet est ensuite une combinaison des différentes entrées. Ainsi deux utilisateurs peuvent modifier de manière concurrentielle des propriétés identiques ou liées. Ce niveau est requis pour effecteur des manipulations coopératives. Il faut ensuite trouver là aussi un moyen de faire comprendre aux utilisateurs que les modifications de l'objet ne sont pas dues uniquement à leur manipulation, comme nous l'avons abordé dans la section précédente.

    Pour ce niveau, il faut introduire la notion de simultanéité. En effet on doit déterminer quand deux actions sur un objet sont simultanées et doivent donc être combinées. Il faut que lorsqu'une machine détecte qu'il y a simultanéité, toutes les autres le détecte également sous peine d'incohérence. Par ailleurs si la latence est trop grande, les utilisateurs auront du mal à collaborer car il ne recevront la réaction de leur partenaire que tardivement.

1.3.1.3 Localisation de la collaboration dans un EVF

De plus, en ce qui concerne les EVFC, la coopération peut intervenir à des niveaux différents, en fonction de l'objectif de la formation :

- le niveau de l'apprentissage : dans ce cas plusieurs apprenants vont apprendre ensemble, il s'agit donc d'un apprentissage collaboratif. Les apprenants vont pouvoir s'observer les uns les autres, discuter de l'objet de la formation et éventuellement s'aider. L'environnement est alors collaboratif, mais la collaboration n'est pas nécessairement une compétence à acquérir. On retrouve beaucoup d'environnements virtuels qui utilisent ce niveau de collaboration dans le champ de l'éducation, où l'apprentissage collectif est souvent utilisé (ex : NICE [JRL+98, RJL+97] conçu pour favoriser l'apprentissage du développement et de la croissance des plantes en biologie).
- le niveau du scénario : c'est le cas lorsqu'il s'agit d'apprendre une procédure collaborative par exemple. La collaboration fait alors partie des compétences à acquérir. Dans ce cas, il y a une différenciation entre l'activité des différents acteurs de l'EVFC, grâce à la notion de rôle. Ils vont alors avoir des actions différentes à réaliser. Les apprenants vont devoir partager le scénario, se répartir les tâches, coordonner leurs actions. On peut alors parler de collaboration orientée tâche. Certains environnement virtuels ayant pour but la modélisation d'équipe d'agents pour la résolution de problème entrent dans cette catégorie ; par exemple dans [Zha07] des agents représentant des vaisseaux doivent collaborer

pour tuer des wumpus (sortes de monstres) et collecter de l'or.

– le niveau d'une action : la collaboration peut intervenir non seulement au niveau du scénario global, mais également au niveau d'une action particulière que l'on peut alors qualifier, elle aussi, de collaborative. Cela peut être le cas par exemple lorsqu'il s'agit de manipuler à plusieurs un même objet (exemple : manipulation des pièces d'un véhicule dans le projet PERF-RV, voir figure 1.9). Les environnements virtuels ayant pour but le prototypage collaboratif par exemple proposent ce type de collaboration.

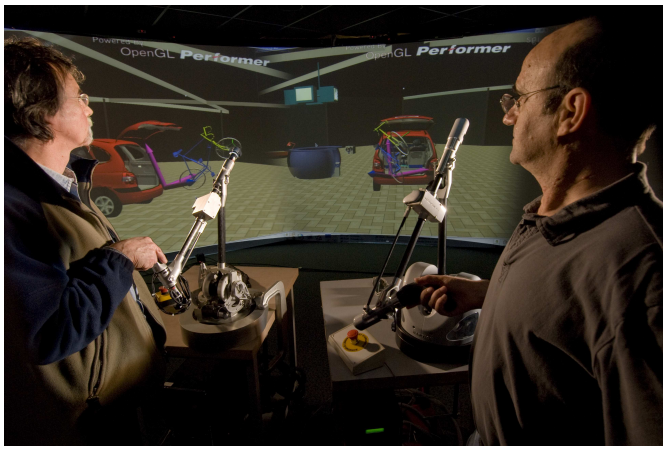


FIG. 1.9 – Manipulation haptique collaborative, projet PERF-RV. (c) CNRS Photothèque - Hubert RAGUET

1.3.1.4 Nature des collaborateurs

Il est intéressant d'analyser également la nature des personnes qui collaborent. En effet, dans un environnement virtuel, on peut trouver plusieurs types d'agents pouvant collaborer : des utilisateurs, des humains virtuels voire des objets "intelligents" (des robots par exemple). La collaboration ne se fait pas la même manière entre des humains virtuels qu'entre des utilisateurs réels par exemple et les contraintes ne sont pas les mêmes pour faire en sorte que chacun des collaborateurs comprenne ce qui se passe. Les échanges de messages, par exemple pour déclencher la collaboration, pourront être de natures différentes. Dans les EVFC, les collaborateurs peuvent être soit des utilisateurs réels (UR), soit des humains virtuels (HV). Trois types différents de collaboration peuvent alors apparaître : une collaboration entre utilisateurs réels (UR/UR), une collaboration entre humains virtuels (HV/HV) et enfin une collaboration entre un utilisateur réel et un humain virtuel (UR/HV).

1.3.1.5 La collaboration dans les 5 EVFC étudiés

Nous avons vu que dans un EVFC la collaboration peut notamment intervenir dans l'apprentissage et/ou dans le scénario. Nous allons nous servir de ce critère pour classer les cinq EVFC que nous allons étudier dans cet état de l'art. Nous préciserons également la complexité

de la collaboration (niveau 1 à 3) ainsi que la nature des collaborateurs. Il existe trois grandes familles d'EVFC : ceux qui proposent un apprentissage collaboratif d'un procédure individuelle, ceux qui proposent un apprentissage individuel d'une procédure collaborative et ceux qui proposent un apprentissage collaboratif d'une procédure collaborative.

- La première catégorie d'EVFC propose exclusivement un apprentissage collaboratif. C'est le cas dans COVET, un EVFC simple qui permet de former plusieurs apprenants simultanément à une procédure de maintenance individuelle. Dans cet EVFC un seul utilisateur peut interagir avec les objets de la scène et les autres se contentent de regarder, de discuter entre eux de la procédure en train d'être accomplie. Cette collaboration se situe donc au niveau de l'apprentissage, qui est collectif, et a lieu uniquement entre utilisateurs réels ; la procédure, elle, n'est pas collaborative. Cet EVFC exploite le niveau 1 de la classification de Margery puisqu'elle exploite essentiellement la communication. Ce niveau est en effet bien adapté à un apprentissage collaboratif. Dans le robinet virtuel d'EDF, on retrouve cet apprentissage collaboratif puisque les apprenants cherchent collectivement à appliquer une démarche de diagnostic de panne afin de trouver la cause du dysfonctionnement du robinet industriel. En revanche cet environnement de formation n'est pas multi-utilisateur : les apprenants ne sont que des observateurs extérieurs, tous réunis dans la même pièce et la communication ne passe pas par l'EVF, elle se fait en direct.
- La deuxième catégorie d'EVFC propose un apprentissage individuel (un seul apprenant à la fois) mais suivant un scénario collaboratif. Dans l'environnement de Dugdale l'apprenant joue le rôle du chef d'une équipe de pompiers. Plusieurs autres participants sont nécessaires dans ce scénario et les concepteurs ont choisi de les faire tous jouer par des utilisateurs réels. Il y a donc un seul apprenant, mais le scénario lui est collaboratif. Là encore la collaboration a lieu uniquement entre utilisateurs réels. Dans MRE il y a un seul formé qui doit collaborer avec des humains virtuels afin de prendre les bonnes décisions dans une situation de crise. L'apprentissage est donc également individuel. En revanche la collaboration a lieu cette fois entre un utilisateur réel et des humains virtuels et également entre humains virtuels. Dans ces deux environnements, le niveau 1 de la collaboration est bien exploitée avec des communications très présentes et assez libres entre les différents participants.
- La troisième catégorie d'EVFC propose d'apprendre collaborativement une procédure collaborative. C'est le cas dans SECUREVI et également dans la version de Steve pour gérer le travail en équipe. Ces deux EVFC ont plusieurs points communs, notamment concernant leur exploitation des niveaux de collaboration. Le niveau 1 est présent car l'apprenant doit donner des ordres avec une communication très codifiée et scénarisée. L'acte de communication est une action primitive au même titre que les actions physiques dans l'environnement (comme par exemple l'action de presser un bouton). De plus, dans la version collaborative de Steve, l'apprenant peut communiquer avec son tuteur Steve et lui poser des questions ou lui demander de l'aide. Le niveau 2 est le niveau principal de ces environnements puisque les utilisateurs agissent physiquement dans l'environnement et ils doivent apprendre à coordonner leurs actions et à se synchroniser, à effectuer leurs propres tâches au bon moment, tout en agissant sur des objets de l'environnement. Dans Steve, la collaboration est d'ailleurs décrite comme une collaboration orientée tâche. En

effet, c'est toujours dans le but d'accomplir une tâche faisant partie de la procédure que plusieurs agents peuvent être amenés à collaborer. Dans SECUREVI tous les niveaux de collaboration sont présents puisque le niveau 3 est également présent. En effet, certaines actions impliquent que plusieurs agents agissent sur le même objet (par exemple pour porter un objet lourd à plusieurs).
D'autres types d'environnements de formation permettent également d'apprendre collaborativement une procédure collaborative comme par exemple les simulations de combat collectives (ex : Twilight City) où formés et humains virtuels peuvent être mélangés au sein d'une même équipe et doivent collaborer pour accomplir leur mission. Dans tous ces environnements plusieurs apprenants peuvent être présents et les autres acteurs sont joués par des humains virtuels. Toutes les combinaisons possibles de collaboration, en fonction de la nature des collaborateurs, sont alors présentes : entre utilisateurs réels, entre un utilisateur réel et un humain virtuel et entre humains virtuels.

On voit donc que la collaboration peut prendre des formes très variées au sein d'un EVFC et ce en fonction de la finalité de cet EVFC. L'apprentissage peut être collaboratif, le scénario également, les acteurs de la collaboration peuvent être réels ou virtuels, et tous les niveaux de la collaboration peuvent être présents.

1.3.2 Un scénario multi-utilisateur et modulable

1.3.2.1 Modèles théoriques : travail en équipe et organisation

Le scénario collaboratif décrit une procédure à réaliser collectivement, par une équipe. La procédure collaborative représente donc la tâche à réaliser par l'équipe, son objectif. Avant de décrire l'objectif de l'équipe, nous allons donc nous intéresser à l'organisation d'une équipe, puis au raisonnement des agents de cette équipe.

Afin de poser un cadre théorique pour l'analyse du scénario collaboratif dans les EVF, il convient donc d'analyser deux domaines de recherche. Le premier concerne les modèles organisationnels : ces modèles ont pour but de décrire les liens sociaux qui existent entre plusieurs personnes devant réaliser un travail de groupe, comment ils s'organisent pour travailler ensemble (comment décrire les responsabilités de chacun par exemple). Le deuxième axe théorique concerne les théories de travail en équipe. Il s'agit là d'exprimer les mécanismes qui font que plusieurs membres d'une équipe vont réussir à accomplir des objectifs communs, en se répartissant les tâches. Ces modèles permettent ainsi d'étendre les raisonnements individuels des agents à des raisonnements à propos de l'équipe.

Modèles organisationnels Une première problématique est de modéliser l'organisation des agents, c'est à dire leurs liens sociaux. Pour cela, plusieurs modèles organisationnels existent. Dans les différents modèles le même mot peut recouvrir des concepts très légèrement différents (par exemple le concept de rôle ou celui de groupe) il convient donc de préciser les définitions de ces notions pour chacun des modèles abordés.

On peut par exemple citer le modèle Aalaadin [FG98] dont les concepts centraux sont l'agent le groupe et le rôle (voir figure 1.10). Un agent est une entité qui joue des rôles au sein de groupes (peu importe la nature de la modélisation de cette entité). Un groupe est un

regroupement d'agents. Un rôle est la représentation abstraite d'une fonction, d'un service ou d'une identification d'un agent au sein d'un groupe. Un agent peut entrer dans un groupe pour y jouer un rôle si une fonction d'acceptation est évaluée à vrai. En particulier une fonction d'admission intéressante proposée est l'acceptation conditionnée par les compétences (il y a un jeu de compétences requises).

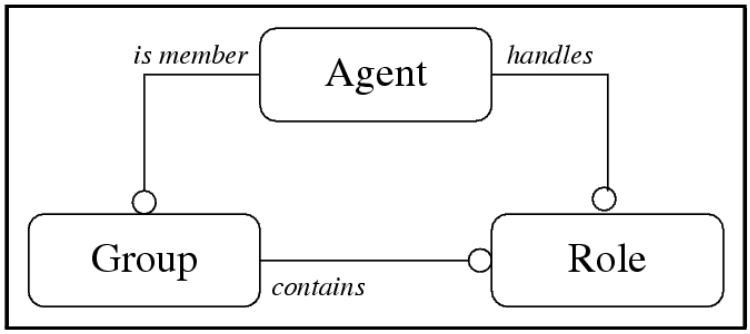


FIG. 1.10 – Le modèle Aalaadin d'après [FG98]

Le modèle MOISE [HBSS00] repose quant à lui sur les concepts de rôle, de lien organisationnel et de groupe. Un rôle est associé à un ensemble de missions. Une mission est composée de buts, de plans, d'actions et de ressources. Le plan est défini par un but à atteindre et représenté par un graphe orienté où chaque nœud est une action ou un sous-but. Une mission a également une force qui peut prendre deux valeurs : obligation ou permission ; obligation signifie que l'agent jouant le rôle doit réaliser la mission alors que permission signifie qu'il peut la faire. Les rôles contraignent les possibilités de comportement de chaque agent dans le système. Un lien organisationnel permet de structurer les échanges sociaux entre les agents ; il s'agit d'un arc orienté entre deux rôles. Il est constitué par un type, un contexte d'utilisation (les missions de chaque rôle pour lesquelles ce lien peut être appliqué) et un ensemble de contraintes. Le groupe est lui constitué par un ensemble de rôles et un sous ensemble des misions de ces rôles.

Le modèle MASCARET [Que02] repose quant à lui sur quatre concepts : l'organisation, le rôle, l'agent et l'élément de comportement (voir figure 1.11). L'organisation sert de facteur structurant, elle fournit un cadre aux interactions des agents qui en font partie. Les auteurs mentionnent deux types d'organisation : l'organisation sociale et l'organisation physique. Les rôles désignent les responsabilités d'un agent dans l'organisation. Ces responsabilités sont définies par un ensemble d'éléments de comportement que doit adopter l'agent jouant le rôle. Un agent doit avoir les capacités requises pour utiliser ces éléments de comportement (un rôle impose donc des pré-requis). Ce modèle a été notamment utilisé pour créer l'EVFC SECUREVI.

Dans les différents modèles d'organisation, on retrouve des concepts communs, la notion d'agent, qui est l'unité de base du modèle, la notion de rôle, et une notion de groupe ou d'organisation pour regrouper plusieurs rôles. L'utilisation du rôle, en revanche, varie un peu d'un modèle à l'autre. Parfois l'agent doit avoir des compétences particulières pour pouvoir jouer un rôle (dans Aalaadin cela dépend de la fonction d'admission utilisée, dans MASCARET il y

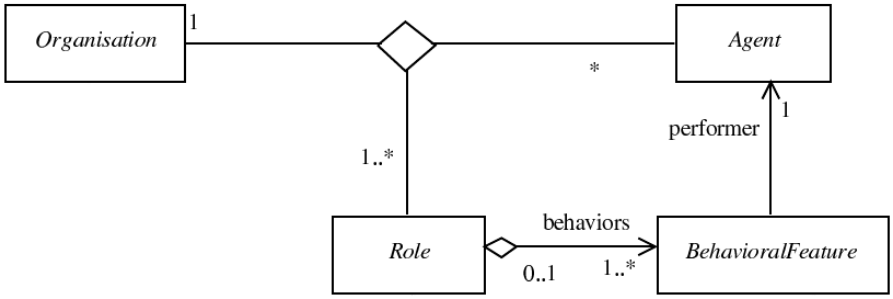


FIG. 1.11 – Le modèle d'organisation de MASCARET

a des pré-requis). Dans le modèle Aalaadin, le rôle est défini dans un sens assez large, il peut en effet représenter des services offerts (donc être traduit par des actions possibles) ou simplement servir à identifier un agent au sein d'un groupe. En général le rôle sert à représenter un ensemble d'actions, soit à accomplir impérativement, soit permises. On peut noter ici que dans un environnement virtuel collaboratif, le rôle est souvent défini comme un ensemble de droits qui indiquent les opérations que l'agent est autorisé à faire [YYWP05]. Mais dans ce type d'environnement virtuel, il n'y a pas de scénario à suivre ni de but formalisé à atteindre, juste des actions possibles.

Travail en équipe La caractéristique fondamentale qui différencie une équipe d'un simple groupe d'agents interagissant est que les membres d'une équipe partagent des objectifs communs. Cannon-Bowers et al.[CBSC90] ont proposé la notion de modèle mental partagé qui représente un savoir (une conscience mutuelle) partagé par les membres de l'équipe sur lequel ils vont pouvoir raisonner pour poursuivre des buts communs. Un modèle mental partagé peut par exemple contenir des informations sur l'état et l'activité de chacun des membres de l'équipe ainsi que des informations sur l'état d'avancement des tâches d'équipe. Ce modèle a été étayé par des études psychologiques comme par exemple le travail de Blickensderfer et al. [BCBS97] qui affirme que le modèle mental partagé est la clé pour un travail d'équipe efficace et que les membres d'une équipe ont des attentes les uns envers les autres.

Plusieurs théories ont été proposées pour expliciter la notion de travail en équipe. Ces théories définissent les mécanismes grâce auxquels les individus accomplissent des tâches d'équipes. Les individus doivent en effet raisonner sur les tâches d'équipe et transformer les objectifs de l'équipe en objectifs individuels. Les trois principales théories sur le travail en équipe [Cao05] sont la théorie des plans partagés (shared plans) [GK96], celle des intentions communes (joint intentions) [CL91] et celle de la responsabilité jointe (joint responsibility) [Jen92]. Dans ces théories, en plus de son propre état mental, l'agent gère une représentation de l'état mental de l'équipe (intentions d'actions ou plan partagé par exemple).

S'appuyant sur ces théories, des architectures informatiques ont été proposées pour modéliser le travail en équipe. On peut par exemple citer STEAM [Tam97] qui est un modèle

hybride qui s'appuie à la fois sur la théorie des intentions communes et celle des plans partagés. Ce système permet à des équipes composées uniquement d'agents de collaborer même en cas d'échec de certains plans. Cette architecture a été utilisée pour fournir des équipes coordonnées à opposer ou à allier à des humains dans des simulations militaires. La plupart des modèles de travail en équipe sont destinés à des systèmes d'agents et ne permettent pas d'intégrer des humains dans une équipe. Une exception est COLLAGEN [RS97] qui est basé sur la théorie des plans partagés et permet une collaboration entre un humain et un agent. Grâce à un mécanisme de reconnaissance de plans [LRS99] ce système permet à un agent d'interagir de manière intelligente avec un utilisateur en réduisant la communication nécessaire.

Ces modèles sont bien adaptés lorsqu'il s'agit de satisfaire un but donné, pouvant être décomposé en tâches d'équipes et en tâches individuelles (par exemple pour de la résolution de problème collaborative), mais faire du suivi de procédure est plus difficile. L'autre problème qui se pose est l'intégration d'utilisateurs réels. En effet, ces modèles supposent des représentations mentales et des mécanismes de raisonnement similaires pour tous les agents. La contrainte est donc de disposer d'une représentation ou d'un modèle symétrique entre l'humain réel et l'humain virtuel.

A ces modèles il faut parfois ajouter la notion de négociation, qui peut être utile en cas de conflits sur les ressources par exemple ou sur les moyens d'atteindre un objectif. Il faut alors utiliser un langage commun permettant de négocier entre les agents pour qu'ils se mettent d'accord.

Un exemple intéressant : CAST CAST (Collaborative Agents for Simulating Teamwork)[YYI+01] est une architecture multi-agents pour simuler le travail en équipe. Elle permet d'intégrer des humains à des équipes d'agents informatiques. Le savoir sur l'équipe est représenté grâce au langage MALLET (Multi-Agent Logic Language for Encoding Teamwork). Il est ainsi possible de donner des informations sur la structure de l'équipe (les membres de l'équipe, les rôles joués par chacun, les capacités ainsi que les responsabilités de chacun) ainsi que sur l'activité de l'équipe (plans et opérateurs individuels ou d'équipe). Une capacité est une association statique entre un agent et un opérateur, elle indique ce que l'agent sait faire. Une responsabilité est aussi une association entre un agent et un opérateur mais elle signifie que chaque fois que l'action aura besoin d'être faite (si elle figure dans un plan) alors l'agent sera responsable de son exécution. Les capacités et les responsabilités ne sont donc pas liées aux rôles mais directement aux agents. L'activité d'une équipe est représentée par des plans (qui possèdent des préconditions et des effets) décomposés en procédures (agencement séquentiel, parallèle, conditionnel ou itératif d'actions) puis en opérateurs (actions atomiques).

L'assignation des acteurs aux actions peut se faire de quatre manières différentes : il est possible d'assigner directement un agent, d'assigner un rôle (sachant que plusieurs agents peuvent jouer le même rôle) ou de ne pas préciser d'assignation. Dans ce cas c'est, s'il y en a, les acteurs qui ont la responsabilité de l'action qui sont considérés comme candidats pour l'action ou, en dernier recours, tous les membres de l'équipe. Il est également possible d'utiliser, à la place des rôles, des variables de rôle qui permettent de préciser un critère particulier évalué dynamiquement pour sélectionner un agent parmi ceux jouant un rôle donné. Un algorithme de sélection dynamique des rôles se charge alors de sélectionner le ou les agents qui vont faire l'action parmi les agents considérés ; Cette sélection se fait en fonction du type de l'opérateur

(AND, OR, XOR). Si c'est un opérateur AND ,tous les candidats vont faire l'action et il y a alors une synchronisation entre eux. Si c'est un opérateur OR, alors tous les candidats vont essayer de faire l'action jusqu'à ce que l'un d'eux réussisse. Si c'est un opérateur XOR, un seul candidat doit faire l'action, il y aura alors négociation entre les agents. Chaque agent possède un état mental partagé qui contient l'état global de l'équipe représenté par un réseau de Pétri. Chaque agent peut ainsi raisonner dynamiquement sur cet état et adapter son comportement en conséquence en anticipant les actions et les attentes des autres par exemple. Le travail en équipe peut ainsi être représenté de manière flexible car les agents sont capables de s'adapter à l'état courant de l'équipe et de l'environnement. L'algorithme de sélection dynamique des rôles évoqué plus haut est lancé par chaque agent et se base sur le modèle mental partagé de l'équipe.

CAST a été utilisé tout d'abord pour créer des équipes constituées uniquement d'agents informatiques. Un humain devait ensuite être intégré à une équipe d'agents, dans un cadre de formation. Un agent (nommé CAST-ITT) représentant un ITS (Intelligent Tutoring System) était alors chargé de faire le lien entre l'équipe et l'humain. Cette application décrite dans [MV01] devait être implémentée puis validée mais il n'y a pas eu de publication ultérieure à ce sujet. CAST a également été couplé à DDD, un logiciel pour simuler des tâches de contrôle et de commandement [XVMP03]. L'objectif était de faire de la formation assistée par ordinateur. Cependant, seul un ensemble réduit des possibilités de CAST est utilisé ; notamment, la sélection dynamique des rôles n'est pas exploitée et l'activité des apprenants n'est pas modélisée. L'architecture CAST a également été étendue grâce au langage RoB-MALLET (Role-Based Multi-Agent Logic Language for Encoding Teamwork)[Cao05, ZCV06] qui intègre des fonctionnalités supplémentaires à propos des rôles. Le concept de position est intégré au langage (une catégorie de personnes qui possède un ensemble de comportements). Un agent possède une position statique (ex : combattant) mais par contre les rôles qu'il joue sont attribués dynamiquement (ex : combattant 1 et combattant 2). Il est également possible de choisir dynamiquement le rôle associé à une action.

L'aspect qui nous semble intéressant dans CAST est la flexibilité amenée par la sélection dynamique du ou des agents qui vont réaliser une action en se basant sur des critères évalués dynamiquement. Les procédures peuvent ainsi être adaptées au contexte d'exécution. En revanche l'échange de messages nécessaire à la synchronisation (pour les opérateurs AND) ou à la négociation (pour les opérateurs XOR) paraît peu compatible avec l'intégration d'humains dans les équipes. L'autre point intéressant est la notion de modèle mental partagé qui permet à tous les membres de l'équipe de raisonner sur un état commun pour l'équipe et qui leur permet notamment d'anticiper les attentes de leurs coéquipiers.

Travail d'équipe procédural Certains modèles de travail en équipe sont spécialisés pour le cas d'un travail procédural. Dans le modèle MASCARET par exemple, des extensions sont réalisées pour modéliser des équipes d'agents (utilisateurs ou humains virtuels) dans le cadre d'un travail procédural et collaboratif. La notion d'équipe remplace alors celle d'organisation. Une équipe est composée de procédures, qui représentent la coordination optimale des différents rôles pour la résolution d'un problème. La procédure est un ensemble de contraintes qui décrivent l'agencement des actions (qui dérivent des éléments de comportement du modèle de base) effectuées par les agents jouant un rôle dans l'équipe. Les procédures représentent

donc des tâches collaboratives pour l'équipe et sont décomposées en tâches individuelles : les actions. On voit ici que le lien entre la procédure et les rôles est fait grâce aux actions, or une action telle qu'elle est définie dans ce modèle n'appartient qu'à un seul rôle. La procédure décrit donc l'agencement des actions mais avec une répartition fixe entre les rôles. Or, dans leur théorie des rôles, Biddle et Thomas [BT66] suggèrent que certaines actions ne sont pas forcément prédéterminées pour être associées à un rôle donné ; une action peut, en effet, être associée à plusieurs rôles et la détermination du rôle qui réalisera l'action est effectuée dynamiquement en fonction de la situation. Zhang [Zha07] a suivi cette recommandation pour créer son modèle d'équipe d'agents orienté rôle, RoB-MALLET[2]. Pour cela, il introduit la notion de variable de rôle qui permet de spécifier plusieurs rôles pouvant réaliser une action donnée. La détermination précise de l'agent qui effectue l'action est ensuite réalisée en évaluant dynamiquement une condition spécifiée (ex : celui qui est le plus près).

Dans un EVFC avec une procédure à respecter, la procédure modélise la tâche collaborative à réaliser par l'équipe. L'état du travail d'équipe est donc représenté par l'état d'avancement de la procédure et le raisonnement des agents sur le travail d'équipe revient alors à analyser cette procédure. La partie gestion du travail d'équipe peut alors être factorisée au sein d'un module chargé de suivre le déroulement de la procédure et d'analyser les futures actions possibles. Ce module symbolise alors un modèle mental partagé. La plupart des EVFC supposent que tous les participants partagent la même volonté d'accomplir la procédure et les notions d'intentions jointes ou de plans partagés décrites dans les théories de travail en équipe peuvent donc être incluses directement dans la représentation de la procédure à savoir le scénario collaboratif.

1.3.2.2 Description d'un scénario

Nous allons commencer par donner la définition des termes procédure et scénario afin de bien comprendre la distinction que nous faisons. La procédure représente la suite d'actions nominales spécifiée par l'industriel pour réaliser une opération donnée (par exemple une opération de maintenance). Chez Nexter les procédures sont décrites dans des cartes de travail (cf annexe A). Le scénario représente la suite d'instructions chargées au minimum de représenter la procédure. Dans un EVF il faut donc traduire la procédure en un scénario. Cette traduction peut impliquer par exemple d'extraire des actions implicites de la procédure ou de désambiguïser certaines actions. Le scénario peut aussi contenir des informations supplémentaires par rapport à la procédure comme différentes branches pour modéliser différentes pannes possibles ou des cas d'erreurs. Par exemple, dans un scénario décrit grâce au langage LORA [MA06], il est possible de spécifier des actions interdites.

Un scénario doit permettre de décire l'agencement à la fois temporel et logique des actions. Dans le début de cet état de l'art nous avons rappelé les méthodes de scénarisation existantes pour un EVF. Rappelons que dans les EVF deux méthodes sont principalement utilisées : une description complète pouvant être représentée sous forme de graphe par exemple et une description par tâches avec une décomposition en sous-tâches, en formalisant les buts et les causes ; la procédure pouvant alors être vue comme un ensemble de contraintes. A ces mé-

[2]Le modèle RoB-MALLET a été utilisé pour créer des équipes constituées uniquement d'agents virtuels, dans le but de faire de la résolution collaborative de problèmes par exemple. A notre connaissance il n'a pas été utilisé au sein d'un EVFC.

thodes s'ajoute une méthode hybride qui consiste en une description partielle de la procédure. Elle utilise des préconditions et laisse certaines actions non écrites dans le scénario dont la planification sera laissée aux acteurs en partant d'actions de base. Ces différentes méthodes sont également utilisées pour décrire un scénario collaboratif. Elles diffèrent simplement par l'ajout d'informations qui définissent qui peut faire chaque action. Lors de notre analyse des différents EVFC étudiés nous préciserons donc le moyen choisi pour représenter le scénario.

1.3.2.3 Scénario collaboratif dans les EVFC

Un scénario multi-utilisateur doit traduire l'asymétrie dans les activités des différents acteurs. La notion de rôle permet alors de différencier l'activité des individus.

Pour mieux comprendre la notion de rôle, on peut prendre des exemples de rôles réellement utilisés dans les EVFC. Dans MRE, l'apprenant joue le rôle du *lieutenant*, le rôle indique donc sa fonction sociale, son métier. Dans Steve, les rôles définis dépendent de la tâche, par exemple la personne jouant le rôle de l'*officier de la salle des moteurs* pour la tâche de gestion de la perte de pression d'huile va jouer le rôle de l'*opérateur de la console centrale* dans la sous tâche de transfert à la salle de contrôle centrale. Ainsi, dans ce cas, un rôle est associé à une tâche et un agent va donc jouer plusieurs rôles en fonction des tâches qu'il va accomplir. Dans GASPAR les rôles représentent également des métiers comme *pilote d'hélicoptère*.

Outre la sémantique associé au rôle, il est intéressant de remarquer dans les EVFC des distinctions sur les prérogatives des rôles. Comme nous l'avons vu dans la section précédente, un rôle peut définir les responsabilités d'un agent dans une organisation, en étant associé à un ensemble d'actions ou de missions que l'agent doit réaliser. Le rôle peut aussi être défini comme une classe de comportements ou de services que propose un agent dans le système ; il traduit ainsi un savoir faire spécifique et amène des compétences particulières. Un rôle permet donc de définir les actions que l'acteur sait, peut ou doit faire. Dans Steve, le rôle permet uniquement de savoir quelles sont les actions du ressort de l'agent, un rôle peut donc être vu comme un ensemble d'actions que l'acteur doit réaliser. Dans SECUREVI, un rôle contient un ensemble d'éléments de comportements. Ces éléments de comportements sont de deux types : les actions métiers et les actions génériques. Les actions métiers sont des actions spécifiques au domaine qui interviennent dans la procédure, par exemple l'action *arroser un feu*. Les actions génériques sont des actions plus basiques et ne sont pas représentées dans la procédure, par exemple l'action *prendre un objet* ou *aller à un point*. La procédure décrit l'agencement temporel des actions métiers, tandis que les actions génériques sont utilisées par un agent pour calculer un plan servant à satisfaire la précondition d'une action métier de la procédure. Dans ce modèle, un rôle impose donc des contraintes (des actions à effectuer dans la procédure) mais il apporte également des possibilités d'actions (les actions génériques). Dans GASPAR, un rôle regroupe un certain nombre de méthodes qui représentent les opérations que les agents sont capables d'effectuer. Un rôle traduit donc des possibilités et c'est la procédure qui ensuite impose des responsabilités aux agents.

Dans un scénario de formation, le rôle permet de définir les tâches qu'un acteur doit effectuer. Il permet donc de déterminer qui fait quoi dans la procédure. Actuellement dans les EVFC, un seul et unique rôle est assigné à chaque action du scénario. La personne jouant le rôle spécifié est alors la seule à pouvoir réaliser cette action. Cela permet donc, à tout instant,

de savoir qui doit réaliser la prochaine action du scénario. Dans SECUREVI la procédure décrit l'agencement temporel d'éléments de comportement représentant des actions métiers, or ces actions métiers sont associées à un seul et unique rôle, comme le montre la figure 1.12. Dans Steve, un champ *roles* est ajouté à la description des tâches. Il permet de préciser les

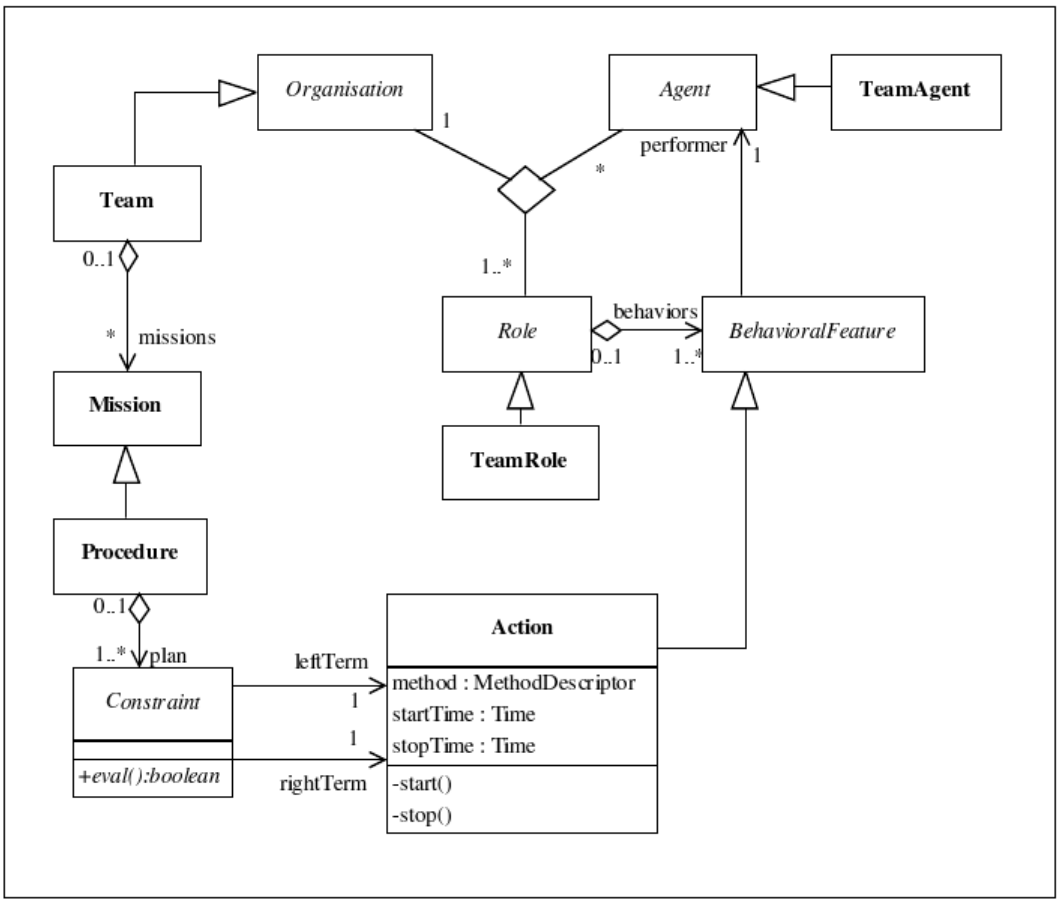


FIG. 1.12 – Le modèle MASCARET : représentation du travail procédural en équipe

rôles nécessaires pour accomplir des sous-tâches individuelles. Si les sous-tâches à accomplir sont également des tâches d'équipe, il faut alors préciser les rôles de la tâche mère qui doivent accomplir chacun des rôles de la tâche fille (voir figure 1.13). Dans GASPAR le scénario est décrit par un diagramme d'activité UML, avec un couloir par rôle. Cette description a l'avantage de permettre de visualiser facilement les liens (en particulier temporels) entre les activités des différents acteurs. Mais avec une telle représentation, on voit bien qu'une action ne peut appartenir qu'à un seul couloir, et donc elle ne peut être associée qu'à un seul rôle (voir figure 1.14). Dans MRE, deux rôles sont en fait spécifiés pour une tâche donnée : la tâche est associée à un membre de l'équipe responsable de son exécution ainsi qu'un autre agent qui doit donner l'ordre d'exécution. Cela permet de refléter la hiérarchie militaire où avant de pouvoir réaliser

Task loss-of-fuel-oil-pressure
Steps transfer-thrust-control-ccs, ...
Causal links ...
Ordering ...
Roles
eoow: (transfer-thrust-control-ccs pacc), ... ;
engrm: (transfer-thrust-control-ccs scu), ...

Task transfer-thrust-control-ccs
Steps press-pacc-ccs, press-scu-ccs
Causal links
press-pacc-ccs achieves ccs-blinking for press-scu-ccs
press-scu-ccs achieves thrust-at-ccs for end-task
Ordering press-pacc-ccs before press-scu-ccs
Roles pacc: press-pacc-ccs; scu: press-scu-ccs

FIG. 1.13 – Steve : représentation d'une tâche (à gauche) et d'une sous-tâche d'équipe (à droite)

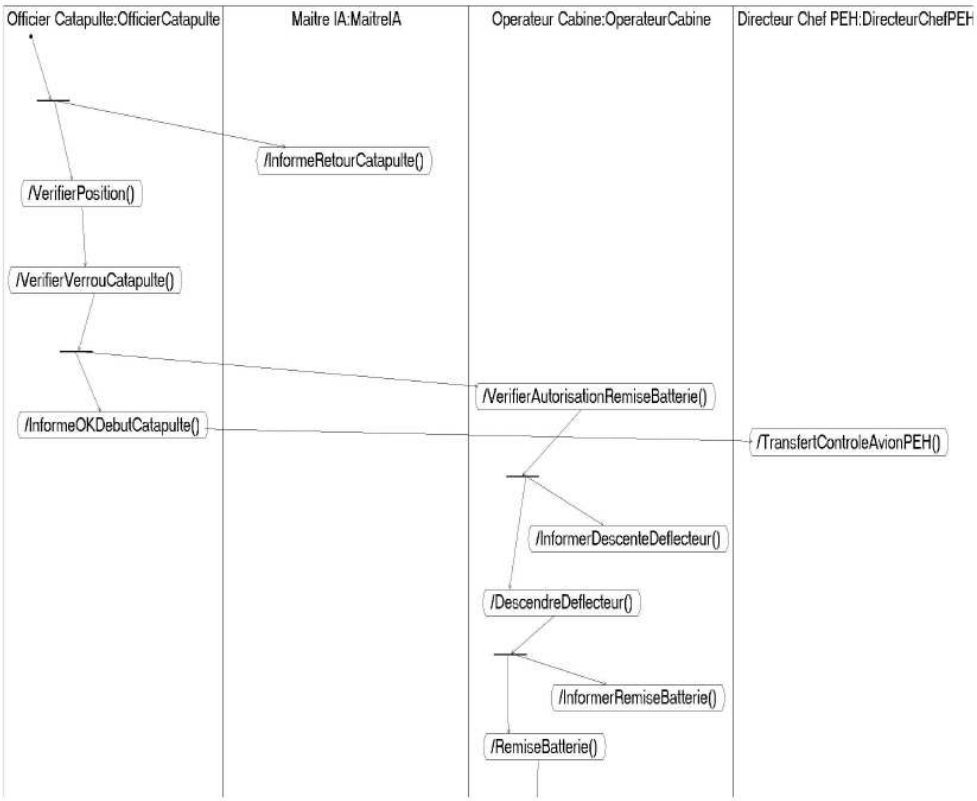


FIG. 1.14 – GASPAR : représentation de la procédure sous forme de diagramme d'activité UML

une tâche, un apprenant doit d'abord avoir reçu l'ordre correspondant.

La vision classique dans un EVFC est donc de considérer que, de la même manière que dans un scénario de théâtre un rôle représente un ensemble de répliques à dire, dans un scénario de formation, un rôle représente un ensemble d'actions à réaliser. Une action du scénario n'est alors associée qu'à un seul rôle. Pourtant, dans certaines procédures, le rôle est plutôt lié aux personnes qu'aux actions. Dans ce cas la succession d'actions à réaliser prime sur l'affectation de ces actions aux personnes présentes : peu importe qui réalise les actions pourvu qu'elles soient réalisées dans l'ordre adéquat. Il peut ainsi y avoir plusieurs réalisations possibles en terme de répartition des actions entre les différents acteurs. Parfois cela n'est visible qu'au niveau d'une action ponctuelle (par exemple il faut à un moment donné ouvrir la porte, mais peu importe l'acteur qui le fait). Dans d'autres procédures, certaines actions sont censées être réalisées par un rôle donné, mais en pratique, si la personne jouant ce rôle n'est pas disponible, une autre personne peut la remplacer. Lorsque le chef n'est pas disponible par exemple il se peut que le sous-chef puisse le remplacer pour certaines actions. Parfois c'est toute la procédure qui est concernée : l'ensemble des tâches à réaliser est bien fixé, mais la répartition de ces tâches entre les différentes personnes présentes peut varier. Prenons comme exemple la procédure de montage d'un meuble. Il y a une succession d'actions à réaliser, mais peu importe qui les réalise. La répartition des tâches n'est pas unique, certaines actions peuvent même être faites à plusieurs ou individuellement, en fonction du contexte. Le but commun à tous les participants est alors la réalisation de la procédure qui est l'élément central et les acteurs sont plutôt vus comme des ressources pour les actions. Cependant les EVFC rencontrés dans notre état de l'art ne permettent pas de modéliser de telles procédures ; ils se limitent aux procédures où l'attribution des actions aux acteurs est fixe.

Une fois la notion de rôle réifiée, il devient possible de spécifier le scénario collaboratif dans l'application, c'est à dire l'enchaînement des actions à réaliser par les différents acteurs. Ce scénario peut néanmoins être décrit de manière plus ou moins souple. Dans les EVFC, il existe principalement trois manières de décrire le scénario : une description complète et exhaustive des actions à réaliser, une description partielle en laissant implicite (grâce à des pré-conditions) le recours à certaines actions et une description minimale avec un seul but à atteindre en laissant implicite le moyen d'y arriver (explicitation de la causalité). La description complète permet d'imposer un déroulement précis des actions d'une procédure mais ne permet pas d'adapter le scénario à un contexte de réalisation différent de celui initialement prévu. Peu d'EVFC néanmoins utilisent cette forme de description à cause de sa longueur et de son manque de souplesse. La description partielle en laissant certaines actions implicites est un bon compromis, permettant une certaine flexibilité du scénario et une adaptation au contexte tout en respectant un ordonnancement strict des actions clés de la procédure. Dans SECUREVI les actions métiers (ex : arroser un feu) et les actions génériques (ex : se déplacer, prendre un objet) sont distinguées. La procédure décrit uniquement l'agencement des actions métier et chaque acteur se construit ses propres plans implicites à partir des actions génériques pour pouvoir satisfaire les pré-conditions des actions métiers de la procédure. D'autres EVFC préfèrent s'appuyer uniquement sur les liens de causalité entre les tâches pour décrire le scénario. Steve utilise un ensemble de buts décomposés en sous-buts jusqu'à atteindre une action dite primitive qui représente une action concrète dans l'environnement. Le scénario permet alors plusieurs réalisations différentes, qui ne respectent pas forcément la procédure de référence en

terme d'ordre des actions notamment. Dans MRE, le scénario permet quatre fins différentes, il n'impose pas de suivre une voie particulière. Grâce aux pré-conditions et à la possibilité de construire des plans implicites, certains scénarios permettent donc une première adaptation au contexte qui consiste à modifier l'ordre de réalisation de certaines actions. En revanche, dans les outils étudiés dans cet état de l'art, aucun scénario ne permet une adaptation au niveau de la répartition des actions entre les différents acteurs.

Par ailleurs le scénario collaboratif doit également décrire l'enchaînement temporel des actions. Pour cela il est possible d'avoir une gestion fine de l'ordonnancement des actions, par exemple en utilisant une logique temporelle sur les intervalles. Une gestion simplifiée consiste à ne considérer que la notion de séquence entre deux actions, grâce à des pré-conditions ou à l'ordre dans un graphe de scénario par exemple. Dans une première version de SECUREVI l'enchaînement des actions est décrit grâce aux opérateurs temporels définis par Allen [All83]. Mais dans ce cas seuls des humains virtuels sont présents dans l'environnement, car il n'est pas possible d'imposer à un utilisateur réel de respecter ces contraintes. Dans la deuxième version, celle qui permet la présence d'utilisateurs réels pour faire de la formation, les enchaînements temporels utilisés sont la séquence et le parallélisme. Dans Steve il est simplement possible de spécifier qu'une tâche doit être réalisée avant une autre. Les autres EVFC n'ont pas de gestion spécifique de l'ordonnancement temporels des actions, mis à part la précédence imposée par la notion de pré-condition.

La description de la procédure à accomplir peut parfois ne pas être formalisée au sein de l'EVFC. Dans l'EVFC de Dugdale, il n'y a pas de description du scénario au sein de l'application ; c'est donc une personne extérieure, un expert ou un formateur par exemple, qui va surveiller l'avancement de l'apprenant et comparer ses actions à celles attendues.

Pour décrire un scénario collaboratif, il convient donc de préciser l'utilisation que l'on souhaite faire de la notion de rôle, le mode de description du scénario qui va influencer sa flexibilité, ainsi que la précision que l'on souhaite atteindre quant à la description des contraintes temporelles entre les actions.

1.3.3 Des humains virtuels autonomes

1.3.3.1 Comment modéliser un humain virtuel ?

L'humain virtuel est en général un agrégat de modèles complexes, il faut alors définir quelles sont ses caractéristiques sur lesquelles on souhaite insister. Ce choix va dépendre de l'utilisation que l'on souhaite faire des humains virtuels et de l'objectif de l'application. On peut classer les différents axes de recherches sur les humains virtuels dans quatre catégories.

- physique. L'accent est ici mis sur le réalisme visuel de l'humain virtuel. Pour cela certains chercheurs se concentrent sur le rendu de certaines parties précises du corps de l'humanoïde (peau, cheveux, yeux [FGBB07], vêtements) et sur les moyens pour les animer. D'autres recherches s'orientent davantage sur le réalisme des mouvements et la crédibilité des postures de l'humanoïde [MKB08], c'est le cas de la biomécanique par exemple.
- relationnel. Lorsque l'humain virtuel doit interagir avec des utilisateurs et que l'on souhaite que ces derniers réagissent comme s'ils étaient face à d'autres utilisateurs, dans

le but de susciter des émotions par exemple, il est nécessaire de se concentrer sur l'expressivité du visage ou des gestes [RGH+06], sur le dialogue naturel, sur les gestes non intentionnels, sur la traduction des émotions.
- comportement. Pour d'autres applications, ce qui est important c'est de donner un comportement à l'humain virtuel [LD02], il peut s'agir de comportements réactifs (s'adapter à des situations données), de capacités cognitives avec de la planification d'actions et du raisonnement, de planification de mouvement pour planifier un déplacement (avec des stratégies d'évitement par exemple), et parfois d'apprentissage afin que l'humain virtuel puisse s'adapter à son environnement.
- fonctionnel. D'autres applications requièrent des capacités plus spécifiques pour un humain virtuel, il peut par exemple lui être demandé de suivre un scénario, de fournir une assistance à un utilisateur, d'avoir un comportement collectif crédible (simulation de foules [PPD07]).

Nous avons cité ici quelques-uns de ces travaux mais il en existe beaucoup d'autres, chacun s'orientant vers un ou plusieurs de ces facettes de l'humanoïde. Mais chacun de ces travaux a un coût de mise en place, aussi bien en ce qui concerne le temps d'intégration que le coût en terme de temps de calcul du processeur. Il n'est pas possible actuellement d'intégrer dans une même application les recherches à la pointe dans chacun de ces domaines. Il est donc important d'identifier quelles sont les caractéristiques de l'humanoïde importantes pour l'application. Dans cette thèse, nous nous focalisons sur la gestion de l'activité de l'humain virtuel dans le cadre d'une formation procédurale et collaborative. L'humain virtuel va donc devoir tenir compte du scénario, de l'état et de l'activité de ses coéquipiers et également de ses préférences de comportement pour choisir des actions à réaliser.

1.3.4 Interactions possibles

Un humain virtuel doit être capable d'agir dans l'environnement, pour cela il doit savoir quelles sont ses possibilités d'interaction avec les objets présents. Différentes techniques existent pour gérer les interactions entre un humanoïde et des objets de l'environnement. Certains auteurs proposent d'imiter le comportement des êtres humains, capables d'observer le monde et d'en déduire les interactions possibles d'après leur analyse et leurs expériences passées. Pour simuler un humain virtuel omniscient, des techniques d'IA peuvent être utilisées, combinées avec des techniques d'apprentissage. Dans ce domaine on peut citer les travaux de Hildebrand [HEHV03]. Il propose de créer ainsi un agent interactif, piloté par un utilisateur. Cet agent est modélisé grâce à un système expert, il possède ainsi un ensemble de règles qui définissent son savoir et les interactions possibles. L'utilisateur communique avec à cet agent en langage naturel, il peut par exemple lui donner un ordre. Si l'agent ne sait pas comment réaliser l'action (par exemple s'il ne sait pas situer l'objet sur lequel porte l'action), il va chercher de l'aide auprès de l'utilisateur qui va pouvoir lui fournir des indications sous forme de déclarations permettant d'augmenter la base de connaissances de l'agent. Cet agent interactif est ainsi capable d'apprendre de nouvelles informations sur son environnement ou sur des possibilités d'interaction. Cependant, ce type de technique, très coûteuse, reste difficile à appliquer dans une application temps réel et les recherches s'orientent davantage sur les environnements informés. L'idée est alors de distribuer une partie du savoir sur les interactions vers les objets

eux mêmes afin de décharger l'humain virtuel.

Dans ce domaine les *smart objects* de Kallmann [KT98] font partie des travaux de référence. L'idée est de déporter tout le savoir sur les interactions à l'intérieur des objets. L'information complète sur l'interaction est donc définie dans l'objet, aussi bien les fonctionnalités de l'objet (ses possibilités interactives) que la définition de l'animation (l'endroit où doit se positionner l'humanoïde pour pouvoir interagir par exemple). Un smart object est défini par 4 caractéristiques d'interaction :

- Les propriétés intrinsèques de l'objet : par exemple le poids de l'objet, une animation de ces différentes parties, etc.
- Les informations d'interaction : ce sont les informations utiles à l'agent qui va vouloir interagir avec l'objet (par exemple l'angle d'approche, les surfaces interactives, etc).
- Le comportement de l'objet : certains comportements de l'objet sont accessibles seulement si l'objet est dans un état précis. Par exemple une porte automatique peut se fermer seulement s'il n'y a personne qui la traverse et si elle est ouverte.
- Le comportement de l'agent : pour chaque comportement de l'objet on décrit le comportement associé que l'agent doit adopter pour interagir.

Lors d'une interaction, un smart object va prendre le contrôle de l'humanoïde et lui faire réaliser pas à pas les étapes nécessaires au bon déroulement de l'action.

Dans l'approche smart object, l'interaction est entièrement définie dans les objets, Badawi [BD04] propose dans son architecture STARFISH de répartir les informations entre l'humanoïde et les objets qui deviennent alors des *objets synoptiques*. Un objet synoptique est composé :

- D'actions STARFISH : ces actions peuvent être des actions atomiques (un jeu de 8 actions de base sont proposées) ou des actions complexes qui sont créées à partir des actions de base, grâce à un automate. Cette partie sert à décrire l'interaction.
- De surfaces interactives : elles définissent les surfaces de l'objet qui permettent des interactions pour l'humanoïde. Il peut s'agir de surface de contact pour saisir l'objet par exemple, ou alors d'endroits où se positionner pour interagir (espace situé devant une porte par exemple).

Les objets synoptiques contiennent une description de l'interaction. L'humain virtuel est alors capable d'exploiter ces informations pour adapter son interaction avec l'objet (il va par exemple adapter ses actions en fonction de sa morphologie). Deux humanoïdes n'auront donc pas forcement le même positionnement pour interagir avec un objet. Les informations sont ainsi réparties entre l'objet et l'humanoïde afin de simplifier la description de l'interaction au sein de l'objet et de laisser l'humanoïde gérer la partie réalisation qui va dépendre de ses propres caractéristiques et du contexte.

La dernière approche que nous pouvons citer est celle des PAR (Parameterized Action Representation) qui sont le fruit des travaux de Badler [BBA+00]. Les PAR servent à définir des actions paramétrables qui vont servir d'intermédiaire entre l'humanoïde et les objets. L'information sur l'interaction n'est donc plus répartie entre les différents participants à l'interaction mais elle est située entre les deux, dans une entité séparée. un PAR est défini par :

- Des objets physiques : les objets de l'environnement manipulés.
- Un agent : l'agent qui va exécuter l'action.
- Des conditions d'application : elles définissent l'état de l'environnement nécessaire à la

réalisation de l'action.
- Des préparatifs : il s'agit d'une liste de couples <condition, action>. Si la condition n'est pas vérifiée alors l'action associée est réalisée.
- Des étapes d'exécution : l'action à réaliser peut être une action primitive ou une action complexe qui sera constitué d'actions primitives, en parallèle, en séquence, ou grâce à une combinaison des deux.
- Des conditions de terminaison : une liste des conditions qui font que l'action est considérée comme exécutée.
- Des effets : mises à jour à effectuer sur l'état du monde une fois l'action effectuée.

Pour représenter les interactions possibles entre un humanoïde et des objets de l'environnement, il est donc possible de décrire ces interactions soit exclusivement à l'intérieur de l'humanoïde, soit à l'intérieur des objets, soit une partie dans l'humanoïde et une partie dans les objets ou encore dans des entités situées entre l'humanoïde et les objets. Ce choix va ainsi contraindre la spécification de l'environnement.

1.3.4.1 Les humains virtuels dans les EVFC

Dans un EVFC, les humains virtuels autonomes apportent plusieurs avantages : ils permettent de s'affranchir de la disponibilité de coéquipiers et peuvent aussi jouer un rôle pédagogique [Buc05]. Dans certains EVFC ces humains virtuels jouent systématiquement certains rôles du scénario, alors que dans d'autres EVFC ils sont interchangeables avec des utilisateurs réels. Il est également possible de leur laisser plus ou moins d'autonomie, vis à vis de la procédure notamment. Dans Steve et SECUREVI, les humains virtuels proposés servent à remplacer un coéquipier manquant et peuvent être interchangeables avec des utilisateurs réels. Les humains virtuels se contentent alors de suivre le scénario en réalisant les actions qui sont de leur ressort ou en remplissant la mission qui leur a été fixée. Dans Steve les humains virtuels peuvent également jouer le rôle de tuteur. Ils assistent alors un apprenant en lui donnant des conseils, en répondant à ses questions, en réalisant, au besoin, des actions à sa place. Dans MASCARET, le modèle qui est à la base de l'application SECUREVI, la notion de rôle pédagogique apparaît, indépendamment du rôle dans le scénario. En revanche dans SECUREVI cette possibilité n'a pas été exploitée. Dans MRE certains rôles sont systématiquement joués par des humains virtuels qui ne peuvent donc pas être remplacés par des utilisateurs réels. Dans MRE, les humains virtuels sont capables de raisonner, de tenir une conversation et de montrer des émotions. Les émotions peuvent influencer le comportement des humains virtuels.

Les EVFC qui ne proposent pas d'humains virtuels autonomes peuvent le justifier de différentes manières. Dans COVET, les humains virtuels ne sont pas nécessaires puisque le scénario est individuel, il n'y a donc pas de coéquipier à remplacer. Dans l'EVFC proposé par Dugdale, les concepteurs ont choisi de faire jouer tous les participants par des utilisateurs réels afin de profiter de leurs compétences en matière de résolution de problèmes notamment, ce qui permet de réduire la complexité de modélisation des acteurs. En revanche les personnages virtuels (avatars des utilisateurs) que contrôlent les utilisateurs sont tout de même dotés d'une certaine autonomie concernant la réalisation d'actions non intentionnelles (par exemple hocher la tête dans une conversation) et sont aussi capables de s'adapter au contexte (par exemple marcher plus vite près d'un feu).

La modélisation des capacités de raisonnement des humains virtuels dans les EVFC dépend de la modélisation de l'environnement ainsi que de celle du scénario. Cependant on retrouve des similarités, tous s'inspirent plus ou moins de la classique boucle perception, décision, action [Mal97]. La partie décision en revanche varie d'un environnement à l'autre.

Un agent Steve est constitué de trois modules (voir figure 1.15) :

– Perception : ce module garde une vue de l'état actuel du monde. Il surveille donc l'état de la simulation, les actions réalisés par les utilisateurs et les autres agents, la localisation de chacun ainsi que les messages vocaux échangés.
– Cognition : ce module est implémenté en utilisant l'architecture SOAR [LNR87] permettant de développer des comportements autonomes. Il interprête les données provenant du module de perception, choisi des buts appropriés, construit et exécute des plans pour accomplir ces buts et envoie des commandes au module de contrôle moteur.
– Contrôle moteur : il gère des commandes de base telles que se déplacer vers un objet.

Tous les agents partagent le même savoir sur les tâches. De plus les tâches primitives sont assignées à un seul rôle. Chaque agent va donc se construire le même modèle de tâche hiérarchique avec la même assignation des responsabilités. Comme les agents doivent tous suivre la même procédure, il n'y aura donc pas de conflits sur les choix d'action et chacun sait à tout moment ce qu'il a à faire.

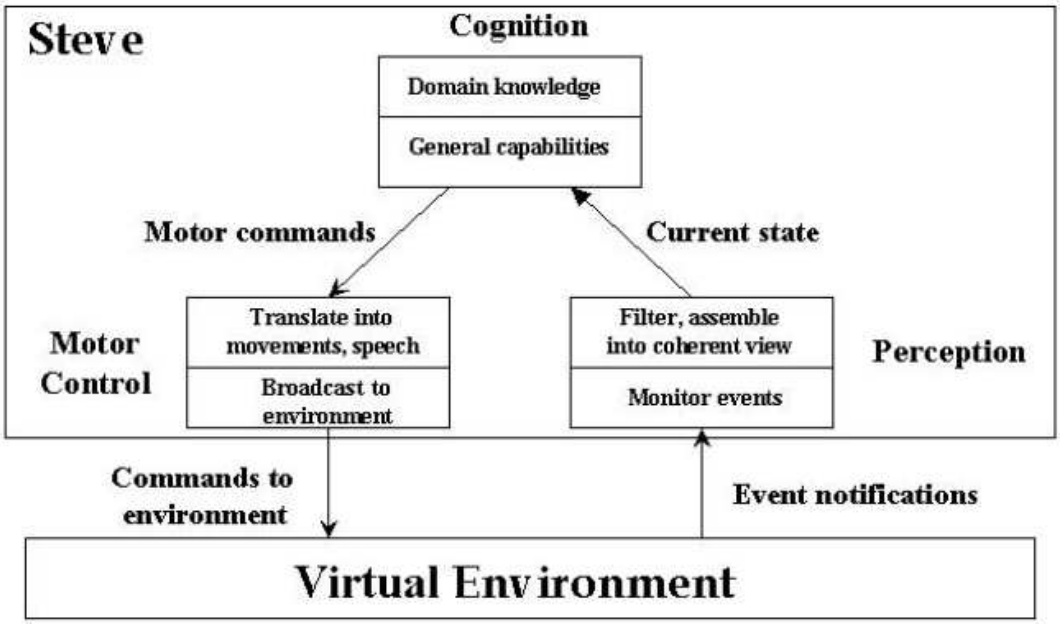


FIG. 1.15 – Comportement d'un agent Steve

Dans SECUREVI les agents peuvent avoir trois types de comportements :

– Réactif : lorsque que l'agent prend part à des interactions avec les objets physiques de l'environnement. Par exemple la température d'un agent va augmenter s'il s'approche d'un feu.

- Collaboratif : l'agent doit réaliser les buts de son équipe et sélectionner des actions en fonction de l'évolution de la procédure.
- Organisationnel : lorsque l'agent ne sait pas résoudre seul un problème, il fait appel à ce type de comportement en se référant à son supérieur hiérarchique.

L'agent est divisé en une partie décisionnelle et une partie opérative (voir figure 1.16). Le raisonnement de l'agent est représenté par son comportement collaboratif et son comportement organisationnel. Le comportement collaboratif est fondé sur un module de sélection d'actions utilisant la base de faits de l'agent. Ce mécanisme de sélection d'actions est lui même divisé en deux parties : une partie suivi de la procédure et une partie calcul de plans. L'agent commence par consulter la première action à exécuter dans la procédure et regarde si c'est à lui de la réaliser (il n'y a pas d'ambiguïté car une action n'est attribuée qu'à un seul rôle). Si ce n'est pas le cas il se met en attente. Si c'est le cas il regarde les préconditions de l'action. Si elles sont vérifiées il demande l'exécution de l'action par la partie opérative de l'agent que nous allons décrire un peu plus tard. Si les préconditions ne sont pas vérifiées, il va calculer un plan implicite pour tenter de les vérifier. Rappelons que dans SECUREVI une distinction est faite entre les actions métiers qui apparaissent dans la procédure et les actions génériques qui n'y figurent pas. Le calcul de plan se fait sur les actions génériques, par chaînage arrière sur les pré et post conditions de ces actions. Si un plan est trouvé la première action est exécutée, sinon l'agent invoque son comportement organisationnel et demande de l'aide à son supérieur hiérarchique. La partie opérative de l'agent est composée de trois modules s'exécutant en parallèle :

- Perception : acquiert la connaissance sur l'état du monde.
- Communication : ce module envoie les messages prévus par la procédure. Il gère également l'envoi et la réception des messages de début et de fin d'action qui sont émis par chaque agent pour informer les autres des actions en cours et faire en sorte qu'ils aient tous la même connaissance sur l'évolution de l'état de la procédure.
- Exécution des actions : réalise effectivement les actions choisies par le module décisionnel.

Dans l'EVFC de Dugdale, il n'y a pas d'humain virtuel puisque tous les intervenants du scénario sont joués par des utilisateurs. En revanche chaque utilisateur pilote un avatar doté de certaines capacités d'action : des actions non intentionnelles (comme les gestes accompagnant un dialogue ou la façon de marcher). Ces comportements sont modifiés par la perception de la situation de l'avatar, son état émotionnel, sa personnalité et son humeur. Le comportement de l'avatar est géré par quatre modules (voir l'architecture sur la figure 1.17) : perception, émotion, comportement et action. Le module de perception détecte les événements dans l'environnement. Le module d'émotion évalue l'importance de l'événement perçu en fonction de la personnalité et de l'humeur de l'avatar et décide de l'émotion ressentie. Le module de comportement évalue la contrepartie émotive des différentes réactions possibles. Enfin le module d'action exécute l'action sélectionnée par le module de comportement.

Chaque EVFC a ses propres mécanismes pour gérer le comportement de l'humain virtuel (ou de l'avatar) et en particulier la phase de prise de décision peut être influencée par divers facteurs (les émotions, le scénario). La modélisation des humains virtuels dans les EVFC peut, de plus, avoir des objectifs très différents. Certains s'orientent vers le réalisme des mouvements, d'autres vers la gestion des émotions, le langage naturel ou encore vers les capacités pédagogiques. Pourtant, peu d'EVFC s'intéressent à fournir un comportement paramétrable pour les

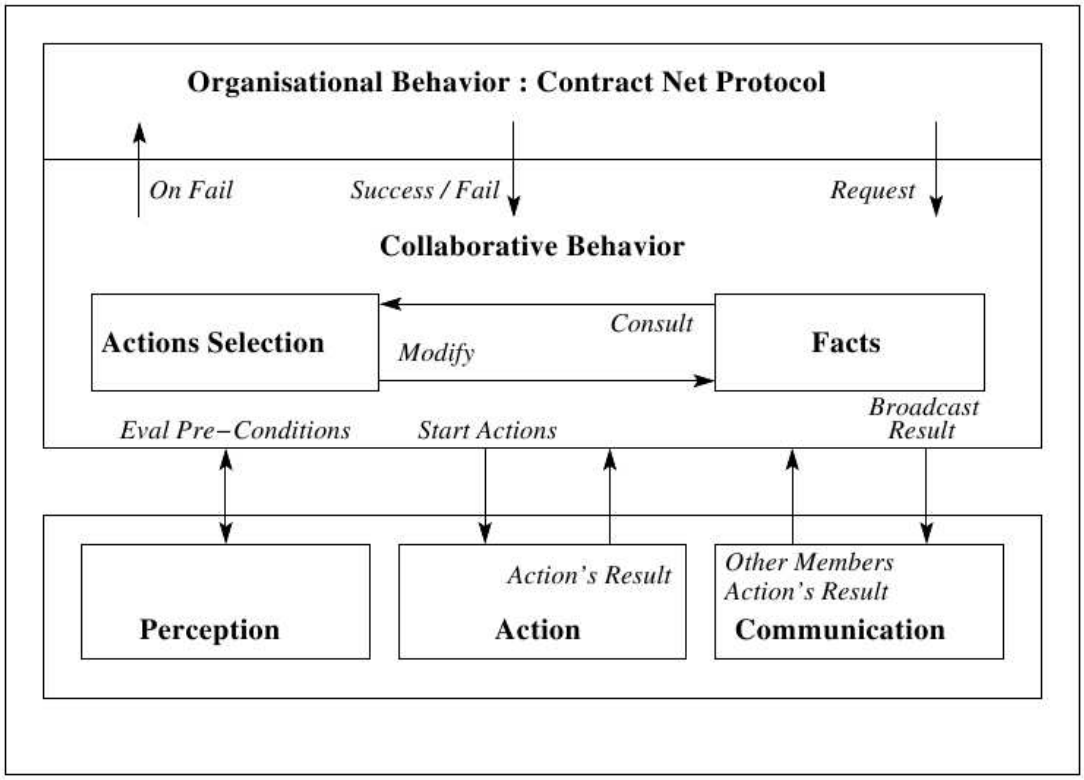


FIG. 1.16 – Comportement d'un agent rationnel autonome dans SECUREVI

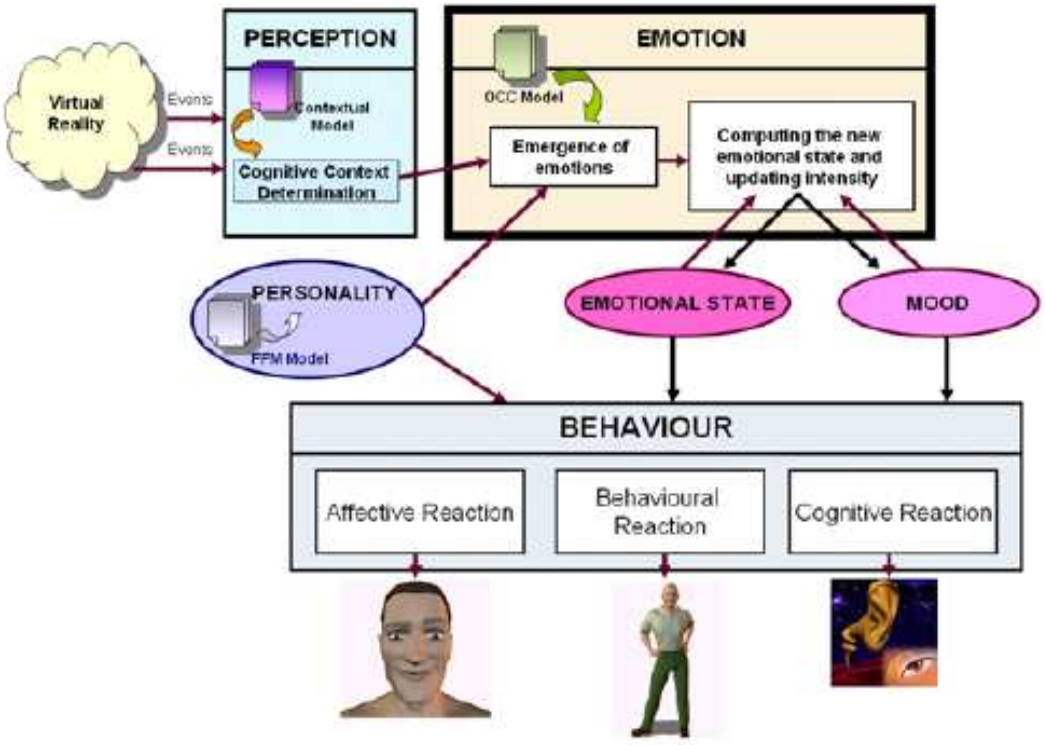


FIG. 1.17 – Comportement d'un avatar, d'après [DPPeJ04]

humains virtuels et la notion de rôle pédagogique est rarement exploitée.

1.4 Synthèse

De l'analyse de ces EVFC, nous déduisons un certain nombre de fonctionnalités qui sont importantes pour un EVFC : la conscience des autres et de leurs actions, la description d'un scénario collaboratif avec la notion de rôle et la possibilité d'avoir des humains virtuels autonomes. Néanmoins nous avons aussi identifié des manques parmi les EVFC existants : la répartition des actions de la procédure entre les acteurs est fixe et les humains virtuels ont rarement un comportement paramétrable. Le tableau 1.1 synthétise les caractéristiques des 5 EVFC analysés. Dans notre travail de recherche, nous nous sommes donc fixés les objectifs suivants : disposer d'un scénario flexible et adaptable au contexte mais qui oblige à respecter strictement la procédure de référence, réviser la notion de rôle afin de la rendre plus souple, permettre la création d'humains virtuels ayant chacun un comportement particulier et capables de venir en aide aux apprenants, pouvoir facilement interchanger un utilisateur réel et un humain virtuel et enfin, permettre différentes formes de collaboration quelque soit la nature des collaborateurs (utilisateurs réels ou humains virtuels). Tous ces points vont être discutés dans les chapitres suivants.

	COVET	EVFC de Dugdale	MRE	SECUREVI	version collaborative de Steve
Collaboration	apprentissage UR/UR niveau 1	scénario UR/UR niveau 2 - niveau 1 exploité	scénario UR/HV HV/HV niveau 2 - niveau 1 exploité	apprentissage, scénario et actions UR/UR UR/HV HV/HV niveau 3 - niveau 2 exploité	apprentissage et scénario UR/UR UR/HV HV/HV niveau 2
Scénario	Pas représenté	Pas représenté	à la manière d'un film (histoire plus que procédure) 2 rôles/action : exécution + ordre d'exécution	procédure d'intervention description partielle : actions métier (graphe ou contraintes) - actions génériques implicites 1 rôle/action. rôle = devoir (actions métiers) + savoir (actions génériques)	procédure de maintenance causalité (plans) 1 rôle/action. rôle = devoir (tâches individuelles)
Humains virtuels	non	avatars intelligents : émotions et gestes non intentionnels	agents Steve + extension (communication, langage naturel)	3 niveaux : réactif, collaboratif, organisationnel	agents Steve + extension (tâches collaboratives) 2 rôles pédagogiques : tuteur + substitut d'un coéquipier manquant

TAB. 1.1 – Synthèse de l'analyse des 5 EVFC

Chapitre 2

Contexte et objectifs

Cette thèse s'inscrit dans le prolongement de la thèse de Nicolas Mollet [Mol05]. Il y propose deux modèles : le modèle STORM pour décrire des objets dotés de comportements ainsi que les interactions entre ces objets et le modèle LORA pour décrire un scénario mono-utilisateur. Cette thèse a abouti à une première version d'un environnement de formation à la procédure, GVT (Generic Virtual Training). Nous proposons d'étendre ces modèles pour représenter l'activité d'un acteur (utilisateur réel ou humain virtuel) et pour réaliser des scénarios collaboratifs. Afin de mieux comprendre les enjeux du travail présenté dans ce mémoire, il est donc important d'avoir une vision globale de GVT dans sa version 1.0 qui permet d'apprendre individuellement (ou avec l'aide d'un formateur) une procédure individuelle.

2.1 Présentation de GVT 1.0 : l'existant

2.1.1 Le projet GVT

La formation est un besoin fort pour les industries, que ce soit la formation technique telle que l'utilisation ou l'entretien de nouveaux équipements ou la formation plus relationnelle pour l'acquisition de compétences en gestion d'équipe par exemple. L'utilisation de la réalité virtuelle pour la formation industrielle est connue pour offrir de multiples atouts [Lou01, Man01, Sto01] : elle permet d'éviter les risques à la fois pour les hommes et pour le matériel, de réduire les coûts (liés à l'immobilisation du matériel, aux éventuels dommages, au déplacement de tous les intervenants) et de s'affranchir de la disponibilité du matériel voire de son existence réelle. Par ailleurs, elle facilite la supervision des apprenants et autorise des actions pédagogiques impossibles à réaliser dans le monde réel. Partant de ce constat, le projet GVT est né. GVT est un projet qui a débuté en 2001 et qui rassemble deux partenaires académiques, l'INRIA et l'ENIB et un partenaire industriel, Nexter Systems. Son objectif est de développer une plateforme permettant de créer facilement des sessions de formation en environnement virtuel. La formation visée est relative à la procédure (par exemple une procédure de maintenance) et non pas aux gestes techniques, considérés comme déjà acquis. Par exemple, s'il s'agit de dévisser plusieurs boulons, nous allons nous intéresser à la séquence d'actions à effectuer (l'ordre de dévissage des boulons) et non pas au geste de dévissage qui est supposé maîtrisé.

FIG. 2.1 – GVT en utilisation immersive. (c) CNRS Photothèque - Hubert RAGUET

L'architecture de GVT a été conçue pour répondre à cette attente, la figure 2.2 montre les principaux composants et les liens qui existent entre eux. Cette architecture globale repose sur sur deux modèles principaux que nous allons décrire maintenant : un modèle d'objets comportementaux et d'interactions, STORM, et un modèle de scénario, LORA. Nous décrirons ensuite comment ils sont intégrés dans l'architecture globale de la plateforme. L'architecture de GVT ainsi que ces modèles sont décrits dans le mémoire de thèse de Nicolas Mollet [Mol05].

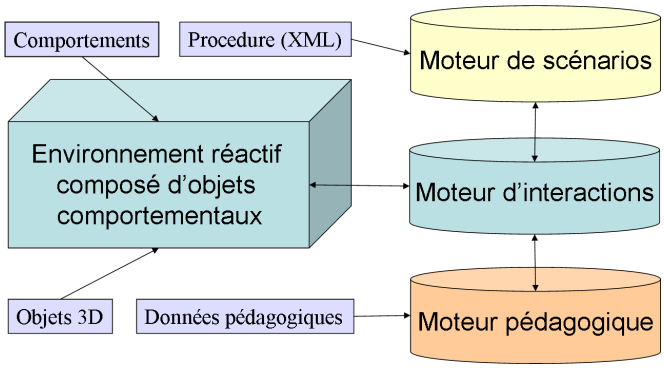


FIG. 2.2 – Vision globale du noyau de GVT

2.1.2 Modèles

Dans cette section nous allons donner un aperçu des concepts clés de chacun de ces modèles.

2.1.2.1 STORM

Le modèle STORM (Simulation and Training Object-Relation Model)[MGA07b] a été conçu dans le but d'améliorer la réutilisabilité des objets et des interactions entre ces objets. Ce modèle est défini en deux parties, un modèle d'objets comportementaux et un modèle d'interactions.

– **Un modèle d'objets comportementaux.** Un objet STORM est composé d'activités internes et est doté d'un ensemble de capacités (voir figure 2.3). Une capacité représente une possibilité d'interaction qu'un objet peut offrir. Elle est composée de deux éléments : une activité publique (le comportement et la réaction de l'objet) et une interface associée (un protocole standard de communication qui autorise l'objet à communiquer avec d'autres objets STORM). Même si la plupart des activités de nos objets sont modélisées en utilisant HPTS++ (Hierarchical Parallel Transitions System ++)[LD02], un langage d'automates hiérarchiques, STORM n'impose pas le mode d'implémentation d'un comportement. Pour créer un nouvel objet STORM, nous lui associons simplement différentes capacités d'interaction. Par exemple une vis a la capacité vis-mâle avec des caractéristiques particulières telles que le diamètre de la vis, le pas du filetage, etc.

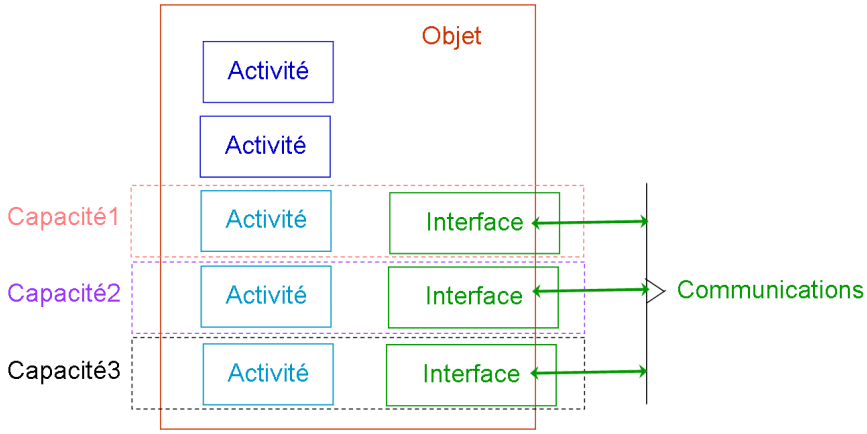


FIG. 2.3 – Modèle d'objet STORM

– **un modèle d'interactions.** La relation STORM représente le lien qui existe quand des objets entrent en interaction. La relation est elle-même un objet STORM ce qui permet de gagner en généricité puisqu'une relation peut alors gérer à la fois d'autres relations et des objets. La relation STORM contient toutes les informations requises pour définir une interaction. Une relation utilise les capacités des objets pour créer une interaction entre eux. Par exemple une relation de visserie va utiliser à la fois la capacité vis-mâle d'un boulon libre et la capacité vis-femelle d'un écrou libre pour établir un lien de visserie entre ces deux objets. L'action visser sur ces deux objets sera alors possible pour un utilisateur s'il possède une clé avec la capacité visseur (voir figure 2.4). Une relation STORM est décrite par un automate ; l'évolution de cet automate va faire transiter d'autres automates représentant le comportement des objets impliqués dans la relation. Une relation STORM peut ainsi être vue comme un engrenage : lorsque la roue de la

relation tourne (lorsque l'automate transite), elle va entraîner avec elle d'autres roues (les automates représentant le comportement des objets) ce qui va faire évoluer l'état des objets impliqués dans cette relation (voir figure 2.6).

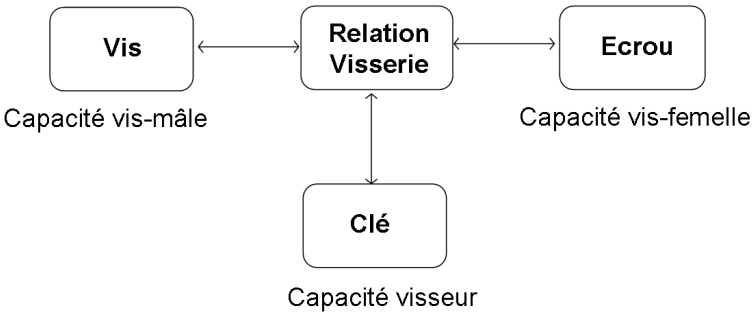


FIG. 2.4 – Exemple d'une relation STORM : la relation visserie

2.1.2.2 LORA

Dans notre contexte industriel, les procédures, et plus particulièrement les procédures de maintenance, sont très strictes (les actions doivent être faites exactement dans l'ordre spécifié), longues (plusieurs centaines d'actions) et complexes (la figure 2.5 montre un extrait d'une procédure réelle). Le langage LORA (Language for Object Relation Application) [MA06] a été conçu pour rendre accessible à des non informaticiens la retranscription de ces procédures de leur version originale (carte de travail par exemple, voir annexe A) dans une version interprétable par GVT. Les auteurs peuvent ainsi décrire complètement la procédure (telle qu'elle est décrite dans les cartes de travail), c'est à dire l'enchaînement des actions, sans nécessairement rendre explicite la causalité grâce à des préconditions et à des buts. LORA est un langage de description de scénarios qui possède à la fois une représentation graphique et une représentation textuelle. Sa philosophie est de décrire ce qui peut être fait à chaque étape du scénario. Il permet de décrire des enchaînements, parfois complexes (séquentiels, parallèles, à choix multiples), d'interactions. Il a été inspiré par différents langages graphiques comme le Grafcet et par différents langages d'automates parallèles et hiérarchiques. Un scénario écrit en LORA comprend un automate principal et peut contenir un ensemble de sous-automates. Chaque automate est composé d'un ensemble de variables (ce qui permet la création d'automates fonctionnels), d'étapes (associées à une action suivant le modèle STORM ou contenant un sous-automate) et de liens entres les étapes. Un automate a un état courant : un ensemble d'étapes actives qui représentent ce qui pourrait être fait à un instant donné. Une étape est associée à une transition dans une relation STORM et les liens sont déclenchés par l'activation de cette transition. La figure 2.6 illustre ce fonctionnement : les états courants sont en rouge et les transitions provoquées par une interaction sont en vert.

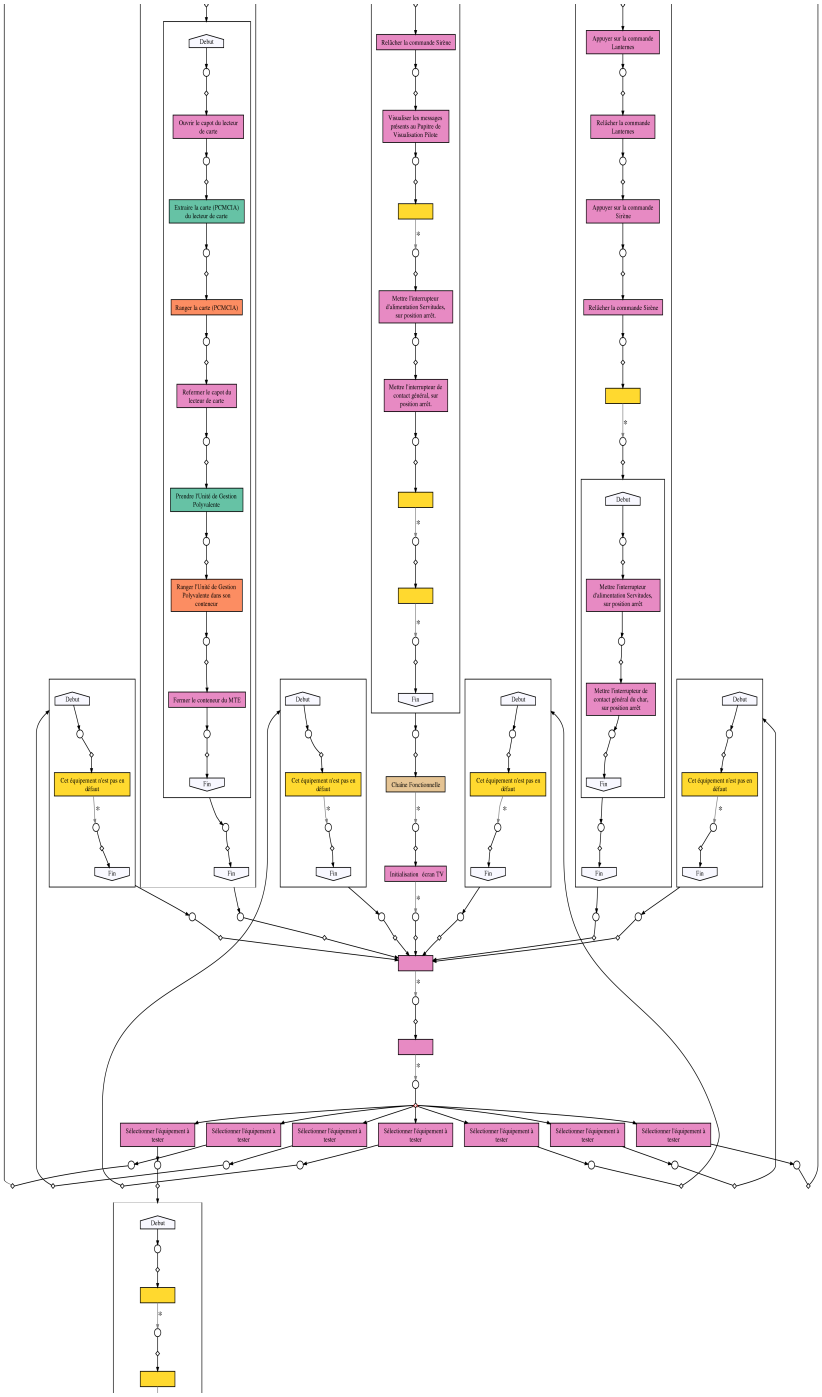


FIG. 2.5 – Extrait d'un scénario de diagnostic de GVT

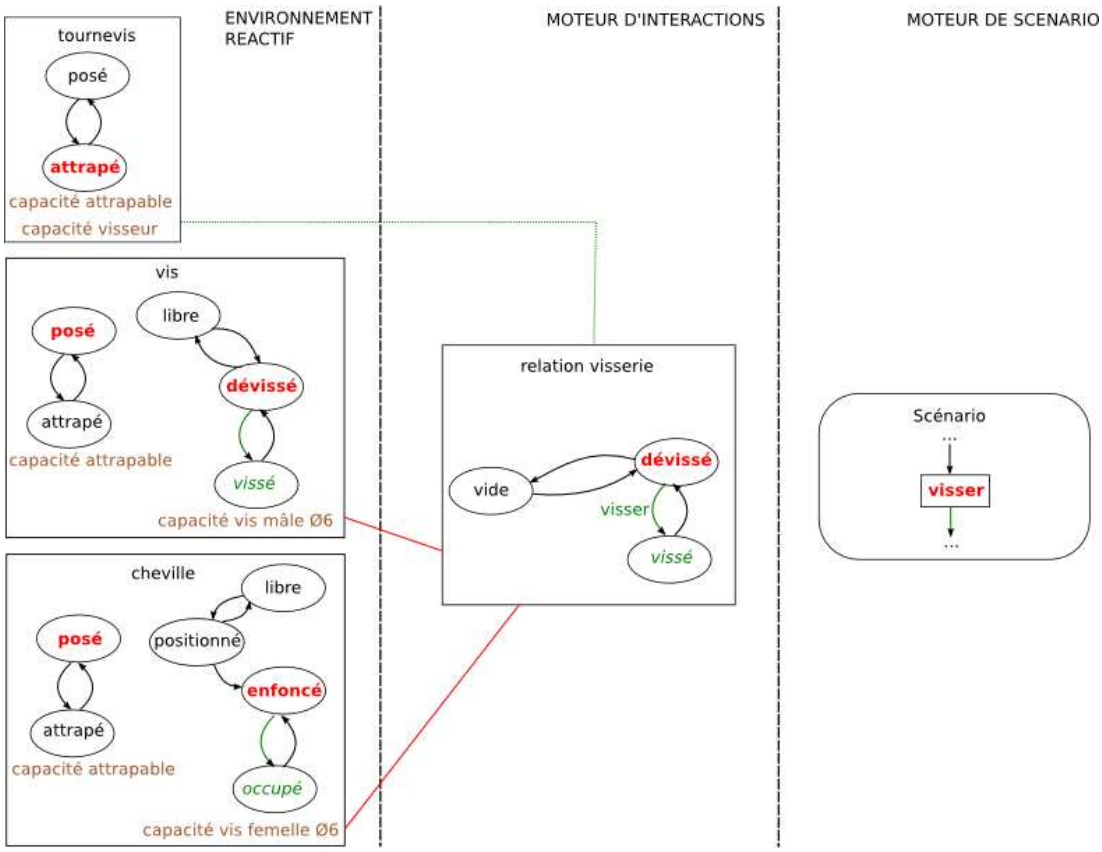


FIG. 2.6 – Evolution des automates lors d'une action

2.1.3 Architecture de GVT

Le noyau de GVT est constitué de quatre éléments principaux (voir figure 2.2), qui s'appuient sur les modèles STORM et LORA que nous venons de décrire brièvement. Cette architecture a été décrite dans [GMG+08] et [MGA07a].

- **Un environnement réactif composé d'objets comportementaux**. Comme nous l'avons vu, le modèle STORM permet de créer des objets dotés de comportements et de capacités d'interaction, pouvant ainsi interagir les uns avec les autres. Par exemple une vis est un objet comportemental et a la capacité vissable avec comme attributs sa nature (vis mâle), son pas de vis et son diamètre. L'ensemble des objets STORM forme alors un environnement réactif.
- **Un moteur d'interactions**. Son rôle est de gérer les interactions complexes qui peuvent intervenir entre les objets STORM. C'est donc ce moteur qui gère toutes les relations STORM. Le moteur d'interactions peut alors dire, à tout moment, quelles sont les relations qui peuvent évoluer et donc quelles sont les actions possibles sur les objets de l'environnement, en fonction de leurs états et de leurs capacités d'interaction. Prenons comme exemple une relation de visserie qui peut relier une vis, une cheville et un tournevis si ces objets ont des capacités compatibles. Un tournevis, ayant la capacité visseur,

pourra alors effectuer l'action visser sur une vis non vissée, positionnée sur une cheville, si la vis et la cheville ont des capacités compatibles (vis mâle et vis femelle, même diamètre).

– **Un moteur de scénario**. Le scénario est la transcription en langage LORA d'une procédure de référence (détaillée par exemple sur un manuel technique provenant d'un industriel). Le moteur de scénario se charge de dérouler ce scénario et peut à tout moment dire ce que l'apprenant peut faire dans la procédure, quelles sont les possibilités d'interaction qui correspondent aux prochaines étapes de la procédure. L'état du moteur évolue donc au fur et à mesure que l'apprenant réalise des actions dans l'environnement. Par exemple à un instant donné il se peut que l'apprenant ait le choix entre dévisser la vis 1 ou dévisser la vis 2. S'il choisit de dévisser la vis 1 alors cette action sera validée et la prochaine action active (la prochaine action de la procédure autorisée) sera alors de dévisser la vis 2.
– **Un moteur pédagogique**. Il utilise les deux moteurs précédents pour décider ce que l'apprenant est autorisé à faire. Il adapte ses réactions à l'apprenant, en fonction de son niveau et de la stratégie pédagogique qui lui a été définie ; on parle alors de pédagogie différenciée. Si un apprenant demande de l'aide, le moteur pédagogique va pouvoir l'aider de manière graduelle, il peut par exemple rappeler la procédure, afficher un document pédagogique (pdf, vidéo, etc), mettre en évidence l'objet avec lequel il faut interagir et il peut même aller jusqu'à se substituer à l'apprenant en demandant au moteur d'interaction la réalisation de l'action. Cette aide va également dépendre du niveau pédagogique de l'apprenant : pour un apprenant novice l'aide ira jusqu'à la réalisation de l'action alors que pour un apprenant plus expérimenté le moteur pédagogique pourra se contenter d'énoncer l'action à réaliser. Ce moteur est utilisé pour assister le formateur. De plus, toutes les actions et demandes d'aide de l'apprenant sont archivées pour quantifier son évolution (voir figure 2.7).

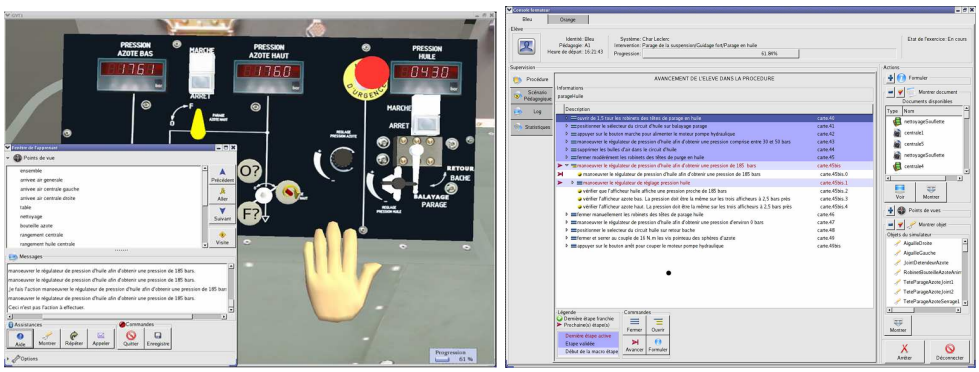


FIG. 2.7 – Interfaces de GVT. A gauche : le poste apprenant, à droite : le poste formateur

2.1.4 Evaluation de GVT 1.0

Un étude a été menée [HGD07, GHD08] par notre partenaire, le CERV, afin d'évaluer l'efficacité de GVT pour apprendre une procédure individuelle. Cette étude s'appuyait sur le cadre de travail théorique défini par Anderson [And83, And95]. En effet, Anderson suppose que l'apprentissage de procédures s'effectue en trois phases. Il commence par la phase cognitive avec l'utilisation d'instructions sous forme déclarative pour guider l'acquisition d'un nouveau savoir-faire. Cette phase est très coûteuse cognitivement. Durant la phase suivante, la phase associative, l'apprenant effectue une transition entre une utilisation lente et délibérée du savoir vers une représentation plus directe de ce qu'il faut faire (règles de production de la forme si... alors ...). Les erreurs sont progressivement détectées et éliminées. Une fois que la procédure a été répétée un certain nombre de fois elle devient de plus en plus automatique et rapide. Dans la troisième phase, la phase autonome, la procédure est acquise ce qui se traduit par une exécution rapide et précise.

12 participants devaient réaliser une procédure de 25 étapes, au cours de 10 essais consécutifs. Cette procédure était extraite d'une procédure industrielle réelle et concernait la maintenance d'un char Leclerc. Les participants disposaient pour cela d'instructions uniquement textuelles. Après une pause d'une semaine, les participants sont revenus pour 3 nouveaux essais. Différents critères de performance ont été mesurés comme le temps de lecture des instructions, le temps d'exécution des actions et le nombre d'erreurs. Cette campagne d'évaluation a permis de valider l'apprentisage d'une procédure individuelle dans GVT par rapport à la théorie d'Anderson. En effet, durant la première phase le temps passé à l'exécution de la procédure était très long (à cause de la charge cognitive et du recours systématique aux instructions) et a amené beaucoup d'erreurs. Le temps de lecture des instructions, d'exécution des actions ainsi que les erreurs a ensuite diminué dans la seconde phase jusqu'à atteindre les performances maximales dans la troisième phase. Les résultats de cette étude (voir figure 2.8) montrent donc une courbe d'apprentissage qui indique que la procédure semble acquise. GVT permet d'apprendre efficacement une procédure puisque même après une semaine les participants retrouvent très vite des niveaux de performances équivalents à ceux obtenus à la fin de la première campagne d'essais. La procédure semble donc mémorisée dans la mémoire à long terme.

Il reste maintenant à mener une étude afin d'évaluer s'il y a transfert entre la procédure acquise en virtuel et la même procédure à réaliser en réel.

2.1.5 Synthèse

La première version de GVT permet de former à des procédures individuelles. Or l'analyse du besoin industriel nous a permis de dégager un besoin de formation pour des procédures collaboratives. Nous avons donc décidé d'élargir le spectre des procédures pouvant être apprises grâce à GVT en permettant l'apprentissage de procédures collaboratives [GMA07]. La version 1.0 de GVT est à la fois opérationnelle et validée, nous pouvons donc nous appuyer dessus pour étendre ses fonctionnalités et offrir maintenant une formation collaborative. Nous nous sommes donc appuyés sur l'architecture existante de GVT que nous avons fait évoluer pour prendre en compte des scénarios collaboratifs dans lesquels des utilisateurs réels peuvent collaborer entre eux ainsi qu'avec des humains virtuels.

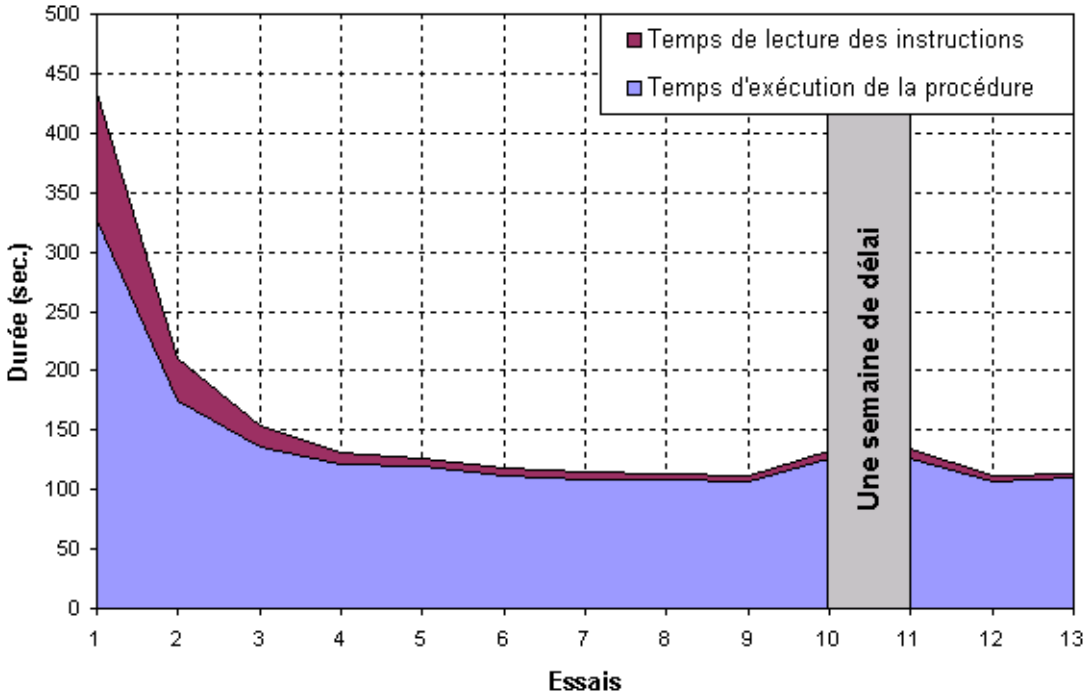


FIG. 2.8 – Mesures de performances

2.2 Objectifs et contraintes

L'objectif de cette thèse était de faire évoluer GVT 1.0 suivant deux axes : l'introduction de la collaboration et celle des humains virtuels, afin d'aboutir à un prototype : GVT 2.0. Comme nous l'avons vu dans l'état de l'art, la collaboration peut intervenir à plusieurs niveaux et notre but était d'offrir une palette la plus large possible des possibilités de collaboration. Concernant les humains virtuels, nous souhaitions pouvoir facilement les interchanger avec des utilisateurs réels et nous voulions aussi pouvoir leur donner un comportement paramétrable afin de pouvoir les utiliser pour remplacer un équipier manquant tout en modulant leurs comportements à des fins pédagogiques.

Mais il fallait satisfaire ces objectifs en respectant un certain nombre de contraintes industrielles. La première contrainte était de partir de l'existant, de s'appuyer sur les modèles existants de GVT pour les faire évoluer. Les grands principes de GVT ne devaient donc pas être remis en cause : la procédure comme élément central et l'homogénéité de description des objets comportementaux grâce au modèle STORM. Une autre contrainte consistait à modéliser de manière similaire l'activité dans le monde virtuel des humains virtuels et celle des utilisateurs, dans le but de pouvoir facilement interchanger utilisateurs réels et humains virtuels. Il fallait également pouvoir paramétrer le comportement des humains virtuels afin de pouvoir leur faire jouer des rôles pédagogiques différents. Enfin, nous voulions qu'au lieu de complexifier l'écriture du scénario, l'introduction de la collaboration aille de pair avec une simplification de celui-ci, tout en le rendant également plus flexible.

2.3 Supports théoriques

Une fois les objectifs posés et les contraintes clairement identifiées, nous avons cherché des solutions pour les remplir tout en nous référant à la litérature pour y puiser des ancrages théoriques solides dès que cela était possible.

Nous avons décidé de nous appuyer sur la théorie des rôles de Biddle et Thomas [BT66] pour décrire l'assignation des acteurs aux actions du scénario. Nous sommes également parti de la thèse de Daniel Mellet d'Huart [Md04] et de sa boucle *anticipation, décision, mise en œuvre de l'action* pour décrire l'activité des acteurs. Nous nous sommes aussi inspirés des rôles pédagogiques abordés par Cédric Buche dans les perspectives de sa thèse [Buc05] pour paramétrer le comportement des humains virtuels. Enfin nous avons choisi d'utiliser le concept de modèle mental partagé [CBSC90] pour représenter la connaissance des agents de la procédure.

2.4 Aperçu des contributions

En nous appuyant sur les modèles que nous avons évoqués dans la section précédente nous allons maintenant donner un aperçu des contributions qui vont être présentées dans la suite de ce manuscrit. Ces contributions s'orientent suivant deux axes : le scénario et l'activité des acteurs.

2.4.1 Scénario

Un de nos objectifs était de poposer une extension au langage de scénarisation LORA pour pouvoir représenter de manière à la fois simple et réaliste des procédures collaboratives. En effet, LORA ayant été conçu pour écrire un scénario individuel, la notion d'acteur y était absente. Pour décrire une procédure collaborative, nous avons vu dans notre état de l'art que la notion de rôle est utilisée. Mais le rôle dans les EVFC est souvent figé : un seul rôle est assigné à une action, et même s'il est parfois possible pour un utilisateur de changer de rôle, il n'y a alors aucun contrôle pour vérifier si ce changement est opportun. La théorie des rôles [BT66] souligne le fait que, dans un travail collectif, l'affectation d'un rôle à une tâche n'est pas toujours effectuée de manière statique et figée mais qu'elle peut être réalisée de manière dynamique en fonction du contexte. Par exemple, prenons le cas d'une procédure de demande de congés. La demande doit être signée de préférence par le chef d'équipe mais, s'il n'est pas disponible, l'assistante est elle aussi autorisée à signer cette demande (cas précis dans l'équipe-projet Bunraku). Dans ce cas, il serait peu judicieux de considérer que l'assistante peut prendre temporairement le rôle du chef d'équipe (en effet cela risque de ne pas plaire à Stéphane), puisqu'elle n'aura pas pour autant les mêmes droits d'action que le chef d'équipe, mis à part pour cette action précise. L'action de signer le document peut donc bien être assignée à deux rôles différents. Le langage de description du travail en équipe RoB-MALLET [Zha07] permet cette assignation dynamique mais il semble être le seul. Parmi les EVFC trouvés dans la littérature, aucun n'aborde cette question de l'assignation dynamique d'un rôle pour une action. Pourtant, comme le soulignent Biddle et Thomas [BT66], la pré-association des rôles aux actions est trop restrictive et ne rend pas compte de la réalité du travail en équipe. En revanche, permettre une assignation dynamique nous oblige à mettre en place un mécanisme capable de déterminer dynamiquement

quel est l'acteur le plus apte à réaliser une action en fonction de critères eux aussi évalués dynamiquement.

Pour permettre une plus grande flexibilité des scénarios, nous proposons l'assignation dynamique des rôles aux actions, de rendre implicite certaines actions basiques et très communes dans la procédure comme la prise et la pose d'outils, d'introduire des pré-conditions sur l'état des mains d'un acteur ainsi que sur l'état d'une relation STORM. Mais la simplification du scénario implique des modules capables de raisonner au dessus du scénario. Ce raisonnement est effectué dans le mécanisme de répartition des actions et par l'intermédiaire d'un module de gestion de ressources propre à chaque acteur.

Nous avons vu dans l'état de l'art que la collaboration peut apparaître à différents niveaux dans un EVFC. Nous souhaitons pouvoir proposer une palette aussi vaste que possible des types de collaboration. Ainsi nous devons réfléchir à la répartition des tâches au sein du scénario (procédure multi-utilisateur), mais également à des actions collaboratives, à réaliser à plusieurs. Les théories sur le travail en équipe suggèrent que les membres de l'équipe doivent exprimer des intentions communes avant de réaliser des tâches collaboratives. Afin de rendre ces actions réalisables de la même manière par un humain virtuel ou un utilisateur réel nous avons décidé de les faire figurer en tant qu'actions dans le scénario ; nous nommons ces actions déclarations d'intention de collaborer.

2.4.2 Activité des acteurs

Modèle d'acteur : la représentation d'acteur Afin de satisfaire notre objectif concernant l'interchangeabilité entre les utilisateurs réels et les humains virtuels, nous avons décidé de proposer un modèle commun permettant de représenter l'activité d'un acteur (qu'il soit réel ou virtuel) de manière homogène. Nous avons donc choisi d'avoir une représentation unique de l'entité pouvant agir dans l'environnement virtuel, que cette entité soit pilotée par un utilisateur réel ou par un humain virtuel. Cette entité doit interagir avec le monde virtuel et sera donc un objet STORM que nous appellerons représentation d'acteur car c'est la représentation à travers laquelle humains virtuels et utilisateurs vont pouvoir agir ; il est possible de faire une analogie entre cette entité et une marionnette qui sera alors dirigée soit par un utilisateur soit par un humain virtuel. Afin d'accroître les possibilités d'actions nous souhaitions également permettre à un acteur d'interagir dans le monde virtuel grâce à ses deux mains. Pour cela on introduira la notion de ressource afin de modéliser les éléments qui permettent à un acteur d'agir (les mains, la vue, etc). Une représentation d'acteur sera donc un nouvel objet STORM capable de gérer des ressources qui seront, elles, des objets STORM classiques. De plus, de nouvelles actions qui vont rendre une ressource occupée et donc indisponible vont être proposées.

Activité Afin de décrire l'activité d'un acteur, nous nous sommes appuyés sur la thèse de Daniel Mellet d'Huart [Md04] qui propose de modéliser l'activité d'un humain au sein d'un environnement virtuel pour l'apprentissage grâce à une boucle *anticipation, décision, mise en œuvre de l'action*. Nous avons adapté cette boucle afin de la rendre plus représentative de ce qui peut se passer lors d'un travail en équipe. Nous décrivons ici succinctement le déroulement de cette boucle, mais sans nous restreindre au seul point de vue de l'acteur, nous voyons cette boucle dans une vision plus globale. La partie anticipation a été scindée en deux parties : une

partie globale qui représente l'anticipation commune sur les actions de tous les membres de l'équipe, et une partie locale qui représente l'anticipation individuelle de chacun sur sa propre action. La phase de décision d'un acteur, suivie par la phase de mise en œuvre de l'action va provoquer, pour les autres acteurs qui n'ont pas été actifs, un rebouclage sur la phase d'anticipation (voir figure 2.9).

Concernant la partie anticipation, nous avons décidé de nous appuyer sur les théories ainsi que des études psychologiques indiquant que lors d'un travail d'équipe les coéquipiers possèdent des modèles mentaux partagés [CBSC90]. Dans le cadre d'une formation à la procédure ces modèles mentaux partagés sont représentés par l'état d'avancement de la procédure et par l'analyse des différentes actions possibles. Cette analyse peut être factorisée puisqu'elle est commune à tous les acteurs. Nous avons donc décidé de mettre en place un module global (appelé module de répartition des actions) chargé d'analyser les actions possibles du scénario et capable d'anticiper sur les possibilités d'attribution de ces actions aux différents acteurs. Ce module réalise donc une anticipation pour l'équipe. Nous avons donc un premier module d'anticipation globale, qui se charge de proposer une répartition des actions entre les acteurs, considérée comme la meilleure pour l'avancement de la procédure en fonction de certains critères.

Ensuite ce module global est complété par un module d'anticipation local à chaque acteur et qui lui permet, en fonction de son profil collaboratif, de choisir quel serait la prochaine action qu'il pourrait réaliser. Chaque acteur dispose donc d'un mécanisme de prise de décision qui se sert du résultat du module de répartition (une suggestion de répartition) et qui la transforme en action visée en fonction des propensions[1] de l'acteur . Ces propensions sont traduites dans le profil collaboratif de l'acteur. Cette anticipation est aussi effectuée pour un utilisateur réel car elle permettra de lui donner des conseils pédagogiques s'il est demandeur. Cette phase d'anticipation personnelle aboutit à une décision théorique. En effet à ce stade rien ne permet de savoir si un acteur donné sera le prochain à réaliser une action et pourra donc mettre en œuvre sa décision anticipée. Cette phase est donc à cheval entre la phase d'anticipation et la phase de décision.

La phase suivante, la phase de décision effective, peut être soit déclenchée par l'utilisateur qui va décider de faire une action dans l'environnement, soit par la pédagogie qui va demander à un humain virtuel de faire une action donnée, soit par un humain virtuel qui va décider qu'il est temps qu'il agisse. C'est la seule phase qui diffère entre un utilisateur réel et un humain virtuel. En effet l'utilisateur réel va choisir lui-même l'action à réaliser alors que l'humain virtuel va traduire l'action qu'il vise en une action directement réalisable (qui peut être par exemple une action prérequise par l'action visée). Suite à cette décision, le scénario et la pédagogie vont entrer en jeu : le scénario va signaler si l'action choisie fait partie ou non de la procédure, et en fonction de cela, la pédagogie va dire si elle autorise cette action ou non. La phase de décision est donc elle aussi répartie entre plusieurs composants de l'application.

Cette décision va déclencher la phase suivante du cycle : la phase de mise en œuvre de l'action. La phase de mise en œuvre de l'action est en revanche commune aux humains virtuels et aux utilisateurs réels puisqu'elle est constituée d'une demande de réalisation de l'action

[1] Une propension désigne une tendance (par exemple la propension à collaborer) ; les propensions d'un acteur permettent donc de définir son comportement

transmise au moteur d'interaction. Le moteur d'interaction va alors réaliser l'action en faisant évoluer les relations nécessaires, les animations correspondantes vont être déclenchées, l'environnement virtuel va lui aussi évoluer. Il sera donc possible pour tous de visualiser les effets de l'action choisie sur le monde virtuel. Le moteur de scénario va alors lui aussi prendre en compte la réalisation de l'action, en évoluant si l'action réalisée faisait partie de la procédure et va alors fournir le nouvel ensemble d'étapes actives (qui correspondent aux nouvelles actions possibles). Le cycle peut alors reboucler sur l'anticipation globale quant à la répartition de ces actions entre les acteurs.

En résumé la boucle anticipation, décision, mise en œuvre de l'action n'est pas toujours complète et peut être décomposée en une phase d'anticipation globale, une phase d'anticipation de décision, une phase de décision effective individuelle, une phase de confirmation de cette décision et une phase de mise en œuvre de l'action (voir figure 2.9). Un seul acteur à chaque cycle réalise la boucle complète (l'acteur 2 sur la figure 2.9), les autres (l'acteur 1 sur la figure 2.9) se contentent alors d'effectuer les phases d'anticipation globale et d'anticipation de décision. Nous avons donc généralisé le modèle de Daniel Mellet d'Huart pour la réalisation collaborative d'une procédure.

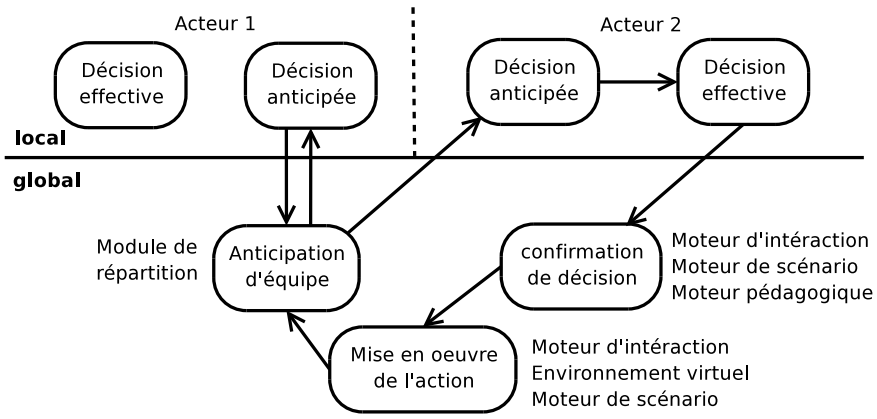


FIG. 2.9 – Boucle d'activité des acteurs et du mécanisme global

Paramétrage du comportement Chou et al. [CCL03] proposent une classification des différents agents pédagogiques pouvant intervenir dans un environnement de formation. Cédric Buche [Buc05] y fait référence et envisage, dans les perspectives de sa thèse, d'intégrer différents rôles pédagogiques pour les humains virtuels dans son système : compagnon, assistant, tuteur, gêneur. Nous avons décidé d'offrir des humains virtuels avec un comportement paramétrable afin de pouvoir proposer de tels rôles pédagogiques. Cependant nous ne voulions pas nous contenter de ces rôles pédagogiques et nous voulions pouvoir décrire plus finement le comportement d'un acteur et en particulier ses propensions à collaborer et à suivre la procédure. Pour cela nous avons donc défini la notion de profil collaboratif. Le profil collaboratif d'un acteur va le guider dans sa phase de décision et lui permettre de choisir la prochaine action à réaliser.

2.5 Définitions

Dans ce manuscrit nous employons des termes d'usage commun mais qui peuvent s'avérer ambigus dans notre contexte. Nous allons donc poser maintenant un certain nombre de définitions des concepts clés que nous allons utiliser dans ce mémoire de thèse.

Utilisateur réel Un utilisateur réel est un être humain qui utilise l'EVFC. La plupart des utilisateurs sont alors des apprenants, mais le formateur est lui aussi un utilisateur potentiel.

Humain virtuel Un humain virtuel est une entité virtuelle qui a pour but d'agir dans l'EVFC comme pourrait le faire un être humain.

Acteur Un acteur représente une entité agissante dans l'EVFC. Cette entité peut être soit un utilisateur réel soit un humain virtuel.

Représentation d'acteur La représentation d'acteur est l'objet STORM à travers l'acteur va pouvoir interagir. C'est donc la modélisation de l'acteur au sein de l'EVFC qui gère l'activité de l'acteur. Cette entité a un certain nombre de caractéristiques (rôle, ressources, capacités) et possède également des capacités de raisonnement (gestion de ressources, module décisionnel).

Avatar L'avatar est la représentation visuelle d'un acteur, son incarnation dans l'environnement.

Activité L'activité représente l'ensemble des actes effectués par une personne. Il peut s'agir d'actes concrets (des actions dans l'environnement) ou d'actes abstraits (des raisonnements par exemple).

Procédure La procédure représente la suite d'actions nominales spécifiée par l'industriel pour réaliser une opération donnée (par exemple une opération de maintenance). La procédure représente l'agencement de référence des actions à réaliser.

Scénario Le scénario représente la suite d'instructions chargées au minimum de représenter la procédure. Dans un EVFC le scénario représente la modélisation de la procédure et peut donc contenir des informations différentes (actions implicites, causalité, actions interdites, etc).

Rôle Le rôle permet de lier un scénario à des acteurs ; il représente les actions que l'acteur qui va le jouer sait, peut ou doit faire. Le savoir est associé au rôle lui-même (capacités). Les possibilités et devoirs viennent du scénario (responsabilités).

Collaboration La collaboration représente le fait d'agir à plusieurs dans le but de réaliser une tâche commune. Cette tâche peut être concrête (une procédure collaborative) ou abstraite (apprentissage collaboratif). Nous considérons les termes collaboration et coopération comme synonymes.

Deuxième partie

Contributions

Chapitre 3

Activité des acteurs

Dans un environnement virtuel de formation, deux types d'entités différentes existent et peuvent agir : les utilisateurs (apprenants ou formateurs) et les humains virtuels avec un degré d'autonomie variable. Ces deux entités vont piloter une représentation d'acteur à travers laquelle ils vont agir. Le terme générique d'acteur représente donc l'association d'une entité capable de prendre des décisions (un utilisateur réel ou un humain virtuel) et d'une représentation d'acteur. Un acteur possède une représentation 3D au sein de l'environnement virtuel et une partie comportementale qui va lui permettre d'agir dans cet environnement. C'est essentiellement au niveau de cette seconde partie que se situent nos contributions. Nous voulons modéliser l'activité des acteurs en suivant deux objectifs : premièrement nous voulons pouvoir interchanger facilement utilisateurs réels et humains virtuels autonomes, deuxièmement nous souhaitons pouvoir créer des humains virtuels avec des comportements différents, qui pourront donc être paramétrés. Pour cela, notre première contribution est un ensemble de modèles permettant de représenter de manière homogène l'activité des deux différents types d'acteurs et permettant de personnaliser le comportement des humains virtuels.

L'activité d'un acteur comprend plusieurs aspects. Un acteur est tout d'abord un objet comportemental de l'environnement qui peut interagir avec d'autres objets, c'est pourquoi un acteur est modélisé comme un objet STORM (nommé représentation d'acteur) avec des capacités d'interaction (voir section 3.1). Afin d'étendre les possibilités d'interaction de base d'un acteur on ajoute la notion de ressource interne. Une représentation d'acteur est donc composée de ressources internes, qui sont elles aussi des objets STORM, sur lesquelles elle va pouvoir raisonner (état courant, états atteignables, séquences d'actions pour atteindre un état donné) ; ce raisonnement sera intégré dans ce que l'on nomme une activité interne dans le modèle STORM. Un acteur fait aussi partie d'une équipe et il doit pouvoir différencier son activité de celle de ses coéquipiers, pour cela il joue un rôle qui va lui amener de nouvelles capacités d'interactions et également des responsabilités vis à vis du scénario (voir section 3.2). Enfin, l'acteur doit être capable de choisir une action à réaliser dans l'environnement, ce mécanisme sera lui aussi intégré en tant qu'activité interne (voir section 3.3). Nous proposons donc un ensemble de modèles pour gérer toutes ces facettes de l'activité d'un acteur. La figure 3.1 illustre la position de l'acteur entre les différents modules existants, ainsi que ses différentes caractéristiques et les mécanismes de raisonnement (activités internes) dont il dispose.

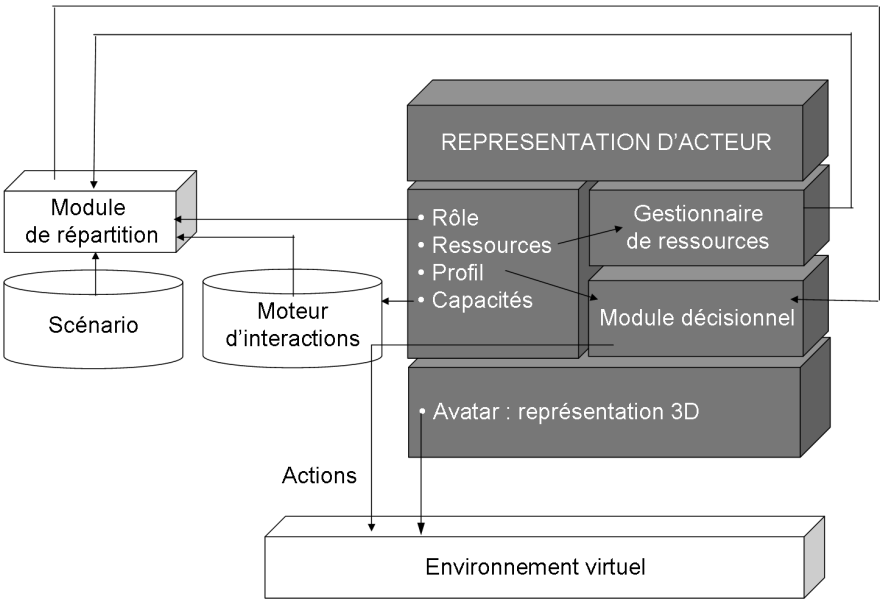


FIG. 3.1 – L'acteur et ses interactions avec les différents modules

3.1 Possibilités d'interaction

Le modèle que nous utilisons dans GVT pour décrire les objets de l'environnement est le modèle STORM que nous avons présenté dans le chapitre précédent. Un acteur est un objet comportemental de l'environnement et est donc modélisé en tant qu'objet STORM ce qui lui permet d'interagir avec les autres objets du monde, y compris avec d'autres acteurs. L'une de nos contraintes était en effet de ne pas particulariser un acteur et de le traiter comme les autres objets comportementaux de l'environnement, afin qu'il puisse bénéficier du mécanisme général d'interaction basé sur le modèle STORM. Comme tout objet STORM, un acteur doit donc être doté de capacités. C'est en effet grâce à ses capacités qu'un acteur pourra participer à des relations et donc entrer en interaction avec d'autres objets. Dans la version mono-utilisateur de GVT, l'utilisateur était représenté par une main (la main constituait l'avatar de l'utilisateur). L'utilisateur ne pouvait interagir que grâce à cette main et ses capacités d'interaction étaient donc limitées à celles de sa main. Nous avons décidé d'enrichir le modèle de l'acteur afin d'étendre ses possibilités d'interactions.

3.1.1 Ressources

Une ressource est une entité susceptible d'être utilisée pour exécuter une tâche et souvent disponible en quantité limitée. Lorsque l'on s'intéresse au comportement d'un humanoïde, pour faire de la planification de tâches par exemple, les ressources considérées sont alors souvent les parties du corps de cet humanoïde (exemple [Nar98][LD02]).

Nous souhaitions offrir la possibilité à un acteur d'interagir non plus uniquement par le biais d'une seule main, mais également grâce à d'autres parties de son corps. Nous voulions

pour cela proposer un moyen générique de modéliser certaines parties du corps d'un acteur pouvant intervenir dans une interaction ; nous avons donc proposé d'introduire la notion de ressource dans nos modèles. Il fallait que ces parties corporelles puissent être dans un état qui les empêche temporairement d'interagir et nous voulions également mettre en place un mécanisme capable de trouver un chemin pour passer d'un état de l'acteur à un autre grâce à des actions de base. Enfin, lorsqu'une action ne s'adressait pas à une de ses ressources en particulier, nous voulions permettre à l'acteur d'interagir avec son corps dans sa globalité.

3.1.1.1 Description des ressources

Pour nous, une ressource est une partie du corps d'un acteur dont le comportement peut être modélisé par un automate et ayant au moins deux états différents. STORM permet de hiérarchiser les objets, grâce à la notion de relation. Une relation est un objet STORM qui relie et gère d'autres objets STORM. Un acteur est alors représenté par une relation insécable entre différentes ressources internes qui seront chacune un objet STORM (cette relation ne pourra être brisée, les ressources d'un acteur seront toujours liées par cette relation de haut niveau). Chaque ressource a donc des capacités propres, qui vont ainsi permettre d'augmenter les possibilités d'interaction de l'acteur.

Dans un premier temps, nous nous sommes focalisés sur les deux mains de l'acteur que nous avons considérées comme ressources. Il sera possible par la suite d'ajouter des ressources complémentaires, telles que la vue (pour mettre en place une relation d'observation par exemple) ou les jambes (pour gérer le déplacement). La ressource main apporte par exemple à l'acteur les capacités d'attraper et de tenir. Chaque main a trois états possibles (cf figure 3.2) :

- **libre**. C'est l'état de repos de la main, lorsqu'elle est vide. La relation STORM appelée relationSaisie ne relie la main à aucun objet.
- **occupée**. Cet état n'existait pas dans la version 1 de GVT. Il indique que la main est engagée dans une relation STORM de type relationOccupante qui la monopolise et la relie à un objet occupant. Cet engagement est obligatoirement dû à une action figurant dans le scénario. Une main *occupée* ne peut plus servir à l'acteur pour interagir. Pour sortir de cet état il faut passer par une action spécifique complémentaire de l'action qui a rendu la main occupée qui va ainsi libérer la main. L'action permettant de désengager la main doit elle aussi figurer dans le scénario, l'acteur n'est donc pas libre de la faire quand il le souhaite.
- **objet attrapé**. Cet état fait généralement suite à une action *prendre* et indique que l'acteur possède un objet dans sa main. L'acteur peut alors utiliser cet objet comme un outil et par conséquent s'en servir pour interagir avec un autre objet. Une relationSaisie relie alors l'objet attrapé à la main et cet objet va ainsi suivre les déplacements de l'acteur. Pour sortir de cet état et repasser à l'état *libre*, l'acteur doit simplement effectuer une action *poser*, possible à tout moment. Un acteur peut en effet effectuer les actions *prendre* et *poser* quand il le souhaite : ces actions n'ont pas besoin de figurer dans le scénario. Cet état est en fait une particularisation de l'état *occupée* ; dans l'état *objet attrapé* l'objet impliqué n'est pas inactif mais permet au contraire à l'acteur qui le possède d'interagir. Par exemple s'il faut retirer une plaque fixée à la verticale par des vis, il va falloir la maintenir avec une main pour ne pas qu'elle tombe lorsque les vis seront retirées. L'ac-

tion *tenir la plaque* (qui figure dans le scénario) rend la main *occupée*, elle devient alors inutilisable tant que l'action *relâcher la plaque* (qui figure aussi dans le scénario et qui va remettre la main dans l'état *libre*) n'aura pas été réalisée ; alors que l'action *prendre le tournevis* (qui ne figure pas dans le scénario) met la main dans l'état *objet attrapé* mais l'acteur peut se servir du tournevis pour interagir et il peut à tout moment effectuer l'action *poser le tournevis* pour libérer sa main.

Pour bien comprendre la différence entre l'état *objet attrapé* et l'état *occupée*, prenons un autre exemple, celui du clou. Le clou est en effet un objet pouvant à la fois mettre la main dans l'état *objet attrapé* et dans l'état *occupée* en fonction du contexte. Lorsque l'acteur prend le clou dans sa boite à outils sa main passe alors dans l'état *objet attrapé*, l'acteur possède le clou et l'emmène avec lui lors de ses déplacements. Il peut prendre et poser le clou comme bon lui semble car ces actions n'ont pas besoin de figurer dans le scénario. Il va ensuite poser le clou sur la surface à clouer et prendre un marteau. Pour enfoncer le clou dans la surface (pour réaliser l'action *clouer* figurant dans le scénario) l'acteur va donc d'une main tenir le clou (en réalisant l'action *tenir le clou* qui figure elle aussi dans le scénario) qu'il avait préalablement positionné et de l'autre il va se servir du marteau. La main qui tient le clou est alors dans l'état *occupée*, cette main ne peut en effet plus interagir directement avec un autre objet, elle ne peut pas non plus interagir par l'intermédiaire du clou. L'acteur a donc une main monopolisée tant qu'il n'aura pas fini de planter le clou.

Nous pouvons également utiliser cet état *occupée* pour simuler une main blessée (donc inutilisable par l'acteur) ou une main dont l'acteur ne peut plus se servir (par exemple car elle tient un objet dont l'acteur n'est pas autorisé à se séparer).

Les états *libre* et *occupée* sont des états standards pour une ressource. L'état *libre* indique que la ressource est disponible pour interagir, l'état *occupée* indique qu'elle est déjà impliquée dans une relation et que les autres interactions ne sont alors plus possibles. L'état *objet attrapé* en revanche est propre à la main.

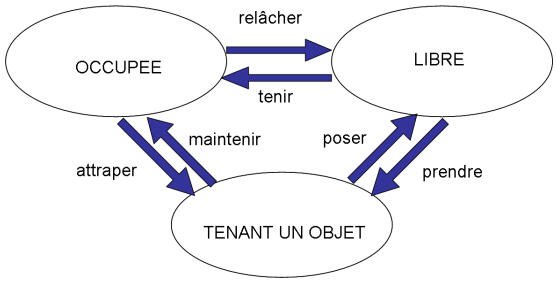


FIG. 3.2 – Les différents états possibles pour une main (les actions permettant de passer d'un état à un autre sont juste données à titre indicatif)

3.1.1.2 Mécanisme de gestion de ressources

Nous venons de voir qu'un acteur est composé de ressources internes. Chaque ressource dispose d'un état. L'acteur possède donc un état courant qui est l'agrégation de l'état de ses

ressources et de son état global. Ces états vont également être exploités dans le scénario par exemple en tant que pré-requis pour une action (nous y reviendrons dans le chapitre suivant). Nous dotons donc l'acteur d'une activité interne lui permettant de raisonner sur ses ressources. Ce raisonnement est effectué par un mécanisme de gestion de ressources. Partant de l'état courant des ressources de l'acteur, ce module est capable de déterminer si un état donné (un état figurant comme pré-condition d'une action du scénario par exemple) est atteignable (grâce à des actions de base) et, le cas échéant, il peut fournir une séquence d'actions à réaliser pour y arriver. Les actions de base qui constituent cette séquence calculée sont appelées actions implicites (puisqu'elles ne figurent pas dans le scénario), nous y reviendrons dans la section 4.5.

Actuellement, le mécanisme de gestion des ressources raisonne uniquement sur l'état des mains de l'acteur : l'état courant sera donc un couple représentant l'état de la main droite et celui de la main gauche. L'état visé sera un couple non ordonné d'états à choisir parmi {*libre*, *tenant un objet*, *occupée*, *indifférent*}. Les états *tenant un objet* et *occupée* sont accompagnés d'une information respectivement sur l'objet attrapé et sur la cause de l'occupation. La séquence d'actions obtenue est constituée d'actions de prise et de pose d'objets que l'acteur devra réaliser pour atteindre l'état visé. Mais les solutions possibles sont rarement uniques (tant en terme d'actions que de séquencement de ces actions) ; le module de gestion des ressources fournit simplement une possibilité parmi celles existantes. Par exemple si l'acteur a une main vide et un tournevis dans l'autre main et qu'il vise l'action *planter un clou* qui a comme pré-requis de posséder un marteau et un clou, la séquence d'actions proposée pourra être la suivante : *poser le tournevis, prendre un marteau, prendre un clou*. Cette séquence n'est pas la seule possible, mais ce qui est important c'est l'existence d'un chemin et la possibilité de disposer de toutes les étapes sur ce chemin.

Nous avons choisi de considérer l'acteur comme parfaitement ambidextre et par conséquent de ne pas différencier dans l'état visé la main droite de la main gauche ou plutôt la main dominante de la main dominée pour prendre en compte à la fois les droitiers et les gauchers. Ce choix augmente un peu la complexité du mécanisme de gestion de ressources qui ne peut pas se contenter de faire une comparaison pour chacune des mains entre l'état actuel et l'état voulu (il y a plus de combinaisons possibles) mais en contrepartie la spécification du scénario est simplifiée. En effet, cela évite pour commencer d'avoir à spécifier la main dominante de chaque acteur. Ensuite, cela évite de devoir préciser, pour chaque action, si la main utilisée va être la main dominante, la main dominée, ou indifféremment l'une ou l'autre. Lorsque l'acteur veut réaliser une action prendre, cela évite qu'il ait à préciser quelle main il souhaite utiliser. Toutes ces informations supplémentaires sont en effet fastidieuses à préciser et ne nous paraissent pas essentielles dans notre contexte de formation. Cela évite également de proposer une action de transfert d'un objet d'une main à l'autre (cette action ferait alors partie des actions pouvant constituer la séquence d'actions fournie par le module de gestion des ressources). Néanmoins, il est possible de spécifier certaines actions ne pouvant être réalisées que par la main dominante par exemple : pour cela il suffit d'ajouter aux capacités requises par l'action une capacité *main dominante* que l'on peut rajouter à la main dominante de l'acteur. Toutefois, si pour d'autres applications nous nous rendons compte que cette distinction entre la main dominante et l'autre main est importante, il suffira de modifier le mécanisme de gestion des ressources : son raisonnement sera en effet simplifié puisqu'il suffira de comparer l'état requis pour la main

dominante avec l'état actuel de la main dominante, et de même pour l'autre main. Une nouvelle action apparaîtra alors dans le calcul de plan qui sera l'action de transfert d'un objet d'une main à l'autre. Il faudra alors changer la spécification dans le scénario des états pré-requis pour une action en considérant un couple ordonné (main dominante puis main dominée par exemple) avec éventuellement plusieurs possibilités (si l'action peut être réalisée indifféremment avec l'une ou l'autre des mains). La gestion ambidextre de l'acteur revient à considérer les mains comme une ressource à deux places. La gestion par main dominante revient à particulariser chacune des deux main en considérant la main dominante comme une ressource à une place et la main dominée comme une ressource à une place.

<table>
<tr><th>Etat cible
Etat départ</th><th>Libre</th><th colspan="2">Objet attrapé</th><th colspan="2">Occupée</th><th>Indifférent</th></tr>
<tr><td>Libre</td><td>Direct</td><td colspan="2">1 action de prise</td><td colspan="2">Impossible</td><td>Direct</td></tr>
<tr><td rowspan="2">Objet attrapé</td><td rowspan="2">1 action de pose</td><td>même objet</td><td>objets différents</td><td colspan="2" rowspan="2">Impossible</td><td rowspan="2">Direct</td></tr>
<tr><td>Direct</td><td>1 action de pose
+ 1 action de prise</td></tr>
<tr><td rowspan="2">Occupée</td><td rowspan="2">Impossible</td><td colspan="2" rowspan="2">Impossible</td><td>Même cause</td><td>Causes différentes</td><td rowspan="2">Direct</td></tr>
<tr><td>Direct</td><td>Impossible</td></tr>
</table>

FIG. 3.3 – La compatibilité entre un état de départ et un état cible grâce aux actions de base uniquement

Nous allons maintenant détailler l'algorithme utilisé pour obtenir une séquence d'actions permettant de passer d'un état des mains de départ à un état cible. En entrée il prend un couple représentant l'état actuel des deux mains et un couple représentant l'état visé pour les deux mains. Son objectif est d'associer chaque état courant à un état cible, en mettant à jour le nombre d'actions de prise et/ou de pose nécessaires. Les différentes associations possibles sont : état actuel de la main droite avec premier état cible, état actuel de la main droite avec deuxième état cible, état actuel de la main gauche avec premier état cible, état actuel de la main gauche avec deuxième état cible. Le tableau de la figure 3.3 illustre la compatibilité entre les différents états, ainsi que les actions de base requises pour passer de l'un à l'autre. Lorsque "impossible" figure dans une case de ce tableau, cela signifie qu'il faut impérativement passer par une action spécifique, figurant dans le scénario, pour pouvoir passer de l'état de départ à l'état cible ; la transition n'est pas possible en utilisant seulement les actions de base. En sortie de l'algorithme nous obtenons un booléen indiquant si l'état visé est atteignable, le nombre de prise d'outils requis, le nombre de pose d'objets requis ainsi que l'objet impliqué dans la première action de la séquence calculée. Nous considérons que si plusieurs actions intermédiaires sont requises, la pose sera la première à effectuer (et donc l'objet à considérer sera celui qu'il faudra reposer). L'algorithme est détaillé sur la figure 3.4. L'étape 2, l'association directe, consiste à rechercher les associations grâce à une compatibilité directe : (*libre*,*libre*), (*occupée*,*occupée*) avec les mêmes causes ou (*objet attrapé*,*objet attrapé*) avec le même objet (ou

un objet ayant le bon type[1]). A la sortie de cette étape, s'il reste des états cibles *occupée* qui n'ont pas été associés (cela signifie qu'il n'y a pas d'état *occupée* comme état de départ ou que la cause de l'occupation est différente), alors on sait qu'il n'y a pas de séquence d'actions de base possible pour arriver à l'état cible en partant de l'état courant (voir le tableau d'association sur la figure 3.3). Ensuite on passe en revue les différentes associations nécessitant des actions implicites, on commence par l'association *libre* et *objet* qui nécessite une actions de prise, puis l'association *objet* et *libre* qui nécessite une action de pose et enfin l'association *objet* et *objet* qui nécessite une action de pose suivie d'une action de prise. A chaque fois on met à jour le compteur d'actions de prise et de pose et la variable indiquant le premier objet à considérer dans la séquence résultat. L'ordre de considération fait que cet objet correspondra toujours à une action de pose s'il y en a une. A la fin de l'algorithme, s'il reste des états de départ non associés et qu'il n'y a pas assez d'*indifférent* pour les associer, alors l'algorithme renvoie faux pour indiquer qu'il n'y a pas de séquence d'actions possibles, sinon il renvoie vrai et renvoie les diverses informations détaillées précédemment.

La séquence d'actions obtenue n'est pas mémorisée entièrement car en réalité elle sera recalculée suite à une action dans l'environnement ; on effectue en effet une reconsidération dynamique du plan d'actions (voir chapitre 5). Seules les informations utiles au mécanisme de sélection d'action sont ainsi fournies : la faisabilité et le nombre d'actions de prise et de pose nécessaires. Le premier objet impliqué sera quant à lui utile pour l'acteur s'il sélectionne l'action associée ; en effet il saura ainsi quelle est la première action à effectuer (si le nombre d'actions de pose requis est supérieur à zéro il s'agira de poser cet objet, sinon s'il y a au moins une action de prise requise il s'agira de prendre cet objet, sinon cela signifie qu'il n'y a pas d'action implicite nécessaire). Les séquences sont toujours calculées en posant d'abord les objets nécessaires puis en prenant les objets manquants, afin de libérer au plus vite des objets dont l'acteur ne se sert pas. En revanche il n'y a pas de raison particulière sur le choix d'un objet à poser plutôt qu'un autre s'il y a plusieurs possibilités. Il serait envisageable d'utiliser des heuristiques pour déterminer l'objet qui a le moins de chance d'être réutilisé et qu'il faudrait donc reposer en priorité (en analysant les actions suivantes par exemple ou en mémorisant les outils qui ont déjà servi).

En résumé le mécanisme de gestion de ressources permet de savoir si un état donné est atteignable à partir d'un état de départ, et il permet également d'obtenir une séquence d'actions possible pour rejoindre l'état cible.

3.1.1.3 Que faut-il considérer en tant que ressource ?

Une ressource est une partie du corps permettant à l'acteur d'agir car elle peut participer à une relation STORM. Mais il n'est pas nécessaire de modéliser toutes les parties du corps avec lesquelles un acteur peut interagir en tant que ressource. Nous venons de voir que pour le moment seules les mains sont considérées en tant que ressources internes de l'acteur. Pourtant l'acteur dispose de bien d'autres parties du corps avec lesquelles il va pouvoir interagir (les yeux pour des actions d'observation, la bouche pour communiquer, les jambes pour se déplacer, etc). Il est possible de rajouter d'autres ressources dans le modèle d'acteur, mais il convient

[1] le nom associé à un état *objet attrapé* peut être soit le nom de l'instance d'un objet (exemple : clou1), soit le type de l'objet (exemple : clou). Les types sont détaillés dans la section 4.5.3

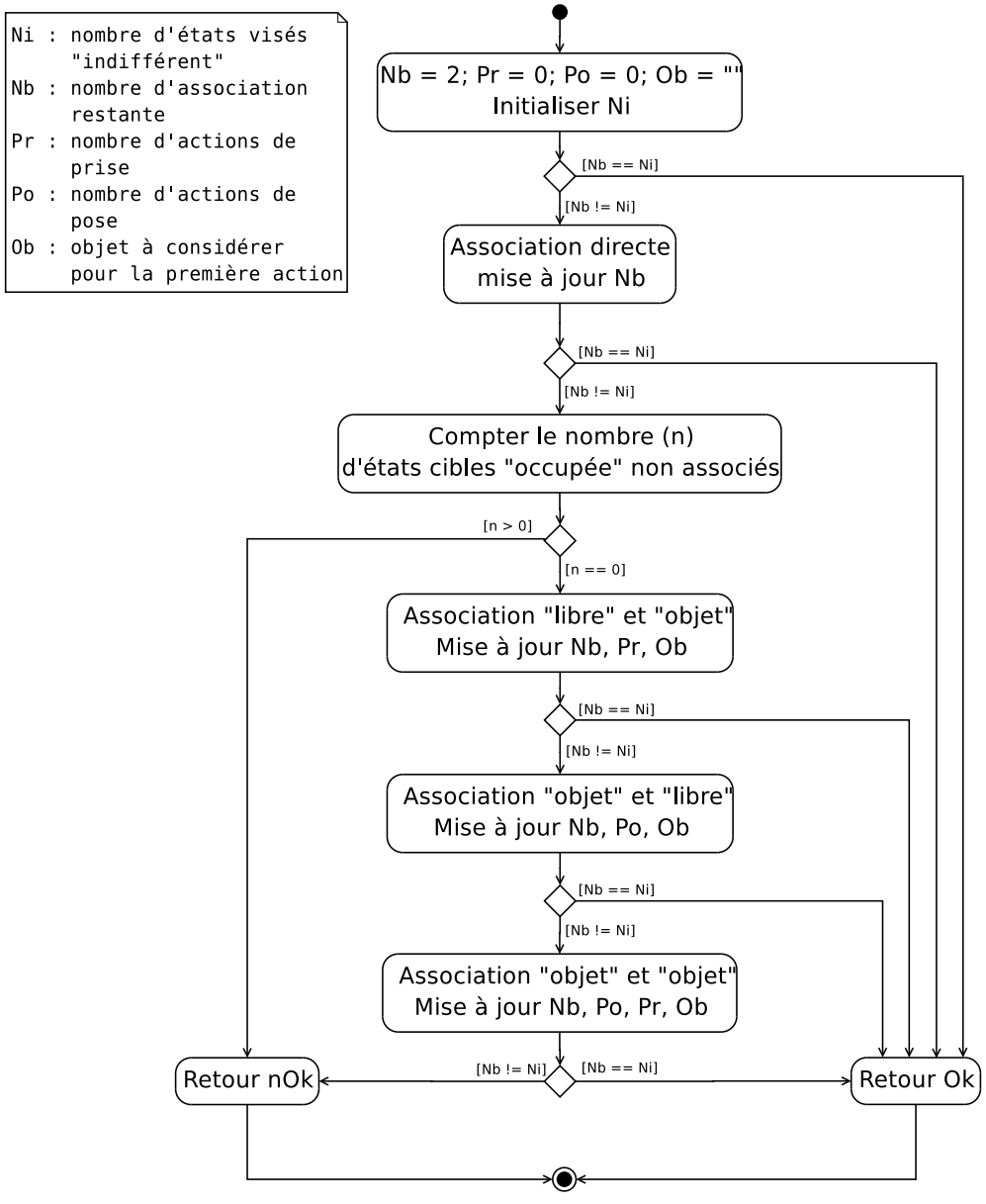


FIG. 3.4 – Algorithme du gestionnaire de ressources

d'évaluer au préalable la pertinence de le faire. En effet, une autre solution consiste à attribuer des capacités globales à l'acteur (les capacités que lui apportent cette partie du corps) plutôt que de créer une ressource interne ayant ses propres capacités.

Les deux moyens de prendre en compte une partie du corps dans notre modèle d'acteur présentent des similitudes concernant leur prise en compte par des relations STORM. Les parties du corps servant à l'acteur pour interagir possèdent des capacités. Par exemple les yeux donnent la capacité de voir. Ces capacités peuvent être associées directement à l'acteur, ce seront alors des capacités globales. Ces capacités peuvent aussi être associées à une ressource particulière et l'acteur ne les possède alors que par transition. Par exemple la capacité voir peut être associée soit à l'acteur soit à une ressource vue. Une relation d'observation va prendre en paramètre un objet STORM avec la capacité voir. Selon le mode choisi pour modéliser les yeux, cette relation impliquera soit l'acteur soit la ressource vue, mais la description de la relation restera la même. Dans les deux cas la relation d'observation pourra désactiver cette capacité tant que l'action arrêter d'observer n'aura pas été réalisée afin d'éviter que l'acteur ou la vue puisse intervenir dans une autre relation du même type (ou une autre relation nécessitant la capacité voir).

La gestion en tant que ressource permet d'avoir accès à des fonctionnalités de raisonnement. L'exploitation des ressources se fait principalement au niveau du scénario. Il est en effet possible de spécifier dans le scénario des préconditions sur l'état d'une ressource. Il est aussi possible de spécifier au module de gestion des ressources des actions de base pour changer l'état d'une ressource (actions implicites pour le scénario). Ce module va alors pouvoir fournir, si elle existe, une séquence d'actions pour amener une ressource dans un état voulu. Le module de gestion des ressources va donc pouvoir raisonner sur cette ressource.

Prenons comme exemple les jambes. Si l'on veut créer une relation de déplacement, cette relation va utiliser une ressource jambes. Une relation occupante pourra alors monopoliser cette ressource pour faire en sorte d'interdire les déplacements. Cette ressource aura donc deux états : *libre* et *occupée*. Par ailleurs on souhaite pouvoir raisonner sur cette ressource puisqu'on voudrait par exemple préciser en précondition d'une action que l'acteur doit se trouver près d'un certain objet pour pouvoir interagir avec. On peut alors préciser l'objet en question dans les préconditions de l'action. Le mécanisme de gestion des ressources va donc vérifier si l'acteur est actuellement proche de l'objet (suivant une heuristique à déterminer). Si c'est le cas alors la précondition sera directement vérifiée. Si ce n'est pas le cas, une action de déplacement sera possible si les jambes sont dans l'état *libre* et qu'il existe un chemin pour aller de la position actuelle de l'acteur à la position cible (ou s'il existe un point de vue proche de l'objet cible si la navigation est effectuée par points de vue). Dans ce cas les capacités globales ne permettent pas de gérer notre problématique et il faut considérer les jambes comme une ressource.

Au niveau de la description des relations, il n'y a pas de différence entre la modélisation d'une partie du corps en tant que ressource ou la modélisation grâce à des capacités globales. La différence entre ces deux modes de modélisation se fait au niveau du scénario : dans une gestion par ressource il va être possible de spécifier des préconditions sur ces ressources et l'acteur va pouvoir raisonner sur ses ressources, alors que pour une modélisation par capacité globale, il n'y a pas de raisonnement possible et, dans le scénario, les relations prendront en paramètre directement l'acteur au lieu d'une ressource particulière. Le choix se fait donc dans la description de l'acteur et la différence d'utilisation se voit au niveau du scénario.

3.1.2 Capacités globales

Nous venons de voir que l'utilisation des capacités globales peut être, pour certaines parties du corps, une alternative à la gestion en tant que ressource. Outre les capacités STORM apportées par ses ressources, un acteur dispose donc de capacités globales. Certaines sont héritées et donc communes à tous les acteurs (par exemple la capacité de communiquer), d'autres sont spécifiques à un acteur. Dans la section 3.2, nous verrons que le rôle joué par un acteur va par exemple lui amener de nouvelles capacités. Les capacités globales vont être utilisées par des relations qui vont impliquer l'acteur dans sa globalité au lieu d'une ressource particulière.

Attardons nous maintenant sur une capacité globale particulière, la capacité *mainsLibres*. Cette capacité est utilisée lorsque l'acteur veut réaliser une action qui nécessite ses deux mains. Cette capacité ne peut pas être attachée à une main en particulier, elle est donc rattachée à l'acteur dans sa globalité et fait partie des capacités communes à tous les acteurs. Cette capacité est désactivée dès qu'une main entre dans un état *objet* ou *occupée* et se réactive si les deux mains sont de nouveau dans l'état *libre*. Il est intéressant ici de gérer les deux mains conjointement puisqu'elles peuvent participer simultanément à la même relation. Les deux mains gérées conjointement pourraient être vues comme une nouvelle ressource mais il est plus simple ici d'ajouter une capacité globale pour représenter leur état et de se servir du mécanisme de raisonnement actuel pour chacune des mains (en précisant dans le scénario lorsque cela est nécessaire qu'il faut que chacune des deux mains soit libre). En s'appuyant sur cette capacité, on peut créer facilement des relations nécessitant qu'un acteur aient ses deux mains libres comme par exemple une relation porter à deux et à deux mains, porter à deux mains, pousser à deux mains, etc . Grâce à cette capacité globale ces actions ne seront proposées que si l'acteur a ses deux mains libres.

3.1.3 Synthèse

Pour conclure, notre modèle d'acteur est adaptable grâce à sa décomposition en ressources et peut être aisément augmenté en fonction des interactions que l'on souhaite offrir à l'acteur. Pour déterminer quelles sont les possibilités d'interaction pour un acteur à un instant donné, il faut regarder les possibilités d'interaction avec l'acteur dans sa globalité et les possibilité d'interaction avec chacune de ses ressources, y compris avec les objets liés à ces ressources (voir figure 3.5). Les possibilités d'interaction de l'acteur dépendent donc de ses capacités globales mais également des capacités de ses ressources ainsi que de leurs états. L'agrégation de l'état des ressources internes de l'acteur permet de déterminer un état global pour ce dernier. Notre modèle d'acteur profite ainsi des avantages du modèle STORM, de sorte que les acteurs peuvent interagir non seulement avec n'importe quel objet STORM mais aussi les uns avec les autres. Cette opportunité leur permet de communiquer ou de collaborer par exemple, peu importe s'il s'agit d'un humain virtuel ou d'un utilisateur réel.

La partie comportementale d'un acteur peut être divisée en deux parties. La première partie, réactive, est identique pour un humain virtuel et pour un utilisateur réel : il s'agit de gérer les possibilités d'interaction à tout moment avec l'acteur ; pour cela c'est le modèle STORM qui est utilisé et notamment la notion de capacité comme nous venons de le décrire. Il nous reste maintenant à décrire la deuxième partie lui permettant de raisonner et de faire des choix

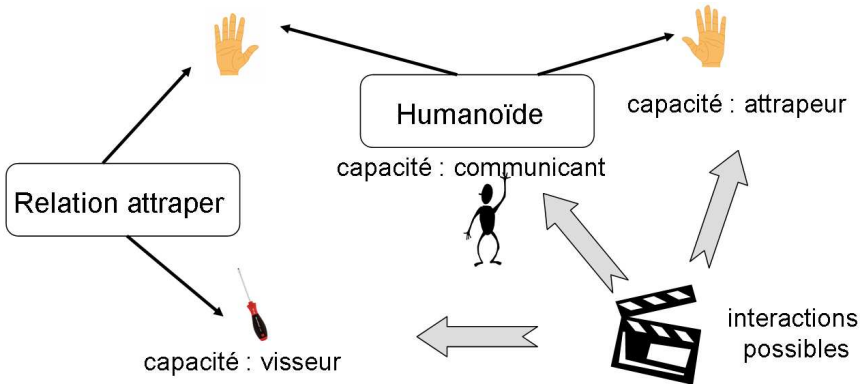


FIG. 3.5 – Interactions possibles avec un acteur

d'action. Cette activité interne d'un acteur consiste à prendre part au groupe, à anticiper sur les actions qu'il va pouvoir faire dans l'environnement et à en sélectionner une. Il s'appuie pour cela sur deux modèles, le premier décrit son appartenance à un groupe et permet de particulariser son activité au sein de ce groupe. Le deuxième modèle lui permet de choisir une action à réaliser. C'est à ce niveau que s'effectue la différenciation entre humains virtuels et utilisateurs réels. Ces deux modèles seront décrits dans les deux sections suivantes.

3.2 Responsabilités au sein du groupe

Comme nous l'avons vu dans notre état de l'art, les modèles organisationnels s'occupent de représenter les liens existant entre les différentes personnes devant réaliser un travail collectif. Nous n'allons pas redéfinir notre propre modèle, mais prendre une sous partie des modèles existants. Nous laissons de côté pour le moment la notion de groupe ou d'organisation. En effet les procédures que nous visons impliquent un nombre assez réduit de participants, avec des fonctions sociales relativement similaires et ces procédures présentent une certaine localité en terme de lieu et d'objets impliqués ; on peut alors considérer que tous les individus appartiennent au même groupe. Nous reprenons par contre la notion de rôle qui est l'élément central de tous ces modèles. Nous allons en revanche modifier quelque peu l'utilisation qui en est faite. Concernant la procédure, nous considérons que la connaissance de la procédure est partagée par tous et notamment que chacun peut suivre sa progression et voir quel est l'état actuel de chacun de ses partenaires. Les acteurs supposent pour leur raisonnement que tous les acteurs partagent la même volonté de faire avancer au mieux la procédure, même si dans les faits ce n'est pas forcément le cas (voir section 3.3).

3.2.1 Définition du rôle

L'environnement étant collaboratif, l'activité d'un acteur doit pouvoir être différenciée de celle de ses coéquipiers et ses responsabilités au sein du groupe clairement identifiées. Pour cela, l'acteur se verra attribuer un rôle. Premièrement, le rôle permet au scénario de s'abstraire

des individus qui vont effectivement le réaliser, tout en permettant de définir qui est autorisé à faire chaque action. Quand un acteur décide de jouer un rôle, il prend donc la responsabilité de faire les actions du scénario qui sont associées à ce rôle. Deuxièmement, le rôle apporte à l'acteur qui le joue des compétences supplémentaires nécessaires à la réalisation de la procédure ; ces compétences seront traduites par des capacités d'interactions supplémentaires pour l'humanoïde. Par exemple un chef aura la capacité de commander, un technicien celle de contrôler une jauge, etc. Le rôle va donc permettre à l'acteur d'agir. Ainsi un rôle est défini par un nom, qui va permettre de faire le lien avec les actions du scénario, et par un ensemble de capacités nécessaires pour jouer ce rôle et qu'il apporte donc à l'acteur qui va le jouer.

3.2.2 Attribution du rôle

Dans différentes architectures de travail en équipe, les agents ont l'équivalent de nos capacités au départ et ensuite le système vérifie qu'ils ont bien les capacités requises pour pouvoir jouer le rôle (ex : RoB-Mallet). Dans ces modèles, on décrit les individus en détail puis on voit quels rôles ils peuvent jouer. Nous avons choisi d'adopter la philosophie inverse que nous estimons mieux adaptée à un contexte de formation : au départ tous les individus ont les mêmes capacités de base ; s'ils décident d'apprendre un rôle on considère alors qu'ils ont de nouvelles capacités requises par ce rôle, leur but est alors d'apprendre à bien les utiliser. Cette vision des choses évite d'avoir à modéliser finement chaque individu en décrivant toutes ses capacités effectives et en regardant à posteriori s'il possède toutes les bonnes capacités pour jouer un rôle ; il se peut qu'un apprenant ne possède pas encore une capacité mais le but de la formation est justement de lui faire acquérir cette capacité et il ne faut donc pas l'empêcher de jouer ce rôle pour autant. En effet, pour la formation, un utilisateur doit pouvoir endosser n'importe quel rôle, du moment que le formateur l'y autorise. De plus il serait très contraignant de spécifier individuellement les capacités de chaque acteur (qu'il soit réel ou virtuel). Nous avons donc choisi de définir des individus génériques et de les adapter en fonction du rôle auquel ils souhaitent se former.

Au lieu de lier fortement le rôle aux actions du scénario, et de laisser les acteurs changer de rôle, en prendre plusieurs, etc, nous avons choisi de lier fortement le rôle à l'acteur pour toute la durée de la session de formation, afin qu'un apprenant se forme à un seul rôle à la fois. Néanmoins, nos modèles permettraient le changement dynamique du rôle d'un acteur en cours de session, mais nous considérons que cela doit être de la responsabilité du formateur.

3.2.3 Utilisation du rôle

Dans l'état de l'art nous avions signalé qu'il existe deux approches différentes pour voir le rôle : on peut le considérer comme un ensemble d'actions du scénario à réaliser (comme une contrainte) ou comme des capacités supplémentaires (comme une extension des possibilités d'actions). Nous choisissons de combiner ces deux visions du rôle. Pour nous un rôle se traduit par un ensemble de capacités d'interaction supplémentaires et il permet ensuite de décrire dans le scénario ce que l'acteur peut ou doit faire. En effet, comme nous le préciserons dans la section 4.4, une action peut être associée à plusieurs rôles, cette action ne représente donc pas forcément une contrainte d'action pour un rôle donné. Les rôles nous servent à exprimer des

possibilités d'actions dans le scénario et permettent aux acteurs de déterminer ce qui est de leur ressort et ce qui ne l'est pas. Si le rôle joué par l'acteur est le seul associé à l'action, alors seul cet acteur sera le seul à pouvoir faire cette action, elle sera alors de sa responsabilité. Mais comme le scénario exprime déjà des possibilités d'actions et non des obligations d'actions, il se peut que l'acteur n'ait pas à réaliser effectivement cette action. En effet, le scénario représente ce qui peut être fait à tout moment dans la procédure, les rôles permettent eux d'ajouter une information sur qui peut faire ces actions. Mais ces actions ne représentent pas des obligations, sauf s'il n'y a qu'une action possible à un instant donné et que cette action ne peut être réalisée que par un seul rôle.

3.2.4 Synthèse

Dans la litérature, un rôle représente souvent les actions que l'acteur **doit** faire. Nous avons choisi de considérer qu'un rôle représente les actions qu'il **peut** faire. Un rôle symbolise donc les responsabilités de celui qui l'endosse, il peut correspondre à un métier (par exemple expert ou technicien) ou à des responsabilités endossées ponctuellement (par exemple un travail particulier peut nécessiter deux personnes, une supervisant le tout jouant donc le rôle de chef et une autre celui d'assistant). Lorsqu'un rôle est attribué à un acteur, cela permet de faire le lien entre l'acteur et les actions du scénario, l'acteur acquiert de nouvelles capacités (qui correspondent aux compétences liées à ce rôle) et endosse les responsabilités associées au rôle (il doit effectuer les actions du scénario qui sont associées à ce rôle). Un rôle donne donc des droits (possibilités d'interactions) mais aussi des responsabilités (actions à réaliser).

3.3 Prise de décision

Un humain virtuel, tout comme un utilisateur réel, doit réaliser des actions dans l'environnement ; pour cela il va devoir choisir des actions à réaliser. Nous proposons un mécanisme capable d'effectuer cette sélection. L'humain virtuel pourra ensuite exécuter effectivement dans l'environnement l'action sélectionnée. Pour l'utilisateur réel, cette action pourra lui être révélée en cas de demande d'aide et lui sera ainsi divulguée comme un conseil pédagogique. Pour choisir une action, l'acteur va disposer d'une analyse de l'état actuel du scénario qui pourra lui permettre d'anticiper l'activité des autres acteurs. Cette phase de prise de décision fait partie d'un mécanisme global de sélection d'action qui sera décrit dans le chapitre 5.

Pour guider ce choix, l'acteur dispose d'un profil collaboratif, constitué d'un ensemble de règles de sélection d'actions auxquelles un poids est attribué (profil = $\{r \in R | P(r) > 0\}$, avec R ensemble des règles proposées, et $P(r)$ le poids de la règle r). Le profil collaboratif permet donc de paramétrer le comportement d'un acteur en décrivant ses propensions c'est à dire les tendances de son comportement à réaliser certains types d'opération, comme par exemple vouloir faire avancer la procédure ou au contraire tenter de perturber les apprenants. Un utilisateur réel aura un profil standard lui indiquant de suivre la procédure et l'action sélectionnée pourra ainsi être utilisée pour lui donner des conseils pédagogiques. Un humain virtuel pourra avoir un profil plus complexe et l'action ainsi sélectionnée sera ensuite exécutée. Le mécanisme de prise de décision est décrit dans la section 5.1.2. Dans cette même section nous détaillerons

les règles actuellement proposées pour créer un profil collaboratif ainsi que la place prise par le profil collaboratif dans le processus global de sélection d'action.

Le profil collaboratif est donc ce qui va permettre de différencier plusieurs humains virtuels en paramétrant leur comportement. Nous pouvons profiter de ce potentiel pour adapter le comportement des humains virtuels à la situation pédagogique que l'on souhaite mettre en place et utiliser le profil collaboratif à des fins pédagogiques. Plusieurs auteurs proposent différents rôles pédagogiques qui peuvent s'avérer utiles dans un contexte de formation. Afin de ne pas nous restreindre à ces rôles pédagogiques et pour pouvoir avoir une variété de comportement plus grande, nous préférons utiliser le terme de profil collaboratif. Le profil collaboratif contient un certain nombre de règles de comportement qu'un humain virtuel pourra avoir dans un scénario collaboratif. Chaque règle représente un élément de comportement, une propension, par exemple aider les autres, suivre la procédure, etc. Un humain virtuel ayant un profil de coéquipier aura tendance à réaliser les actions qui lui sont assignées et à aider les autres s'il en a l'occasion. Au contraire un humain virtuel avec un profil de perturbateur choisira plutôt des actions hors procédure et évitera les actions collaboratives.

3.4 Interchangeabilité entre utilisateur réel et humain virtuel

Nous nous étions fixé comme contrainte de pouvoir interchanger facilement et dynamiquement un utilisateur réel et un humain virtuel. Notre modèle d'activité d'un acteur permet cela car les deux types d'acteurs sont modélisés de manière similaire. Leur seule différence ici se situe au niveau du profil collaboratif qui peut être complexe pour un humain virtuel alors que nous préférerons un profil standard pour un utilisateur réel. Lors du passage d'un utilisateur réel à un humain virtuel nous pourrions donc simplement avoir à changer le profil collaboratif de l'acteur. Néanmoins, rien n'empêche d'avoir un humain virtuel avec le même profil collaboratif qu'un utilisateur réel, ni un utilisateur réel d'avoir le même profil collaboratif qu'un humain virtuel. Dans ce cas, l'utilisateur réel pourra ignorer le profil collaboratif de l'humain virtuel en ne faisant pas appel à l'aide pédagogique (qui elle peut tenir compte du profil collaboratif) et un humain virtuel avec le même profil collaboratif qu'un utilisateur réel aura simplement tendance à suivre à la lettre la procédure.

Nous venons de voir qu'un humain virtuel s'appuyait sur son profil collaboratif pour sélectionner une action à réaliser. Une fois l'action sélectionnée par l'utilisateur (par le menu de choix d'action) ou par l'humain virtuel, la réalisation de l'action est identique pour un humain virtuel et un utilisateur, les demandes d'actions passent par le même canal. La durée de l'action par exemple est fixe et l'utilisateur ne peut pas l'influencer. Le passage d'un humain virtuel à un utilisateur réel est donc possible dynamiquement, il suffit juste de désactiver la réalisation automatique d'action détaillée dans la section 5.1.3.

3.5 Synthèse

La figure 3.6 synthétise les différents attributs d'un acteur dont dépend son activité et qui définissent ainsi son comportement : ses capacités, son rôle et son profil collaboratif. Ces attributs sont exploités par différents modules qui vont s'en servir pour filtrer des actions possibles

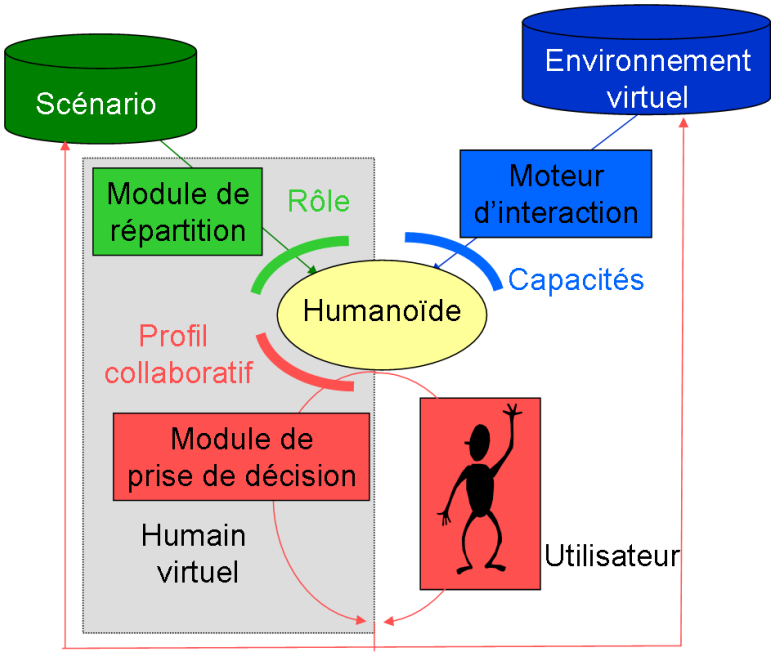


FIG. 3.6 – Intégration du modèle de l'activité d'un acteur

provenant de différentes sources. Les flèches représentent ainsi le flux des actions entre les différentes modules. La partie grisée du schéma représente les ajouts par rapport à la version mono-utilisateur de GVT. Les ajouts consistent donc à ajouter un nouveau filtre entre un acteur et le scénario, ce qui est indispensable puisque désormais le scénario est multi-utilisateur et toutes les actions ne sont donc pas destinées à être réalisées par un acteur donné. Le deuxième filtre, le profil collaboratif, est nécessaire pour interfacer plusieurs acteurs, pour paramétrer leur comportement vis à vis des autres acteurs avec lesquels ils partagent la scène.

Nous avons vu qu'un acteur dispose de différentes capacités : des capacités héritées, de capacités spécifiques, des capacités provenant de son rôle et des capacités provenant de ses ressources. Les capacités d'interaction sont exploitées par le moteur d'interaction (voir introduction) qui sert d'intermédiaire entre l'acteur et l'environnement virtuel. Le moteur d'interaction filtre les actions de l'environnement pour ne fournir à l'acteur que les actions possibles en fonction de son état et de ses capacités. Le rôle est exploité par le mécanisme de répartition des actions qui sert d'intermédiaire entre l'acteur et le scénario. Le module de répartition filtre les actions courantes du scénario pour ne fournir à l'acteur que les actions autorisées en fonction du rôle qu'il joue, enrichies par des informations sur la pertinence pour lui de réaliser cette action. Le profil collaboratif est exploité par le mécanisme de prise de décision de l'acteur. Ce mécanisme va filtrer les actions que l'acteur pourrait faire pour obtenir une action que l'acteur va effectivement réaliser, qui sera ainsi visible par les autres acteurs, qui va changer l'environnement virtuel et éventuellement faire évoluer le scénario. Le rôle, le profil collaboratif ainsi que les mécanismes de répartition des actions et de prise de décision seront davantage détaillés

dans la section 5 qui décrit de manière globale tout le processus de sélection d'action.

Une partie du comportement de l'acteur, la partie consistant à analyser le scénario et l'état des autres acteurs, a été mutualisée au sein d'un module centralisé. Ce module fait le lien entre le scénario (central) et les acteurs. La question de son positionnement se posait alors, fallait-il rattacher ce mécanisme aux acteurs et le rendre local ou le rattacher au scénario et le rendre global. Afin d'éviter des analyses redondantes et trop d'échanges d'information entre tous les acteurs nous avons donc choisi de rendre ce module central.

Chapitre 4

Scénario collaboratif

Un scénario collaboratif doit décrire l'activité de plusieurs personnes ; il regroupe donc des passages où l'activité est plutôt individuelle, des passages où l'activité est plutôt collective avec des interconnexions (liens temporels et logiques) entre les activités de chaque membre du groupe et des passages où l'activité est réellement collaborative. Afin de décrire un scénario collaboratif, il a donc fallu étendre la spécification du langage LORA [MA06] et créer un langage plus expressif : LORA++.

Dans une procédure collaborative, la collaboration peut s'exprimer de manières diverses. Elle peut tout d'abord se traduire par des liens temporels entre des actions devant être réalisées par des personnes différentes, nous allons donc nous intéresser dans la section 4.1 aux contraintes temporelles dans le scénario. Elle peut également se traduire par une communication entre les différents acteurs, ce que nous verrons dans la section 4.2. Enfin, certaines actions du scénario devront être réalisées par plusieurs acteurs, ce sont les actions collaboratives que nous détaillerons dans la section 4.3. Toutes ces formes de collaboration doivent pouvoir être réalisées entre utilisateurs réels, entre utilisateur réel et humain virtuel ou entre humains virtuels.

De plus, afin de répondre aux besoins de notre cadre applicatif, nous voulons décrire une procédure collaborative de manière suffisamment souple pour qu'elle puisse s'adapter au contexte, mais aussi de manière suffisamment précise pour que sa réalisation respecte strictement la procédure de référence. Notre objectif est triple, nous souhaitons sélectionner dynamiquement les acteurs qui vont réaliser chaque action du scénario (comme le suggère la théorie des rôles [BT66]) afin de rendre le scénario plus adaptatif, nous cherchons à réduire la complexité de spécification du scénario par l'auteur de scénario et nous voulons permettre aux industriels de tester facilement des nouvelles procédures en leur proposant une mise au point incrémentale des fiches d'interventions. Pour remplir ces objectifs, il a fallu revoir l'écriture des scénarios et en particulier proposer une utilisation de la notion de rôle différente de ce qui se fait dans les EVFC actuels, ce que nous détaillerons dans la section 4.4, puis introduire une gestion de l'implicite (section 4.5) et enfin nous proposons plusieurs stratégies pour remédier à certaines situations de blocage des acteurs (section 4.6).

4.1 Contraintes temporelles

Notre objectif ici est d'exprimer dans le scénario les différents moyens permettant à plusieurs acteurs de coordonner leur activité, tout en gardant un scénario simple à écrire (par l'auteur de scénario) et à comprendre (grâce à sa description graphique). Nous voulons donc pouvoir rendre compte de la diversité des situations collaboratives mais sans ajouter trop de contraintes quant à la description du scénario.

Durée des actions Le premier type de contrainte temporelle dans le scénario concerne la durée des actions et le lien entre la durée d'une action dans l'environnement virtuel et la durée réelle de réalisation de cette action. C'est en réalité un problème d'usage et le choix d'avoir ou non une corrélation doit être guidé par le besoin pédagogique. En effet, dans un environnement virtuel de formation, ce n'est pas l'hyper-réalisme qui est recherché mais bien la pertinence pédagogique. En fonction du type de formation, le formateur peut vouloir insister sur l'importance de certaines actions en augmentant leur durée ou au contraire réduire le temps d'actions non importantes. Le projet FIACRE [Lou01] pour la formation des agents de conduite de la SNCF illustre bien ce principe. Dans cette application l'apprenant doit se rendre compte du temps passé à marcher jusqu'à l'aiguillage, cela fait partie de sa formation ; il y a donc équivalence entre le temps mis à marcher dans l'environnement virtuel et le temps passé à marcher dans le monde réel. Par contre, dans d'autres environnements, les déplacements sont effectués de manière instantanée car ce n'est pas un point important de la formation et qu'ils peuvent même s'avérer être contre productifs. Nous avons donc fait le choix de ne pas nous contraindre à avoir des durées d'actions réalistes ou une correspondance entre la durée de l'action virtuelle et celle de l'action réelle. Les actions sont toutes réalisées par le système, même si elles sont lancées par l'utilisateur et la durée d'une action donnée est donc constante.

Agencement temporel des actions Le deuxième type de contrainte temporelle sert à exprimer l'ordonnancement des actions de la procédure. En effet une procédure doit décrire l'agencement temporel des actions et en particulier le fait qu'une action doit se faire avant, pendant ou après une autre action. De plus dans un EVFC, la coordination entre les différents acteurs peut être un point important de la formation.

Pour décrire l'agencement temporel des actions de la procédure il est possible d'exprimer la procédure comme un ensemble de contraintes temporelles entre les actions. Ces contraintes peuvent être exprimées par les opérateurs d'une logique temporelle, comme par exemple la logique temporelle de Allen [All83] qui s'applique aux intervalles. Lors de la création du langage LORA, la possibilité d'exprimer la procédure par un ensemble de contraintes (qu'elles soient uniquement temporelles ou également logiques) avait été exclue. En effet, pour un même ensemble de contraintes, il existe généralement plusieurs déroulements possibles ; l'obtention d'un déroulement fixé nécessite alors l'ajout par l'auteur de scénario de contraintes implicites, ce qui s'avère très contraignant. Ce choix n'est pas remis en cause par l'introduction de la dimension collaborative. L'utilisation d'opérateurs temporels pour décrire la procédure permet certes d'exprimer une synchronisation fine entre les différents intervenants, mais elle n'est pas très adaptée pour notre contexte de formation. En effet, dans notre type d'application, il n'y a pas que des humains virtuels qui doivent réaliser le scénario, il y a aussi des utilisateurs réels.

Il est donc impossible de déléguer à un gestionnaire de contraintes l'analyse du scénario, la synchronisation entre les acteurs et le déclenchement des actions. En effet un utilisateur réel lance lui-même ses actions. Par contre il ne peut pas contrôler le moment où l'action se terminera car la durée de l'action est fixe. Il est alors difficile d'imposer des contraintes temporelles trop fortes aux utilisateurs car ils risquent ne pas pouvoir les respecter. Il faudrait donc prévoir une gestion spécifique en cas d'échec sur la réalisation de ces contraintes. Dès lors, il n'est pas judicieux d'avoir des opérateurs de synchronisation très fins. Parmi les opérateurs de Allen, l'opérateur *meet* par exemple est impossible à faire respecter dans un EVFC et l'opérateur *overlaps* n'a pas grande signification si l'utilisateur ne peut pas modifier la durée d'une action. Par contre, ce type d'opérateurs temporels plus précis est mieux adapté dans le cadre de réalisation automatique de scénario par des agents virtuels pouvant recevoir des ordres d'un ordonnanceur central (par exemple pour vérifier la faisabilité d'une procédure).

Une application de formation requiert donc une expressivité temporelle plus simple utilisant seulement deux opérateurs : la séquence et le parallélisme. A ces deux opérateurs on peut ajouter une action permettant d'attendre une durée fixée. En effet pour traiter l'aspect multi-utilisateur il nous suffit de pouvoir décrire des actions devant être réalisées les unes à la suite des autres et des portions de scénario pouvant être réalisées en parallèle, avec des points de synchronisation. Beaucoup de langages de scénario proposent ainsi ces deux opérateurs de base (par exemple dans le projet GASPAR [MSBQ07] ce sont des diagrammes d'activité UML qui sont utilisés). Notre langage de scénario, LORA, permet lui aussi d'exprimer ce type d'ordonnancement simple, même s'il a été conçu pour décrire un scénario mono-utilisateur. Le parallélisme permettait en effet à un apprenant de faire évoluer en parallèle plusieurs branches de scénario. La séquence exprime le fait qu'une action doit être terminée avant que l'autre puisse débuter. Le parallélisme exprime le fait que deux actions (ou branches d'actions) peuvent être réalisées en même temps ou dans un ordre quelconque. En pratique on ne connaît donc pas l'action qui sera réalisée en premier. Une information logique est ensuite rajoutée à ces branches pour définir si elles doivent être réalisées toutes les deux (branchement en ET) ou si l'une des deux branches est suffisante (branchement en OU).

Un autre point important est que la procédure de référence papier (celle qui figure dans les cartes de travail, voir annexe A) est décrite de manière séquentielle, avec parfois des alternatives, et utilise donc ces deux opérateurs de base plutôt que des contraintes temporelles précises. Sa retranscription dans un langage comme le langage LORA est donc plus directe. De plus le graphe de scénario ainsi obtenu permet de visualiser facilement les différents déroulements possibles de la procédure.

Il est important de noter qu'avec ces opérateurs simples, il est toutefois possible d'exprimer des opérateurs plus complexes comme l'opérateur *overlaps* évoqué précédemment. Pour cela il suffit de scinder l'action en deux et de séparer ainsi le déclenchement de l'action et l'arrêt de l'action. Un utilisateur pourra ainsi faire varier la durée virtuelle de cette action en décidant du moment de sa terminaison. Cette décomposition peut être utile pour des actions non ponctuelles dont la durée a une importance. On peut alors distinguer deux types d'actions : les actions ponctuelles et les actions durables. Prenons comme exemple l'appui sur un bouton. Une action ponctuelle d'appui sur un bouton a pour conséquence d'appuyer sur le bouton puis de le relâcher aussitôt. Par exemple pour un bouton d'arrêt d'urgence, l'action est ponctuelle puisque un appui suffit à arrêter la machine et le bouton est immédiatement relâché. On peut

aussi décomposer cette action en deux : l'action d'appuyer et l'action de relâcher le bouton. Tant que le bouton est appuyé, un comportement spécifique peut alors être activé. Par exemple dans les stations de gonflage de pneu, tant qu'on appuie sur le bouton de gonflage l'air passe et la pression dans le pneu augmente ; lorsqu'on relâche le bouton l'air ne passe plus. Avec cette décomposition il est alors possible d'intercaler dans le scénario d'autres actions entre le démarrage et l'arrêt de l'action donnée.

4.2 Communication

Dans l'état de l'art nous avons vu que la communication est un élément important pour un environnement virtuel collaboratif puisqu'elle est nécessaire au niveau de collaboration le plus élémentaire : le niveau 1 de la classification de Margery [MAP99]. Cette communication, très libre, peut être utilisée pour faciliter l'apprentissage. Cependant, nous avons également noté qu'il existe un autre type de communication, plus codifié, qui peut apparaître dans la procédure. Notre objectif est donc d'exploiter ces deux types de communication.

Dans les procédures collaboratives, la communication a souvent une place importante, comme nous avons pu le voir dans les différents EVFC analysés dans l'état de l'art. En revanche la gestion de la communication peut se faire de différentes manières : elle peut servir à faciliter l'apprentissage, elle peut être libre ou alors scénarisée. Nous avons choisi de différencier la communication informelle, qui a pour but de faciliter l'apprentissage en échangeant avec les autres participants, de la communication formelle faisant partie de la procédure. En effet, dans certaines procédures industrielles, les échanges verbaux sont codifiés et décrits dans la procédure de référence. Ce deuxième type de communication est alors considéré comme une action à part entière et est scénarisé comme n'importe quelle autre action.

Chaque acteur va donc disposer d'une relation de communication. Les différentes actions possibles pour cette relation correspondront aux différentes phrases que l'acteur peut utiliser. Ces phrases peuvent par exemple correspondre à des ordres (exemple : "démarre le moteur"), des notifications (exemple : "le moteur est allumé") ou des réponses (exemple : "la jauge affiche une valeur élevée"). Dans le scénario ces actions apparaîtront comme n'importe quelles autres actions. Il sera par exemple possible d'exprimer différents ordres possibles qu'un chef peut donner à un de ses subordonnés ; pour cela il suffit alors de faire une disjonction en OU exclusif, avec sur chaque branche l'ordre suivi par la réponse attendue (qui peut être elle aussi une réponse verbale ou qui peut être une action dans l'environnement). L'exemple de la figure 4.2 est extrait de la procédure dite "d'harmonisation du char Leclerc". Il s'agit de mettre en correspondance tous les axes de visée par rapport à l'axe de tir. Dans cet extrait qui concerne la séquence de pointage du canon sur la mire, deux acteurs interviennent : l'opérateur à la lunette de bouche (LB) et le tireur (PT) (voir leur localisation sur la figure 4.1). L'opérateur à la lunette de bouche est chargé de regarder la mire à travers la lunette de bouche et de donner des ordres au tireur pour aligner le canon sur la mire. Le tireur donne alors l'impulsion correspondante sur l'interrupteur entouré en rouge sur la figure 4.1. L'extrait de scénario de la figure 4.2 concerne le réglage en site du canon. Les actions de l'opérateur à la lunette de bouche sont en bleu et celles de l'acteur au poste tireur sont en jaune.

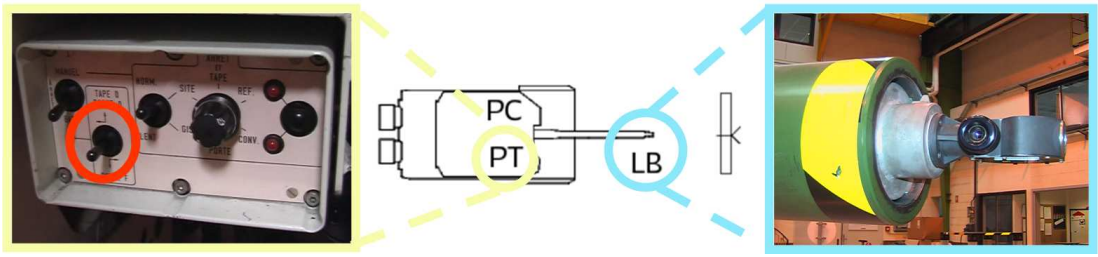


FIG. 4.1 – Positions des acteurs pour la procédure d'harmonisation

4.3 Actions à plusieurs

Notre objectif est de permettre d'utiliser tous les types de collaboration dans notre EVFC. Dans l'état de l'art nous avons montré que dans un EVFC la collaboration peut se situer au niveau de l'apprentissage. Cette première forme de collaboration s'appuie sur la communication entre les apprenants dont nous venons de parler. La collaboration peut également se situer au niveau du scénario, avec des actions de communication ou une coordination des activités de chacun, nous avons également abordé ces aspects. Enfin la collaboration peut se situer au niveau d'une action et c'est ce que nous allons évoquer dans cette section.

Nous allons donc maintenant raffiner davantage le niveau de collaboration en nous intéressant à la modélisation d'une action collaborative, c'est à dire une action à réaliser à plusieurs. Une action collaborative est définie grâce à une relation STORM qui prend comme paramètres plusieurs acteurs. A la différence d'une action de communication où l'un des acteurs est actif et choisi de parler à un autre acteur qui lui est passif (il se contente de recevoir la communication), ici nous allons nous intéresser à des actions où les acteurs impliqués sont tous actifs. Avant la réalisation de cette action collaborative chaque acteur doit réaliser une action préparatoire afin de signaler qu'il est prêt à collaborer. Ces actions préparatoires peuvent être regroupées sous le terme générique de *déclaration d'intention de collaborer*. La figure 4.3 illustre la succession d'actions à réaliser pour porter une table à deux. Ici l'acteur 1 a déjà déclaré qu'il était prêt à collaborer (les éléments rouges représentent l'état courant), l'acteur 2 fait alors de même (éléments verts) et l'action porter la table est alors automatiquement lancée (éléments bleus).

Dans certains cas, la déclaration d'intention peut être remplacée par une action réelle : dans l'exemple que nous venons de voir l'acteur pourrait par exemple soulever un côté de la table pour indiquer qu'il est prêt à porter la table comme l'illustre la figure 4.4.

L'action collaborative peut être vue comme une synchronisation entre le comportement de plusieurs acteurs. En effet comme l'illustrent les figures 4.3 et 4.4, l'action collaborative à proprement parler va faire évoluer non seulement l'automate de la relation STORM mais également les automates représentant l'état de chacun des acteurs qui vont évoluer de manière similaire. Nous avons donc ici une mise en correspondance des automates de comportement des différents acteurs impliqués.

Une action collaborative est donc une action impliquant plusieurs acteurs et au moins un objet sur lequel porte l'action. Cet objet sert d'intermédiaire et c'est sur lui que les acteurs vont pouvoir indiquer qu'ils sont prêts à collaborer. Dans l'exemple précédent il s'agissait d'une

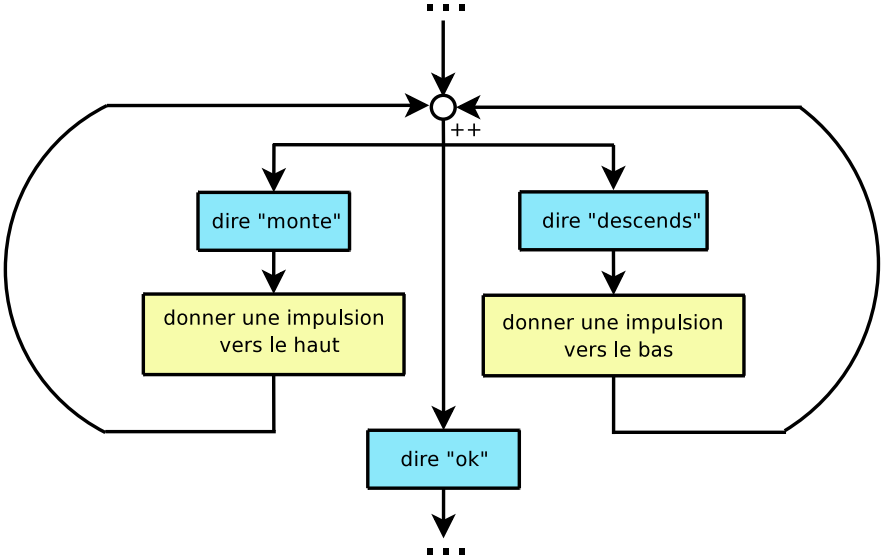


FIG. 4.2 – Extrait de la séquence de pointage du canon sur la mire - réglage en site

table. Chaque acteur signale donc qu'il veut interagir avec cet objet et il en résulte que les deux acteurs agissent ensemble sur ce même objet et collaborent. Les acteurs n'interagissent pas directement, il agissent simplement sur le même objet. L'action de communication est une action collaborative particulière puisqu'elle n'implique pas d'objet ; c'est une relation directe entre deux acteurs. Elle ne nécessite donc pas de déclaration d'intention de collaborer.

Le scénario décrit la procédure nominale. Il fait donc apparaître, le cas échéant, des actions collaboratives. Comme nous l'avons mentionné précédemment chaque acteur considère que tous les acteurs partagent la même volonté d'accomplir au mieux la procédure. Ainsi les acteurs ayant un profil collaboratif leur indiquant de suivre la procédure (par exemple les apprenants) auront intérêt à collaborer dès que la procédure le leur indiquera. En revanche cela ne veut pas dire que tous les acteurs seront toujours prêts à collaborer ; ils peuvent, en fonction de leur profil collaboratif ou du contexte, décider de ne pas collaborer (par exemple s'ils ont un profil de perturbateur ou s'ils considèrent qu'une autre action est plus urgente). Dans ce cas la portion de scénario où figure l'action collaborative sera en attente.

4.4 Assignation des personnes aux actions

Lors de l'écriture d'un scénario collaboratif, il faut décrire l'assignation des personnes aux actions du scénario. Notre objectif est d'autoriser une répartition flexible et dynamique des actions du scénario entre les acteurs ; nous proposons donc une nouvelle méthode pour assigner plusieurs rôles et/ou personnes à une action du scénario donnée ; c'est ensuite dynamiquement que l'acteur qui va réaliser l'action sera choisi (grâce à la contribution présentée dans le chapitre

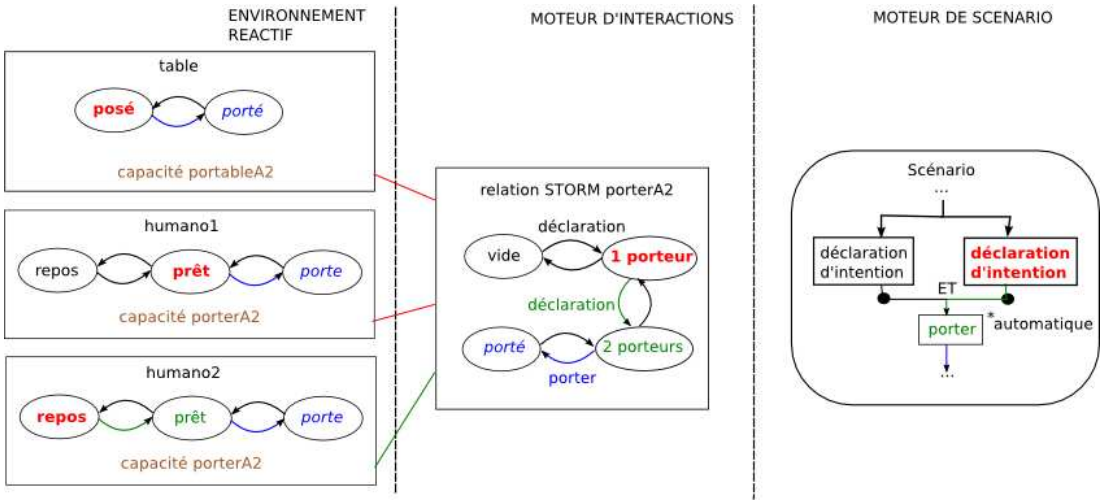


FIG. 4.3 – Evolution des automates lors d'une action collaborative, avec déclarations d'intention

5). L'intérêt est alors d'écrire un seul scénario correspondant à une procédure de référence, mais de le rendre adaptable à des contextes de réalisation différents (nombre de participants variable, un critère différent à prioriser, modification de la situation initiale, outils disponibles différents, etc). Grâce à notre nouvelle méthode de description, le même scénario peut ainsi avoir une multitude de réalisations possibles en fonction du contexte. Pour décrire l'assignation des personnes aux actions du scénario nous proposons deux moyens : l'utilisation des rôles ou l'utilisation des fils d'activité.

4.4.1 Constats

Lors de notre état de l'art nous avons constaté un manque de souplesse dans les EVFC actuels quant à la répartition des actions entre les différents acteurs. En effet, un seul rôle est assigné à une action du scénario. Cette attribution statique ne permet pas de traduire la réalité de certaines procédures où les actions à réaliser sont bien définies mais où leur répartition est susceptible de varier en fonction du contexte de réalisation. Biddle et Thomas [BT66], dans leur théorie des rôles, corroborent ce constat et stipulent que lors d'un travail en équipe l'assignation des tâches peut s'effectuer de manière dynamique en fonction du contexte d'exécution. Ils soulignent ainsi qu'une assignation dynamique des rôles aux actions permet de mieux refléter la réalité.

Notre contribution consiste en une nouvelle définition de la notion de rôle. Dans le chapitre 3 nous avons vu que le rôle est associé à un ensemble de capacités STORM apportées à l'acteur qui joue ce rôle. Un rôle permet donc de représenter ce que l'acteur **sait** faire. L'utilisation du rôle dans un scénario collaboratif permet en général de spécifier les responsabilités de l'acteur jouant ce rôle, c'est à dire ce qu'il doit faire. Nous proposons d'assigner plusieurs rôles à une action du scénario et d'effectuer dynamiquement l'affectation finale. Le rôle va ainsi représenter non seulement ce que l'acteur **doit** faire mais également ce que l'acteur **peut** faire (si son

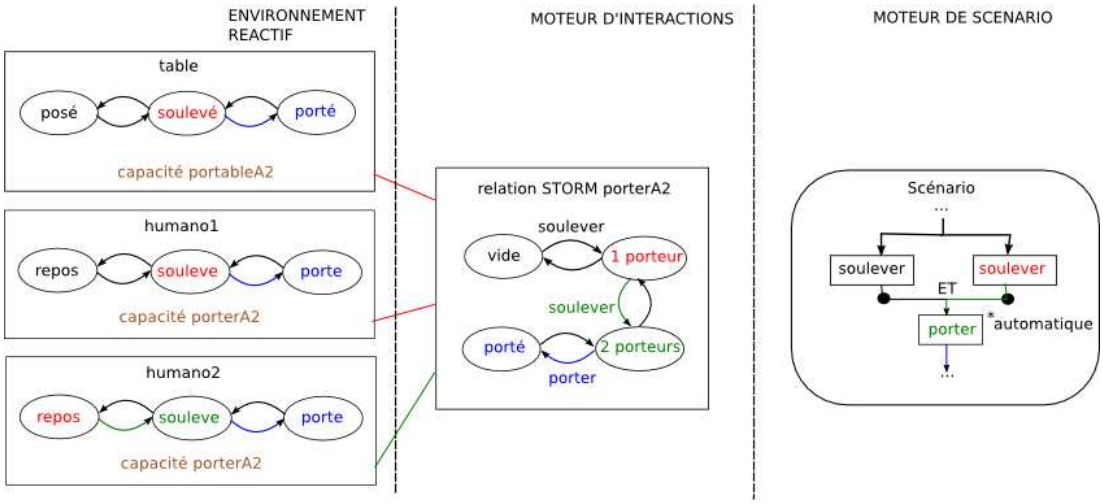


FIG. 4.4 – Evolution des automates lors d'une action collaborative ; les déclarations d'intention sont remplacées par des actions physiques

rôle n'est pas le seul associé à une action du scénario).

Le première méthode d'assignation des personnes aux actions sera donc l'utilisation des rôles, en précisant pour une action donnée les rôles pouvant effectuer l'action. Nous pourrons donc définir action par action, les rôles et donc les acteurs autorisés à réaliser l'action. Cependant, comme nous autorisons l'assignation de plusieurs rôles par action, nous ne savons donc pas pour une action donnée le rôle qui aura réalisé l'action précédente. Cela peut poser des problèmes pour préserver la continuité dans la réalisation de certaines suites d'actions du scénario.

Imaginons maintenant un scénario qui contient un ensemble d'actions devant obligatoirement être réalisées par la même personne (par exemple signer une série de documents). Plusieurs rôles sont autorisés à réaliser ces actions mais une fois que la première action a été effectuée c'est impérativement la même personne qui doit effectuer toutes les actions suivantes. Ce cas de figure n'est pas possible à décrire grâce à l'assignation par rôle qui ne tient pas compte des actions passées et qui ne permet donc pas d'exprimer un ensemble d'actions devant être réalisées par la même personne à moins que cette personne ne soit fixée à priori. C'est pour cela que nous avons mis en place le concept de fil d'activité. Les fils d'activités sont donc destinés à être utilisés pour des ensembles d'actions liées devant être réalisées par la même personne (ou au moins lorsqu'on considère que cela serait préférable). Les fils d'activité permettent ainsi d'assurer une certaine cohérence dans la réalisation de séries d'actions, en s'assurant qu'une fois que la première action aura été réalisée par une personne donnée, ce sera à elle de réaliser les actions suivantes. Il devient alors possible de spécifier qu'une séquence d'actions doit être réalisée par la même personne.

Prenons un exemple. Pour réaliser la procédure d'harmonisation du char Leclerc que nous avons évoquée dans la section 4.2, le nombre de personnes intervenant peut varier. S'il n'y a que deux personnes, le chef peut alors remplacer l'opérateur devant se trouver à la lunette de bouche pour donner des indications au tireur lors de la séquence de pointage. Les ordres donnés

au tireur pour régler la position du canon peuvent donc être donnés par deux rôles différents (le chef et l'opérateur à la lunette de bouche). Par contre lorsqu'un acteur jouant l'un de ces deux rôles donne le premier ordre, c'est lui qui doit continuer la séquence jusqu'au bout.

La deuxième méthode d'assignation des personnes aux actions sera donc l'utilisation des fils d'activités. Nous allons maintenant décrire la spécification du scénario grâce à ces deux méthodes et ensuite leur utilisation lors de l'exécution du scénario.

4.4.2 Description dans le scénario : assignation des personnes aux actions

4.4.2.1 Rôles

Nous avons proposé une extension du langage LORA pour intégrer cette notion de rôle. Lors de l'écriture d'une action, il est maintenant possible d'ajouter un ou plusieurs champs rôle pour préciser l'assignation des rôles aux actions. Un champ rôle (R) est délimité par une balise XML *role* et est défini par $R = (n, p)$ où n est le nom du rôle (ou *any* pour préciser que tous les rôles sont autorisés) et p sa priorité, à choisir parmi $\{$*interdit*, *secondaire*, *prioritaire*$\}$. Par défaut, tous les rôles sont autorisés et sont tous considérés comme ayant la même priorité. La figure 4.5 montre la description XML d'une action du scénario. Le rôle du technicien est prioritaire pour effectuer cette action ; cependant les autres rôles sont également autorisés mais avec une priorité plus basse.

```
<etape ordre="Devisser" description="Devisser la poignee">
      <lienSurveillance ordre="1" numConnecteur="0"/>
      <action>
            <role ordre="technicien" priorite="prioritaire"/>
            <role ordre="any" priorite="secondaire"/>
            <relation ordre="visserie" action="devisser"/>
            <parametre ordre="tournevis"/>
            <parametre ordre="visPoignee"/>
      </action>
</etape>
```

FIG. 4.5 – Description XML d'une action avec assignation de rôles

4.4.2.2 Fils d'activité

Lors de l'écriture d'une action du scénario, en plus des champs rôle (optionnels) nous pouvons rajouter un champ fil d'activité (Fa). Ce champ commence par la balise XML *fil* et est défini par $Fa = (n, m)$ où n est le nom du fil d'activité et m le mode d'utilisation : exclusif ou préférentiel. Deux modes d'utilisation du fil d'activité sont en effet proposés : le mode exclusif qui indique que seul l'acteur associé au fil peut faire l'action, et le mode préférentiel qui permet de considérer l'acteur associé au fil comme prioritaire pour l'action, mais qui autorise également tous les autres acteurs à réaliser l'action avec une priorité secondaire (à la condition qu'ils sachent la réaliser et qu'ils possèdent donc les bonnes capacités).

Reprenons l'exemple de la séquence de pointage lors de la procédure d'harmonisation (voir figure 4.2). Pour chacun des ordres pouvant être donnés en premier ("monte" ou "descends"),

nous spécifions deux rôles autorisés et nous précisons un fil d'activité (voir figure 4.6). Les ordres suivants ("monte", "descends" et "ok") utiliserons alors ce fil en mode imposé.

```
<etape ordre="Monte" description="Dire monte au tireur">
      <lienSurveillance ordre="1" numConnecteur="0"/>
      <action>
            <role ordre="opérateurLB" priorite="prioritaire"/>
            <role ordre="chef" priorite="secondaire"/>
            <fil ordre="pointage_ordres" mode="exclusif"/>
            <relation ordre="communiquer" action="monter"/>
            <parametre ordre="humano"/>
            <parametre ordre="tireur"/>
      </action>
</etape>
```

FIG. 4.6 – Description XML d'une action avec fil d'activité

Par ailleurs un fil d'activité, tout comme un rôle, peut être passé en paramètre d'un automate fonctionnel. Un automate peut donc être facilement réutilisé en adaptant le fil ou les rôles que l'on veut utiliser.

4.4.3 Utilisation à l'exécution du scénario : affectation dynamique des personnes aux actions

Grâce à notre méthode de description du scénario, il se peut que plusieurs personnes soit assignées à une action (plusieurs rôles spécifiés ou un fil d'activité en mode préférentiel). Lors de l'exécution du scénario le système doit donc déterminer une liste de candidats pour une action active. Cette liste pourra servir dans un but de contrôle, c'est à dire pour vérifier que l'acteur qui demande l'action est bien autorisé à la faire. Cette liste pourra également servir de point de départ à un mécanisme chargé d'évaluer le meilleur candidat pour une action du scénario donnée. Ce mécanisme sera décrit dans le chapitre 5.

Pour une action contenant des champs rôle, la liste de candidats est déterminée de la manière suivante :

- Les acteurs ayant un rôle autorisé sont tout d'abord ajoutés (ils sont tous ajoutés si le rôle *any* est précisé).
- Ensuite les acteurs ayant un rôle interdit sont enlevés de la liste des candidats.

Pour une action utilisant un fil d'activité (déjà initialisé), la liste de candidats dépend du mode d'activation :

- Si le fil est utilisé en mode exclusif seul l'acteur associé au fil est candidat.
- Si le fil est utilisé en mode préférentiel tous les acteurs sont candidats.

L'utilisation du fil d'activité nécessite quelques explications. Lors de l'exécution du scénario l'utilisation d'un fil d'activité se déroule en deux temps. La première phase, la phase d'initialisation, consiste à associer un acteur à un fil d'activité lorsque le fil est rencontré pour la première fois dans le scénario. La deuxième phase revient à substituer l'acteur au fil d'activité

à chaque fois que ce fil sera de nouveau rencontré dans le scénario. Nous allons détailler un peu plus chacune de ces deux phases.

phase d'association (initialisation) Lorsqu'une action du scénario devient active, nous vérifions si un fil d'activité est spécifié pour cette action. Si c'est le cas et que c'est la première fois que ce fil d'activité est rencontré dans le scénario, on le mémorise. Pour déterminer les acteurs autorisés à réaliser cette action ce sont les rôles assignés à l'action qui sont alors utilisés. Lorsqu'un acteur va réaliser cette action, il sera associé à ce fil d'activité. Cette association est définitive et concerne directement l'acteur (et non pas son rôle).

Pour cette phase, on utilise les rôles et on peut donc spécifier des rôles prioritaires. Ces rôles vont servir à déterminer l'acteur qui se verra attribuer l'action. Il faut donc bien choisir l'action qui va servir pour l'initialisation du fil d'activité car les critères pris en compte pour choisir l'acteur qui sera associé au fil seront ceux liés au contexte courant de réalisation de cette action (nombre d'outils en main des candidats, proximité, etc..). Une fois le fil attribué, il n'est plus possible de changer l'acteur associé au fil, mais il est toujours possible d'utiliser le fil en mode préférentiel qui permet de ne pas imposer cet acteur pour une action. Le choix de l'action de départ n'est donc pas anodin. Il est également possible d'avoir plusieurs actions pouvant servir à initialiser le fil d'activité, à différents endroits de la procédure. Ce sera alors l'action réalisée en premier qui va servir pour l'association entre l'acteur et le fil.

Dans l'exemple de la séquence de pointage, au premier passage trois actions sont possibles, *dire "monte"*, *dire "descends"* et *dire "ok"* ; pour chacune d'elles un nouveau fil d'activité est précisé, le fil *pointage_ordres*. Deux rôles peuvent réaliser ces actions, le chef et l'opérateur LB. Lorsqu'une de ces actions sera réalisée, l'acteur qui l'aura réalisée sera associé au fil d'activité.

phase de substitution Lorsqu'un fil d'activité qui a déjà été initialisé est rencontré dans le scénario, les rôles spécifiés sont alors ignorés et seul le fil d'activité est considéré, ainsi que son mode d'activation (exclusif ou préférentiel). Il est important de bien choisir le mode d'utilisation du fil d'activité. En effet, en mode exclusif, un seul acteur peut réaliser l'action et il n'y alors pas d'aide possible de la part des autres acteurs. Le mode préférentiel permet de laisser davantage de flexibilité dans la réalisation du scénario, tout en spécifiant qu'il serait préférable que ce soit l'acteur associé au fil qui fasse l'action. Il suffit alors de donner beaucoup de poids au critère de priorité (voir section 5.1.1.2) pour accentuer la préférence accordée à l'acteur associé au fil d'activité. Ce mode permet donc l'apparition d'une forme de collaboration implicite.

Reprenons l'exemple de la séquence de pointage. Le fil d'activité *pointage_ordres* a été associé à un acteur. Lors de l'activation suivante pour les actions *dire "monte"*, *dire "descends"* et *dire "ok"*, les rôles seront alors ignorés, c'est le fil qui sera considéré en mode imposé et donc seul l'acteur ayant donné le premier ordre pourra donner les ordres suivants.

Au fur et à mesure que le scénario est exécuté, nous pouvons mémoriser les acteurs ayant effectué chacune des actions. Il sera ainsi possible, à la fin de la session de formation, de représenter graphiquement le déroulement précis du scénario.

4.4.4 Représentation du déroulement du scénario

Notre scénario dispose d'une représentation sous forme de graphe qui permet de faire apparaître les liens temporels entre les actions (séquence et parallélisme). Une fois le scénario réalisé il peut être intéressant d'extraire les activités individuelles de chacun et d'y faire apparaître des informations et des liens qui ne figuraient pas dans le scénario d'origine, comme par exemple les situations de blocage et les liens de collaboration implicite qui en résultent. Pour cela on peut utiliser un diagramme d'activité UML par exemple, avec un couloir par acteur. Une telle représentation n'est pas possible avant la réalisation du scénario car généralement on ne connaît pas à l'avance l'acteur qui va réaliser une action. Les différentes possibilités de diagramme d'activité peuvent donc être nombreuses.

4.4.5 Synthèse

Notre approche consiste donc à respecter à la lettre la procédure de référence qui décrit la suite d'actions à réaliser (avec parfois plusieurs alternatives), mais dans cette procédure, plusieurs déroulements peuvent être possibles. Les activités de chacun des acteurs ne sont en effet pas figées. L'idée novatrice est ici de proposer une assignation multiple au niveau des actions des rôles ou acteurs autorisés puis une affectation dynamique de la personne qui va réaliser l'action. Nous avons ainsi supprimé le déterminisme sur l'acteur qui va pouvoir réaliser une action particulière du scénario, en proposant une autre définition de la notion de rôle et en liant le rôle aux acteurs plutôt qu'aux actions. Le lien exclusif se fait en effet entre l'acteur et le rôle et non plus entre l'action et le rôle. Classiquement dans un EVFC, grâce à l'association exclusive d'un rôle à une action du scénario, le rôle permet de décrire les actions que l'acteur doit faire. Pour nous le rôle représente ce que l'acteur sait, peut et doit faire. Plusieurs rôles peuvent ainsi être autorisés pour une même action du scénario, assortis alors d'une indication quant à leur priorité. Dans la littérature nous n'avons pas trouvé d'EVFC proposant cette liberté. En revanche, cela implique qu'un humain, qu'il soit réel ou virtuel, ne sait pas toujours si c'est à lui de réaliser l'action suivante de la procédure. Nous proposons donc, pour y remédier, un mécanisme qui nous permettra de déterminer qui est la personne la plus adaptée pour réaliser une action donnée à un moment donné, dans un contexte donné. Ce mécanisme sera décrit dans le chapitre 5. Nous serons ainsi capables de déterminer à tout moment qui est l'acteur le mieux placé pour faire l'action.

Il est toutefois important de noter qu'il est possible d'écrire un scénario très strict où la répartition des tâches est fixe. Pour cela il suffit d'assigner un seul rôle à chaque action si la procédure l'impose. Notre approche permet donc simplement de représenter une plus grande variété de procédures.

Nous proposons deux méthodes pour décrire les acteurs qui peuvent réaliser les actions du scénario. La première, l'usage des rôles, permet de décrire l'assignation des acteurs aux actions pour des actions ponctuelles ou indépendantes. L'usage des fils d'activité permet de décrire des séries d'actions à réaliser par la même personne. Il est bien sûr possible de combiner au sein d'un même scénario l'utilisation des fils d'activité et l'utilisation des rôles aux niveaux d'actions spécifiques. On gagne ainsi en flexibilité et en simplicité pour spécifier les rôles autorisés pour les actions du scénario.

Cette liberté sur l'assignation des rôles aux actions amène des propriétés intéressantes et nouvelles pour un EVFC : le scénario est plus flexible et adaptable au contexte, les acteurs ont plus de liberté sur leurs choix d'action (actions faisant partie de la procédure) et peuvent par exemple s'aider les uns les autres. Nous détaillerons ces avantages dans la section 5.3.

4.5 Gestion de l'implicite

Afin de simplifier la tâche de l'auteur de scénario, nous avons cherché à simplifier le scénario en permettant d'y insérer de l'implicite. Nous allons donc rendre implicite certaines actions de base qui ne seront plus écrites dans le scénario et qui seront remplacées par la spécification de pré et post conditions sur l'état des ressources (section 4.5.1). Nous allons aussi définir des préconditions sur l'état d'une relation STORM (section 4.5.2). Nous allons également typer les objets afin de pouvoir raisonner dans le scénario sur ces types et non plus seulement sur les instances des objets (section 4.5.3). Mais ajouter de l'implicite dans le scénario implique une complexité accrue à résoudre au niveau du moteur de scénario. En effet on ne peut plus être certain que les actions implicites ont été réalisées ; il va falloir également désambiguïser automatiquement certaines actions ce qui auparavant était le travail de l'auteur de scénario.

4.5.1 Pré et post conditions sur l'état des ressources

Dans la description des procédures industrielles (dans les cartes de travail par exemple) certaines actions sont implicites, par exemple les actions de prise d'outils. Or notre langage de scénario imposait à l'auteur d'extraire des procédures ces actions implicites afin de les écrire dans le scénario. Pour nous rapprocher davantage de la description réelle de la procédure, nous avons donc décidé de simplifier le scénario en rendant implicites des actions basiques comme prendre un outil et poser un objet. Il faut alors préciser pour chaque action du scénario qui requiert un état précis concernant une ressource de l'acteur une pré-condition et une post-condition sur cet état.

Comme nous l'avons vu dans la section 3.1.1.1, pour le moment nous gérons uniquement les mains comme ressources pour l'acteur. Il faut donc spécifier dans le scénario, là où cela est utile, l'état des mains de l'acteur requis pour faire l'action (pré-condition) et l'état des mains après l'action (post-condition). Cela permet de raisonner sur ces états afin de déterminer les actions prérequises à l'action (prise d'outil et pose d'objets). Cela permet également d'identifier les actions qui rendent une ressource occupée et qui sont donc potentiellement une source de blocage. Chaque condition est donc un couple d'état, un pour la main droite, l'autre pour la main gauche ; l'ordre n'a pas d'importance puisque nous avons décidé de considérer l'acteur comme ambidextre (voir section 3.1.1.2). Ces états sont à choisir parmi les trois états que peuvent avoir une main de l'acteur : *libre*, *objet attrapé* (l'objet peut servir d'outil) ou *occupée*, auxquels s'ajoute l'état *indifférent* qui indique que la condition sera valide quelque soit l'état effectif de la main. Pour l'état *objet attrapé* on précise également l'objet requis et pour l'état *occupée* la cause de l'occupation (la relation concernée). La figure 4.7 illustre l'utilisation des pré et post conditions.

En réalité la spécification des pré et post conditions est facultative (notamment afin de garder la compatibilité avec les anciens scénarios) et seulement indicative. En effet la faisabilité

de l'action sera évaluée par le moteur d'interaction en fonction des capacités et des états des relations et objets impliqués. Le moteur d'interaction ne vérifie donc pas explicitement la précondition écrite dans le scénario, mais en pratique si elle n'est pas vérifiée l'action ne sera pas possible. Le fait d'écrire la précondition dans le scénario pourrait alors paraître redondant avec les informations contenues dans la définition des relations. Les pré et post conditions explicitées dans le scénario servent en fait pour raisonner : c'est grâce à elles qu'il sera possible de déterminer les actions implicites nécessaires à la réalisation d'une action du scénario. Comme nous l'avons vu dans la section 3.1.1.2, le gestionnaire de ressources va en effet se servir de la précondition pour calculer une séquence d'action permettant d'atteindre cet état cible à partir de l'état courant de l'acteur. Les post conditions vont servir pour effectuer des raisonnement hors ligne par exemple (voir la section 4.6.2). En effet, là aussi c'est le moteur d'interaction qui gère les conséquences des actions et donc l'évolution des objets et relations STORM.

```
<etape ordre="Devisser" description="Devisser la poignee">
      <lienSurveillance ordre="1" numConnecteur="0"/>
      <action>
            <role ordre="technicien" priorite="prioritaire"/>
            <role ordre="any" priorite="secondaire"/>
            <condition>
                  <pre etat="objet" detail="tournevis"/>
                  <post etat="objet" detail="tournevis"/>
            </condition>
            <relation ordre="visserie" action="devisser"/>
            <parametre ordre="tournevis"/>
            <parametre ordre="visPoignee"/>
      </action>
</etape>
```

FIG. 4.7 – Description XML d'une action avec pré et post condition sur l'état des mains

Les actions de prise et de pose d'objets sont des actions très courantes dans une procédure de maintenance, le fait de ne plus avoir à les écrire dans le scénario permet donc de simplifier considérablement l'écriture de ce dernier. Par ailleurs rendre ces actions implicites permet une meilleure adaptabilité du scénario (possibilité de se resservir d'un outil déjà en main) et évite des situations incohérentes. En effet, un acteur pourra alors choisir de reposer un outil s'il ne s'en ressert pas tout de suite ou au contraire de le conserver s'il en a besoin pour réaliser une autre portion du scénario. L'acteur ne sera donc pas contraint d'adopter la solution retenue par l'auteur de scénario qui aurait pu conduire à des situations absurdes où il aurait dû reposer un outil et le reprendre aussitôt.

4.5.2 Préconditions sur l'état d'une relation

En plus des préconditions sur l'état des ressources d'un acteur nous proposons pour les actions du scénario des préconditions sur l'état d'un type de relation STORM donné sur un objet donné. L'action correspondante ne sera alors possible que si la précondition est vérifiée. Il faut donc préciser l'objet sur lequel porte la relation, le type de la relation ainsi que l'état

voulu. Par exemple, la figure 4.8 montre une action du scénario qui consiste à appuyer sur un bouton pour stopper le moteur. En précondition on précise qu'il faut que le moteur soit en marche en disant que la relation de type relationActif sur l'objet moteur doit être dans l'état actif.

```
<etape ordre="stopMoteur" description="Appuyer sur le bouton arrêt">
      <lienSurveillance ordre="4" numConnecteur="1"/>
      <action>
            <preCondition objet="moteur" relation="relationActif"
            etat="actif" description="moteur en marche"/>
            <relation ordre="relationBougeBouton" action="pingPong"/>
            <parametre ordre="boutonArret"/>
            <parametre ordre="Main"/>
      </action>
</etape>
```

FIG. 4.8 – Description XML d'une action avec précondition sur l'état d'une relation

Contrairement aux préconditions sur l'état d'une ressource les préconditions sur l'état d'une relation ne sont pas juste indicatives, elles sont discriminatoires. L'action sera en attente tant que la précondition ne sera pas vérifiée. Une fois la précondition vérifiée, l'action deviendra alors possible.

Ces préconditions ne servent qu'à faire une vérification pour pouvoir réaliser une action du scénario, en revanche rien ne nous permet pour le moment de raisonner sur les actions pour trouver les actions nécessaires à la vérification de la précondition. Grâce à ces préconditions il est donc possible de faire des vérifications sur les actions réalisées jusque là et de proposer des alternatives en fonction de l'état actuel de l'environnement. Le scénario est ainsi plus adaptatif.

4.5.3 Types

Dans une procédure de maintenance, certaines actions sont à répéter plusieurs fois mais avec des instances d'objet différentes. Il peut alors être fastidieux de décrire toutes les combinaisons possibles pour enchaîner ces actions. Par exemple s'il faut planter 10 clous dans une planche et qu'on en a 50 à disposition, peu importe les instances des clous utilisées et l'ordre dans lequel ces clous vont être plantés. Il faudrait alors faire différents embranchements avec les instances possibles à utiliser : cela serait extrêmement long et fastidieux et sans grand intérêt. Pour simplifier le scénario, nous avons donc décidé de regrouper les objets par catégories en leur attribuant un type. Dans le scénario il sera alors possible de préciser non plus l'instance d'un objet à utiliser mais son type. Dans notre exemple tous les clous de notre boite de clou seront du type clou. Il suffira alors d'enchaîner (ou de mettre en parallèle) 10 actions qui prendront toutes comme paramètre un marteau et un clou.

Par ailleurs l'utilisation des types est nécessaire pour certaines actions. Par exemple si l'on veut décrire l'action tenir le clou. Les paramètres sont donc le clou et la main. Mais lors de l'écriture de cette action on ne sait pas quel est l'acteur qui va réaliser cette action et donc quelle sera l'instance de main qui va servir pour tenir le clou. Il est donc obligatoire d'utiliser des types et non des instances dans ce cas là.

Les types sont précisés lors de la description d'un objet servant à initialiser le moteur d'interaction, en plus de ces capacités. Un objet peut avoir plusieurs types. Les types peuvent ensuite être utilisés dans la description du scénario pour un paramètre d'une action mais également dans une pré ou une post condition de cette action. Dans le scénario il n'y a pas de distinction sur l'utilisation des types ou des instances. C'est ensuite lors de l'analyse de l'action qu'on cherchera d'abord si un objet porte ce nom, et si ce n'est pas le cas, c'est qu'il s'agit d'un type.

A la réalisation du scénario, il est facile de vérifier que l'action demandée par l'apprenant correspond bien à l'action décrite dans le scénario (vérifier que si l'apprenant veut planter le clou3 cela correspond bien à l'action courante du scénario qui spécifie que l'apprenant doit planter un clou) mais il est plus difficile de déterminer l'instance de clou à utiliser à partir de l'action du scénario : si l'humain virtuel veut réaliser cette action il va devoir déterminer quelle instance de clou utiliser. Pour cela le système va passer en revue les différents objets du bon type. La première chose à vérifier est que cette instance peut effectivement réaliser l'action (par exemple un clou déjà planté ne peut plus l'être à nouveau). Ensuite, suivant les actions, il sera préférable d'utiliser une instance plutôt qu'une autre. Par exemple pour prendre un clou il vaut mieux prendre un clou qui n'a pas de possesseur. Si ce n'est pas possible on pourra, en dernier recours, voler un clou à quelqu'un. En revanche, pour l'action *poser un clou*, il faut poser un clou que l'acteur possède.

4.5.4 Synthèse

Nous avons profité du fait de proposer une extension multi-utilisateur au scénario pour simplifier son écriture. Il n'est désormais plus utile de préciser les actions de prise et de pose d'objets dans le scénario. A la place nous avons introduit des pré et post conditions pour les actions du scénario qui décrivent l'état des mains de l'acteur. Nous avons aussi proposé d'introduire des préconditions sur l'état d'une relation STORM qui permettent ainsi de spécifier des alternatives dans le scénario en fonction de l'état du monde. Nous avons également introduit un typage des objets. Pour paramétrer les actions du scénario il est maintenant possible d'utiliser ces types plutôt que de préciser l'instance d'un objet. La procédure devient ainsi plus simple à retranscrire pour l'auteur de scénario.

4.6 Gestion des situations de blocage

Nous avons vu dans le chapitre précédent que les ressources d'un acteur peuvent être dans un état *occupée* et ne sont alors plus disponibles pour interagir. Le scénario peut ainsi conduire un acteur à une situation de blocage si toutes ses ressources sont occupées et que les actions lui permettant de libérer ses ressources ne sont pas accessibles. Il est donc intéressant de repérer les actions du scénario qui modifient la disponibilité d'une ressource car elles peuvent être la source d'un blocage (section 4.6.1). De manière similaire à la gestion des interblocages dans les systèmes distribués, nous pouvons adopter deux stratégies pour remédier à ces situations de blocage : la prévention ou la guérison. La stratégie de prévention consiste à analyser le scénario hors ligne afin de détecter ces situations pour les éviter ensuite lors de l'exécution de

la procédure (section 4.6.2). La stratégie de guérison consiste à inciter un autre acteur à régler le problème (section 4.6.3).

Le choix de la stratégie à adopter pour gérer les situations de blocage est effectué en paramétrant le mécanisme de sélection d'action (voir chapitre 5). En effet, dans le module de répartition des actions, il est possible de mettre en place des critères pour indiquer qu'une action va conduire vers un blocage (stratégie de prévention) ou qu'une action va permettre de débloquer quelqu'un (stratégie de guérison). Ces critères seront ensuite exploités par le mécanisme décisionnel des acteurs. Il faut donc définir dans le profil collaboratif des acteurs des règles permettant de gérer ces situations de blocage (ne pas sélectionner une action menant vers un blocage pour la stratégie de prévention, sélectionner une action permettant de débloquer quelqu'un pour une stratégie de guérison).

4.6.1 Actions qui modifient la disponibilité des ressources

Certaines actions du scénario sont plus particulièrement susceptibles d'amener l'acteur vers un état de blocage où il ne pourra plus réaliser aucune action du scénario. C'est le cas pour les actions qui rendent une ressource occupée. Une action a est dite occupante si $\exists i|(precond(a)[i] \neq occupe) \wedge (postcond(a)[i] = occupe)$, avec i l'indice d'une pré et post condition pour l'action a ($i \in \{0,1\}$) ; à l'inverse une action est dite libératrice si $\exists i|(precond(a)[i] = occupe) \wedge (postcond(a)[i] \neq occupe)$

4.6.2 Stratégie de prévention : analyse hors ligne du scénario

Les modèles que nous avons mis en place (notamment les pré et post conditions) permettraient d'effectuer des raisonnements hors ligne sur le scénario qui pourraient ensuite être exploités en ligne. Nous pourrions par exemple nous intéresser aux actions occupantes afin de prévenir en ligne une situation de blocage. Pour cela il faudrait considérer une portion de scénario entre une action occupante et l'action libératrice associée. Pour chaque rôle pouvant réaliser l'action occupante, il faudrait ensuite extraire les actions de cette portion de scénario devant impérativement être réalisées par ce rôle. Il serait ainsi possible d'en déduire le nombre de ressources libres requises initialement pour que l'acteur jouant ce rôle puisse réaliser cette portion du scénario. S'il en faut plus de deux, cela signifie que pour que la portion de scénario soit réalisable par ce rôle il faut plusieurs personnes jouant le même rôle. A l'exécution de la procédure nous pourrions alors vérifier le nombre de personnes jouant ce rôle. S'il n'y en avait qu'une cela signifierait que la portion de scénario n'est pas faisable par l'acteur ayant ce rôle et nous pourrions alors l'empêcher de s'engager dans cette voie qui le conduirait inévitablement à une situation de blocage.

4.6.3 Stratégie de guérison : des humains virtuels capables de collaboration implicite

Il n'est pas toujours possible ni souhaitable d'éviter les situations de blocage. Lorsqu'un acteur se retrouve bloqué, il faut donc envisager des solutions pour le débloquer. La première méthode que nous proposons consiste à utiliser les autres acteurs qui réalisent le scénario. Ces

derniers sont capables de détecter qu'un de leur coéquipier est bloqué et ils peuvent alors tenter de l'aider si leurs profils collaboratifs les y incitent.

Pour cela, nous allons tenter de deviner les actions à effectuer pour faire progresser le scénario jusqu'à ce que l'acteur bloqué puisse libérer des ressources. Nous allons pour cela utiliser la notion de branche. En effet le scénario est constitué de plusieurs embranchements qui représentent diverses séquences d'actions pouvant être réalisées en parallèle (pour traduire des ET ou des OU). Cette notion de branche peut alors être utilisée pour donner une hiérarchie au scénario et ainsi évaluer la proximité de plusieurs séquences d'actions. Nous proposons alors d'utiliser une convention de nommage pour repérer plus facilement les différentes branches. Chaque action du scénario sera ainsi préfixée par la description de sa branche. Le scénario commence à la branche 1. A chaque nouvel embranchement, on ajoute à ce préfixe une sous-numérotation, par exemple au premier embranchement les branches seront nommées 1.1, 1.2, 1.3 etc. A chaque début d'embranchement on ajoute ainsi un point suivi du numéro de l'embranchement et à chaque regroupement on enlève le dernier chiffre et le dernier point.

Il est donc possible d'évaluer la proximité de plusieurs actions dans le scénario en comparant le nom des branches auxquelles elles appartiennent. Plus le préfixe commun est long plus les actions sont proches dans le scénario.

Lorsqu'un acteur est bloqué, c'est à dire lorsqu'il ne peut plus réaliser aucune action du scénario, on va alors regarder la dernière action du scénario qu'il a réalisé. On va alors chercher parmi tous les actions actives celle qui a la branche la plus proche. On supposera alors que cette action est celle qui a le plus de chance de débloquer l'acteur bloqué. On pourra alors inciter les acteurs à choisir cette action pour débloquer l'acteur bloqué.

Une autre solution de guérison proposée est de faire apparaître un humain virtuel nommé Merlin qui sort de nulle part et tente de débloquer l'acteur bloqué. Une fois ce dernier débloqué Merlin repart, comme par enchantement (voir section 5.4.2).

4.7 Synthèse

Nous proposons donc plusieurs niveaux d'introduction de la collaboration : un niveau global, celui du scénario, avec une synchronisation entre les différents acteurs ; un niveau intermédiaire où grâce à des actions de communication des enchaînements collaboratifs action/réaction apparaissent ; et enfin un niveau fin avec des actions à réaliser à plusieurs. Pour toutes ces formes de collaboration, il n'y a aucune distinction faite sur la nature du collaborateur, la description de la collaboration reste la même que l'acteur soit réel ou virtuel. De plus nous proposons de décrire le scénario collaboratif de manière à le rendre plus adaptable en terme de répartition des actions grâce à une utilisation différente du rôle et plus adaptable au contexte grâce à notre gestion de l'implicite. Nous avons aussi proposé différentes stratégies pour remédier à des situations de blocage : une stratégie de prévention qui nécessite une analyse hors ligne du scénario et une stratégie de guérison qui met à contribution les autres acteurs.

Ces changements impliquent de spécifier pour les actions du scénario les rôles autorisés ou le fil d'activité à utiliser ainsi que les pré et post conditions sur l'état des ressources de l'acteur. Nous avons proposé LORA++, fondé sur LORA, qui intègre ces nouvelles notions. L'originalité de ce travail est de proposer des procédures collaboratives où l'acteur qui va réaliser une

action donnée n'est déterminé qu'au moment de faire l'action en fonction de critères évalués dynamiquement. Il n'est donc plus indispensable de fixer à priori la répartition des actions entre les acteurs. Cette liberté apporte des propriétés intéressantes : plus de flexibilité sur la répartition des tâches tout en gardant un séquencement strict des actions, apparition de collaboration implicite, possibilité de se servir de GVT pour mettre au point itérativement une procédure. Mais ces modifications nécessitent également de mettre en place un mécanisme capable de tirer partie de cette flexibilité nouvelle et que nous allons décrire dans le chapitre suivant.

Chapitre 5

Mécanisme de sélection d'actions

Nous avons décrit le modèle de l'activité d'un acteur dans le chapitre 3 puis le scénario collaboratif dans le chapitre 4. Dans ce chapitre nous allons présenter le mécanisme qui permet de faire le lien entre l'activité des acteurs et le scénario. En effet, les acteurs jouent des rôles et ces rôles sont utilisés dans la description du scénario, les acteurs ont des capacités et ces capacités sont nécessaires pour réaliser les actions du scénario. Enfin, les acteurs gèrent des ressources qui possèdent un état courant et les actions du scénario possèdent des préconditions qui décrivent des états requis pour ces ressources. Tous ces éléments vont donc servir à faire le lien entre les acteurs et le scénario. De plus nous avons vu qu'il est possible d'assigner plusieurs rôles à une action du scénario ; plusieurs acteurs vont ainsi pouvoir réaliser une même action et il va falloir déterminer qui devrait la réaliser. Le rôle du mécanisme de sélection d'action va alors être de filtrer et d'analyser les différentes actions courantes du scénario et de transmettre cette analyse aux acteurs. Les acteurs vont alors pouvoir se baser sur cette analyse pour choisir l'action qu'ils vont effectuer. Notre objectif est, en effet, de donner une certaine autonomie aux humains virtuels sur leurs choix d'action mais nous voulons que ce choix soit guidé par la connaissance de ce qui serait le plus bénéfique pour faire avancer la procédure (éventuellement pour mieux pouvoir l'enfreindre).

Nous allons commencer par décrire dans la section 5.1 le fonctionnement du mécanisme de sélection d'action (présenté également dans [GA08]), puis nous présenterons un exemple pour illustrer son utilisation dans la section 5.2. Nous décrirons ensuite les avantages de ce mécanisme (section 5.3), l'utilisation pédagogique que l'on peut en faire (section 5.4), la gestion des conflits (section 5.5) et pour finir les extensions possibles pour ce mécanisme (section 5.6).

5.1 Fonctionnement

Nous proposons un mécanisme de sélection d'action [GA08] dont l'objectif est double : d'une part, permettre à un humain virtuel de décider de la prochaine action à réaliser et, d'autre part, pouvoir conseiller un utilisateur réel sur son choix d'action. Pour prendre cette décision un acteur se base sur l'état actuel du scénario avec un certain nombre d'actions possibles qu'il doit partager avec ses partenaires en fonction des rôles de chacun notamment. La première étape (décrite dans la section 5.1.1) du mécanisme de sélection d'action consiste donc à ana-

lyser chacune des actions possibles du scénario pour déterminer le meilleur candidat. Cette partie est mutualisable et nous avons donc choisi de la réaliser dans un module centralisé. Le résultat est une proposition de répartition des actions, la meilleure du point de vue du scénario (la répartition qui permet de faire avancer la procédure de façon optimale). La seconde étape (décrite dans la section 5.1.2) consiste, pour chaque acteur, à sélectionner une action. Ce choix sera guidé par son profil collaboratif qui va par exemple l'inciter ou non à suivre la suggestion du module de répartition. La dernière étape (décrite dans la section 5.1.3) n'est pas consécutive aux étapes précédentes ; elle a lieu lorsqu'un humain virtuel décide de réaliser une action et ne concerne donc qu'un seul humain virtuel. Cette étape consiste à réaliser l'action sélectionnée dans la deuxième étape si elle est directement réalisable ou une action implicite requise au préalable. La figure 5.1 donne une vision synthétique du mécanisme de sélection d'action.

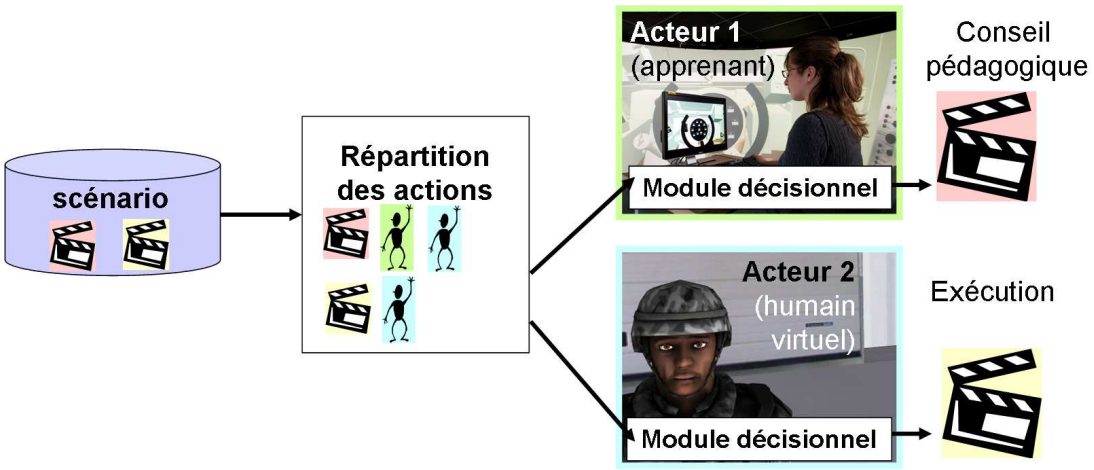


FIG. 5.1 – Mécanisme de sélection d'action

5.1.1 Répartition des actions

5.1.1.1 Principe

Dans les EVFC rencontrés dans la littérature un seul rôle est associé à une action du scénario ; chaque acteur sait donc, à tout moment, les actions qu'il doit réaliser. Nous avons proposé d'associer plusieurs rôles à une action du scénario, les acteurs ne sauront donc pas automatiquement s'ils doivent réaliser une action du scénario. Ils vont donc devoir se comparer aux autres. Nous proposons un module capable d'effectuer cette analyse pour pouvoir ensuite la transmettre aux acteurs et en particulier aux humains virtuels. Nous allons donc fournir aux acteurs un module qui va servir d'intermédiaire entre eux et le scénario et qui va analyser pour eux les actions du scénario et déterminer le meilleur candidat pour chaque action.

La première brique du mécanisme de sélection d'action consiste donc en un module global de répartition des actions du scénario entre les différents acteurs. Son rôle consiste à analyser les actions actives du scénario (c'est à dire autorisées à un instant donné) pour évaluer le

coût de chacune de ces actions pour chaque acteur et ainsi effectuer un classement des acteurs pouvant la réaliser. Le mécanisme de répartition comporte quatre étapes que nous allons décrire maintenant : l'évaluation de critères, le calcul d'un score, le classement des acteurs et la transmission des informations aux acteurs.

1. **Evaluation de critères.** Le module commence par récupérer auprès du moteur de scénario toutes les actions actives. Soit a une action active du scénario, h un humanoïde ayant un rôle autorisé pour l'action a, on évalue alors $V_c(a,h)$, c'est à dire la valeur du critère c pour ce couple. Les critères disponibles actuellement sont détaillés dans la section 5.1.1.2.
2. **Calcul d'un score.** Un score est ensuite calculé pour chaque couple action, humanoïde : $S(a,h) = \sum_c V_c(a,h) \times P_c$, avec $S(a,h)$ le score de l'humanoïde h pour l'action a et P_c le poids attribué au critère c. Certains critères peuvent ne pas être pris en compte pour le calcul du score et servir uniquement lors de la phase de prise de décision (voir section 5.1.2). Le score calculé ici représente une mesure de la pertinence pour l'humanoïde h de réaliser l'action a.
3. **Classement.** Pour chaque action possible, nous effectuons un classement des acteurs en fonction de leur score. Il est possible de choisir si on autorise ou non d'avoir plusieurs acteurs ex æquo.
4. **Transmission.** La dernière étape consiste à envoyer à chaque acteur les actions qu'il est autorisé à faire, son score, son classement ainsi que la valeur des différents critères évalués.

Ce module dispose de toutes les actions actives fournies par le moteur de scénario, il connaît le rôle de chacun des acteurs et peut communiquer avec le moteur d'interaction et avec les différents acteurs. Ce module conserve également un historique sur les actions réalisées par les différents acteurs.

Une étape intermédiaire du mécanisme de répartition consiste à chercher les acteurs qui ne peuvent réaliser aucune action. Ces acteurs sont alors considérés comme bloqués. Pour chaque acteur bloqué on cherche la dernière action qu'il a réalisée (mémorisée dans un historique) et on regarde la branche du scénario à laquelle appartient cette action. On cherche ensuite l'action active qui est la plus proche possible de cette branche et qui peut être réalisée par un autre acteur. Si on trouve une telle action on ajoute une information à cette action pour indiquer qu'elle est susceptible de débloquer un acteur. Cette information pourra ensuite être utilisée par le module de prise de décision d'un acteur.

Le mécanisme de répartition des actions est relancé complètement à chaque évolution du moteur de scénario. En revanche lors d'une action hors scénario (action hors procédure ou action de prise ou pose d'objet par exemple) la répartition n'est que partiellement relancée et seuls les critères susceptibles d'avoir changé sont réévalués.

Le module de répartition des actions va permettre à un humain virtuel de se positionner par rapport aux autres acteurs. Nous supposons qu'un utilisateur réel peut connaître l'état des autres acteurs, les rôles qu'ils jouent, les actions précédentes réalisées par chacun et qu'ils possèdent tous une parfaite connaissance de la procédure. L'utilisateur est ainsi lui aussi capable de se situer par rapport aux autres acteurs, de déterminer s'il est le meilleur candidat pour réaliser une action donnée du scénario ; il peut donc évaluer, en fonction de la situation, s'il est préférable

que ce soit lui qui réalise l'action ou s'il devrait laisser un autre la faire. Nous considérons donc qu'un utilisateur réel est capable d'effectuer l'analyse effectuée par le module de répartition des actions. En pratique il se peut que l'utilisateur réel ne sache pas exactement se situer par rapport aux autres, il pourra alors demander une aide pédagogique.

Le rôle de ce module est de fournir une suggestion de répartition des actions du scénario entre les différents acteurs. Ce n'est qu'une proposition qui représente la répartition que l'on considère comme optimale en fonction des critères évalués afin de faire avancer au mieux la procédure. Une fois cette suggestion envoyée à chaque acteur, les modules de prise de décision de chaque acteur prennent le relais (cf. section 5.1.2).

5.1.1.2 Paramétrage : critères pour la répartition

Avant de passer en revue les différents critères à évaluer, le mécanisme de répartition effectue une première sélection des candidats et en élimine certains grâce à plusieurs filtres. Un premier filtre concerne l'adéquation de l'acteur : si l'acteur n'est pas associé à l'action, que ce soit grâce à son rôle ou par le biais d'un fil d'activité, nous n'allons pas considérer l'acteur en tant que candidat pour l'action. Le second filtre concerne les capacités de l'acteur : s'il n'a pas les capacités requises pour réaliser l'action il ne sera pas non plus considéré.

Actuellement les critères évalués sont les suivants :

- **faisabilité immédiate** : ce critère indique si l'acteur peut d'ores et déjà réaliser l'action. Pour évaluer ce critère le module de répartition des actions va faire une requête auprès du moteur d'interaction pour savoir si l'action est actuellement possible.
- **facilité** : ce critère indique le nombre d'actions de prise et de pose d'objets nécessaires. Pour évaluer ce critère le module de répartition des actions va faire appel au gestionnaire de ressources de l'acteur qui va lui fournir les informations nécessaires.
- **disponibilité** : ce critère indique si les outils nécessaires à l'action et non encore possédés par l'acteur sont disponibles (s'ils ne sont pas en la possession de quelqu'un d'autre). Pour évaluer ce critère le module de répartition des actions va faire appel au moteur d'interaction pour déterminer si les objets requis ont un possesseur (s'ils sont impliqués dans une relation de saisie).
- **priorité** : ce critère indique si l'acteur joue un rôle considéré comme prioritaire pour réaliser l'action, ou alors secondaire.
- **continuité niveau plan d'action** : ce critère indique que l'acteur vient de prendre un objet requis pour l'action. Pour cela il suffit de regarder si la dernière action réalisée par l'acteur est la prise d'un objet requis par l'action.
- **continuité niveau branche de scénario** : ce critère indique la proximité entre la branche de la précédente action réalisée par l'acteur et celle de l'action considérée. Le module de répartition se sert pour évaluer ce critère de l'historique associé à l'acteur et compare la branche de la dernière action et la branche de l'action considérée.
- **occupation** : ce critère indique si l'action occupe des ressources, en libère une, ou si elle n'a pas d'influence sur l'état des ressources.
- **déblocage** : ce critère indique que l'action est susceptible de débloquer un acteur.

Tous les critères évalués ne sont pas forcément pris en compte dans le calcul du score : seuls ceux qui ont un poids non nul sont utilisés. Les critères non utilisés pour le calcul du score sont

simplement évalués dans le but d'être utilisés par le module décisionnel des acteurs ; c'est par exemple le cas actuellement pour les critères occupation et déblocage. Pour paramétrer ensuite le mécanisme de répartition d'action il faut donc attribuer des poids aux différents critères proposés. Il faut également traduire toutes les valeurs symboliques possibles des différents critères (qui peuvent être un booléen, un chiffre, etc) en une valeur numérique qui va servir à calculer le score de l'acteur.

Il est bien sûr possible de rajouter des critères à évaluer comme par exemple la proximité entre l'acteur et l'objet avec lequel il doit interagir.

5.1.2 Prise de décision

5.1.2.1 Principe

Comme nous venons de le voir, le module de répartition des actions envoie une suggestion de répartition à chaque acteur. Cette suggestion sert de base aux humains virtuels pour choisir la prochaine action à réaliser et peut aussi être utilisée afin de donner aux utilisateurs réels des conseils pédagogiques sur la prochaine action à réaliser ou afin de leur signifier si leur choix d'action était judicieux ou non. Le module de répartition des actions est donc complété par un mécanisme de prise de décision qui permet à chaque acteur de sélectionner la prochaine action à réaliser, en s'appuyant sur cette proposition de répartition et en se basant sur son profil collaboratif. Ainsi, un humain virtuel pourra très bien décider, si son profil collaboratif le conditionne à le faire, d'aller contre la proposition du module de répartition dans le but de perturber l'apprenant par exemple. Rappelons que le profil collaboratif est défini par un ensemble de règles de sélection d'actions auxquelles on associe un poids (voir section 3.3).

Le module de prise de décision prend en entrée des actions que l'acteur pourrait réaliser. Pour chaque action considérée, un score est calculé : $S(a) = \sum_r V_r(a) \times P(r)$ avec $S(a)$ le score de l'action a, r une règle du profil collaboratif, $V_r(a) = \{0,1\}$ la valeur de la règle r pour l'action a, et $P(r)$ le poids de la règle r défini dans le profil. Toutes les règles du profil collaboratif de l'acteur sont ainsi appliquées successivement et si l'action vérifie la règle (si $V_r(a) = 1$), alors son score est incrémenté du poids de cette règle. L'action sélectionnée sera ensuite celle ayant le score maximal. En cas d'égalité, actuellement c'est la première action trouvée qui sera sélectionnée, toutefois il est possible d'effectuer plutôt un choix aléatoire.

Pour le moment, les actions considérées dans ce module sont uniquement les actions du scénario permises pour l'acteur. Seules les actions provenant du modèle de répartition peuvent donc être sélectionnées. Il est néanmoins possible d'ajouter d'autres actions à prendre en compte comme des actions d'un type particulier (ex : actions perturbatrices) ou des demandes pédagogiques.

La partie prise de décision est lancée pour chaque acteur dès qu'il reçoit une nouvelle répartition c'est à dire dès qu'une action a été effectuée dans l'environnement virtuel et que le mécanisme de répartition a été lancé. Chaque acteur sélectionne donc une action puisqu'on ne connaît pas à ce stade l'acteur qui sera le prochain à agir. Dans la section 5.5 nous discuterons des conflits qui peuvent intervenir si plusieurs acteurs choisissent la même action.

5.1.2.2 Paramétrage : règles pour définir le profil collaboratif

Comme nous l'avons vu dans la section 3.3, un acteur dispose d'un profil collaboratif pour guider son choix d'action. Concrètement, il s'agit d'un ensemble de règles de sélection d'action qui sont activées. Pour le moment les règles prédéfinies proposées sont :

- **privilégier les actions où l'acteur est le mieux classé** : l'acteur est alors incité à suivre la répartition proposée par le module de répartition, puisqu'il va avoir tendance à ne sélectionner que les actions pour lesquelles il est considéré comme le meilleur candidat. Si tous les acteurs suivent cette règle, il y a peu de chance de voir des conflits sur le choix d'une action, sauf sur les actions où plusieurs acteurs sont classés premiers. En revanche si cette règle est la seule activée, il est possible qu'un acteur se retrouve avec aucune action à réaliser.
- **privilégier les actions pour lesquelles le score de l'acteur est maximal** : cette règle incite l'acteur à toujours choisir une action (sauf s'il ne peut réaliser aucune action) qui sera celle où il a le meilleur score. Cette règle peut provoquer des conflits sur le choix d'action (voir section 5.5) car un acteur pourra très bien choisir une action même s'il n'est pas le meilleur candidat, simplement parce que c'est la meilleure action pour lui ; en effet il est alors possible que l'acteur classé premier pour cette action décide également de réaliser cette action. Cette règle permet donc d'avoir des acteurs qui seront actifs dès qu'ils le pourront.
- **privilégier les actions pour lesquelles l'acteur a le meilleur score parmi les actions où il est classé premier** : cette règle se substitue aux deux précédentes et permet en général d'avoir une seule action privilégiée (à moins d'avoir deux actions pour lesquelles l'acteur est classé premier et a le même score).
- **si un acteur est bloqué, privilégier l'action suivante sur la branche qu'il avait entamée pour tenter de l'aider** : cette règle rend possible l'apparition de la collaboration implicite. Elle se base sur le critère déblocage renseigné par le module de répartition et qui indique si une action est susceptible de débloquer un acteur. Pour voir apparaître une collaboration implicite il faut donner à cette règle un poids plus fort que les autres afin que la ou les actions associées aient le score maximal.
- **si un acteur vient de prendre un objet, éviter les actions qui nécessitent de prendre à nouveau cet objet** : cette règle permet de détecter un conflit sur un outil. En effet, si un acteur vient de prendre un objet et qu'il doit à nouveau le prendre, c'est qu'un autre acteur lui a pris entre temps. Dans ce cas on incite l'acteur à choisir plutôt une autre action, en décrémentant le score associé aux actions qui nécessite la prise de l'outil source du conflit. Plus de détails sur la gestion des conflits seront données dans la section 5.5.
- **si un acteur n'a sélectionné aucune action et qu'il a un objet en main, poser cet objet** : cette règle permet à un acteur inactif de ne pas monopoliser un outil.

Nous ne proposons actuellement que quelques règles afin d'illustrer le fonctionnement de ce mécanisme, mais il est aisé de définir de nouvelles règles, qui vont par exemple se baser sur des critères fournis par le module de répartition. Les profils collaboratifs sont par conséquent extensibles. Par exemple, pour créer un profil de perturbateur, il va falloir créer et activer des règles du type *privilégier les actions hors scénario, privilégier les actions perturbatrices*,

privilégier les actions ne respectant pas la continuité, privilégier l'inaction, etc.

5.1.3 Réalisation de l'action

A chaque cycle du mécanisme de sélection d'action, un acteur va réaliser une action. Du point de vue global il y aura donc un seul acteur qui va passer par une phase de réalisation de l'action, les autres acteurs vont retourner à une phase de prise de décision sans avoir réaliser d'action. La phase de réalisation d'action sera différée par rapport aux deux phases précédentes (répartition et décision) puisqu'elle n'aura lieu que lorsqu'un acteur (réel ou virtuel) décidera d'agir.

Un utilisateur réel va réaliser une demande d'action en sélectionnant l'action qu'il souhaite réaliser. Un humain virtuel, tout comme un utilisateur réel, va réaliser des actions dans l'environnement. Il va donc falloir qu'il choisisse une action à effectuer. Cette sélection a été préparée par le module décisionnel. Quand arrive le moment d'effectuer une action, l'humain virtuel n'a plus qu'à traduire l'action du scénario qu'il vise en une action à réaliser qui peut être soit directement l'action du scénario visée soit une action préparatoire (action implicite). L'humain virtuel va donc devoir, le cas échéant, transformer l'action sélectionnée par son module décisionnel en une action réalisable. Deux cas se présentent alors :

- Soit l'action est directement réalisable : l'humain virtuel va alors signaler au moteur d'interaction qu'il souhaite réaliser cette action.
- Soit l'action sélectionnée requiert des actions intermédiaires : dans ce cas l'humain virtuel consulte la séquence d'actions implicites calculée par son module de gestion des ressources.
  - Si la première action prérequise est une action de pose : l'objet à poser est connu, il suffit de trouver un endroit où poser cet objet. Pour cela un paramètre indique l'endroit par défaut où poser les objets.
  - Si la première action prérequise est une action de prise d'objet : il se peut que plusieurs actions correspondent. En effet l'action prérequise peut être *prendre un clou*. Il va falloir dans ce cas choisir la main à utiliser (celle qui est libre) et l'objet à prendre puisqu'ici ce n'est pas une instance qui est précisée mais un type (on va alors chercher le premier objet qui possède le bon type et qui est disponible). On pourrait ici considérer plus de critères pour sélectionner l'objet comme par exemple sa proximité avec l'acteur.

Une fois l'action effective à réaliser choisie, on fait une demande auprès du moteur d'interaction. Le mécanisme de sélection d'action sera ensuite relancé pour un nouveau cycle.

5.1.4 Vue d'ensemble du mécanisme de sélection d'action

La phase de répartition des actions, globale, est immédiatement suivie par une phase décisionnelle pour chaque acteur. Chaque acteur sélectionne ainsi une action qu'il vise. Ces deux phases sont généralement suivies par une période d'attente où l'on attend qu'un acteur décide de réaliser une action. Si c'est un utilisateur alors le mécanisme de sélection d'action est relancé. Si c'est un humain virtuel alors une phase de réalisation de l'action est lancée qui lui permet de traduire une action visée en une action concrète à réaliser pour préparer cette action

visée. Le mécanisme de sélection d'action est alors là aussi relancé.

Le mécanisme de sélection d'action est paramétrable à chacune de ses étapes. Pour la partie répartition des actions, il est possible de prendre en compte des critères variés et de les pondérer de manière globale pour toute l'exécution du scénario. Il faut donc lors du paramétrage du module de répartition d'actions spécifier les critères à prendre en compte et donner un poids à chacun d'eux. Chaque humain virtuel peut aussi être paramétré individuellement afin de définir son profil collaboratif. Pour cela il suffit de spécifier les règles à prendre en compte pour sa phase de prise de décision, ainsi que le poids à attribuer à chacune d'elles. Pour les humains réels, il faut définir un profil collaboratif par défaut qui consiste à suivre la procédure. En effet, il ne faut pas perdre de vue l'usage c'est à dire la formation à la procédure. Ce profil sera utilisé pour sélectionner la prochaine action qui sera conseillée à l'utilisateur s'il demande de l'aide. Ce double paramétrage permet à la fois de paramétrer la répartition globale pour un scénario, en donnant plus d'importance à certains critères et également de raffiner ce paramétrage au niveau de chaque humain virtuel en le dotant d'un profil collaboratif propre.

Nous nous étions fixé comme objectif de pouvoir interchanger facilement humains virtuels et utilisateurs réels. La modélisation de leur activité est strictement identique, l'utilisateur réel a simplement un profil collaboratif standard qui va lui permettre d'obtenir des conseils pédagogiques (ou de savoir si son choix d'action est judicieux ou non). La partie prise de décision sera alors opérationnelle de la même manière que pour un humain virtuel, mis à part que le résultat de cette décision pourra être divulgué en tant que conseil si l'utilisateur en fait la demande, mais l'action ne sera réalisée qu'à la demande de l'utilisateur, alors que pour l'humain virtuel la réalisation est automatique. C'est là la seule différence entre les utilisateurs réels et les humains virtuels. La demande d'action, qu'elle provienne d'un utilisateur ou d'un humain virtuel, passe ensuite par le même canal et son traitement sera donc identique. Un utilisateur peut donc très facilement être remplacé par un humain virtuel dynamiquement, et inversement, de manière tout a fait transparente pour les autres utilisateurs. On peut se référer à la figure 5.1 pour avoir une vision synthétique du mécanisme de sélection d'action.

5.2 Exemple

Afin d'illustrer le fonctionnement du mécanisme de sélection d'action et également les notions d'adaptation au contexte et de collaboration implicite évoquées dans les sections précédentes, nous allons examiner le déroulement d'un scénario court sous différentes conditions. Cet exemple va ainsi permettre de montrer la diversité des exécutions d'une procédure due à la modification de quelques paramètres.

La procédure à réaliser est simple et consiste en diverses actions de bricolage. La procédure est composée de deux branches à réaliser en parallèle. La première consiste à retirer la poignée d'une porte de placard. L'acteur doit donc tenir la poignée avec une main pendant qu'il dévisse la vis qui retient la poignée avec un tournevis. Ensuite il peut retirer la vis puis déposer la poignée. La deuxième branche consiste à planter 4 clous dans une planche. Chaque branche doit normalement être réalisée par une seule personne. On utilise donc des fils d'activité, un pour chaque branche, en mode préférentiel. Pour la première action de la branche 1 le rôle *ouvrier 1* est prioritaire et le rôle *any* secondaire alors que pour la branche 2 c'est le rôle

ouvrier 2 qui est prioritaire et le rôle *any* est là aussi secondaire. Deux humains virtuels, HV1 et HV2, jouent respectivement les rôles de *ouvrier 1* et *ouvrier 2*.

5.2.1 Déroulement normal - règle de sélection simple

Score pour la répartition : priorité du rôle comme critère principal.
Règle du profil collaboratif :
-Privilégier les actions pour lesquelles l'acteur a le meilleur score parmi les actions où il est classé premier.

HV1 réalise les actions de la première branche, tandis que HV2 réalise parallèlement les actions de la deuxième branche. Tout se déroule normalement comme l'illustre la figure 5.2. Sur cette figure les actions en bleu sont les actions de la branche 1, les actions en orange celles de la branche 2 et les actions en noir les actions qui ne figurent pas dans le scénario (actions implicites).

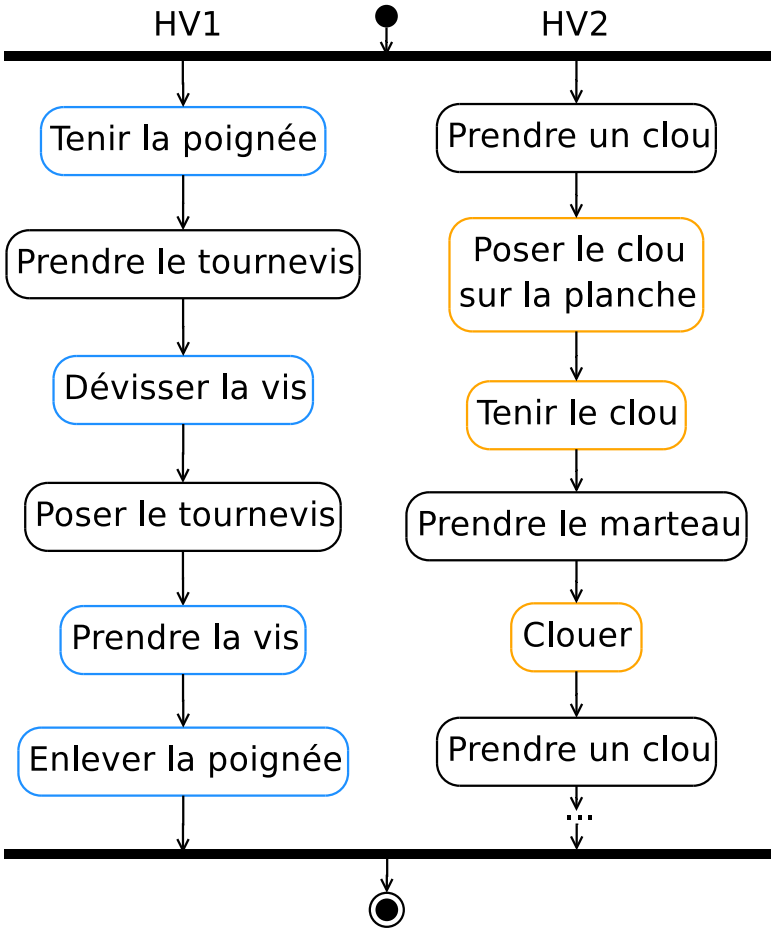


FIG. 5.2 – Diagramme d'activité - déroulement normal

5.2.2 Déroulement dans le noir - règle de sélection simple

Score pour la répartition : inchangé.
Règle du profil collaboratif : inchangée.

Imaginons maintenant que l'on change les conditions de réalisation de la procédure, qui doit désormais être réalisée dans le noir. L'humain virtuel ayant pour rôle *ouvrier 1* tient donc une lampe de poche dans une main et n'est pas autorisé à la lâcher : HV1 a donc initialement une main dans un état *occupée*.

HV1 commence par tenir la poignée et a alors ses deux mains occupées : l'une tient la lampe de poche et l'autre la poignée. Il ne peut plus continuer sa branche et est donc bloqué. HV2 est alors classé premier à la fois pour l'action suivante de sa branche mais également pour l'action suivante de la branche 1 (l'action *dévisser la vis*) puisque HV1 ne peut pas la réaliser. Mais le score de HV2 pour les actions de sa branche est plus élevé puisqu'il est prioritaire pour ces actions. HV2 va donc poursuivre sa branche jusqu'au bout. Lorsqu'il aura terminé, alors il ne lui restera plus que l'action de la branche 1 à réaliser et il va donc la sélectionner. HV2 va donc prendre le tournevis, dévisser la vis puis la retirer et HV1 sera alors débloqué.

HV1 est donc resté bloqué un long moment sans rien pouvoir faire. HV2 ne lui est venu en aide que lorsqu'il n'avait plus d'autres actions à réaliser, ce qui aurait pu prendre très longtemps dans un scénario plus long.

5.2.3 Déroulement dans le noir - règles de sélection plus évoluées

Score pour la répartition : inchangé.
Règles du profil collaboratif, par importance décroissante :

-Si un acteur est bloqué, privilégier l'action suivante sur la branche qu'il avait entamée pour tenter de l'aider.

-Privilégier les actions pour lesquelles l'acteur a le meilleur score parmi les actions où il est classé premier.

Le déroulement de la procédure reste le même jusqu'à ce que HV1 soit bloqué avec une poignée et une lampe de poche en main. Là, HV2 détecte ce blocage, abandonne ce qu'il était en train de faire et vient tenter d'aider HV1 en prenant le tournevis puis en réalisant l'action qui suit sur la branche 1 : dévisser la vis (voir figure 5.3). HV1 est toujours bloqué, HV2 continue donc en prenant la vis. HV1 peut maintenant réaliser l'action déposer la poignée et libérer une main. HV2 va alors poser la vis et reprendre la branche 2 là où il l'avait laissée. HV2 est donc venu spontanément aidé HV1, sans que cela soit écrit dans la procédure (nous avions juste utilisé un fil d'activité en mode préférentiel) c'est ce qu'on appelle la **collaboration implicite**.

5.2.4 Conclusions tirées de l'exemple

Cet exemple montre que la même procédure peut avoir des déroulements très différents en fonction du contexte et des règles de sélection d'action appliquées. Il est également possible de jouer sur le poids attribué aux différents critères par le module de répartition. Le mécanisme de sélection d'action permet donc d'adapter une procédure à un contexte de réalisation différent, lui donnant plus de souplesse. Une collaboration implicite peut alors apparaître, mais il est également possible de s'en servir pour mettre au point une procédure en écrivant cette procédure

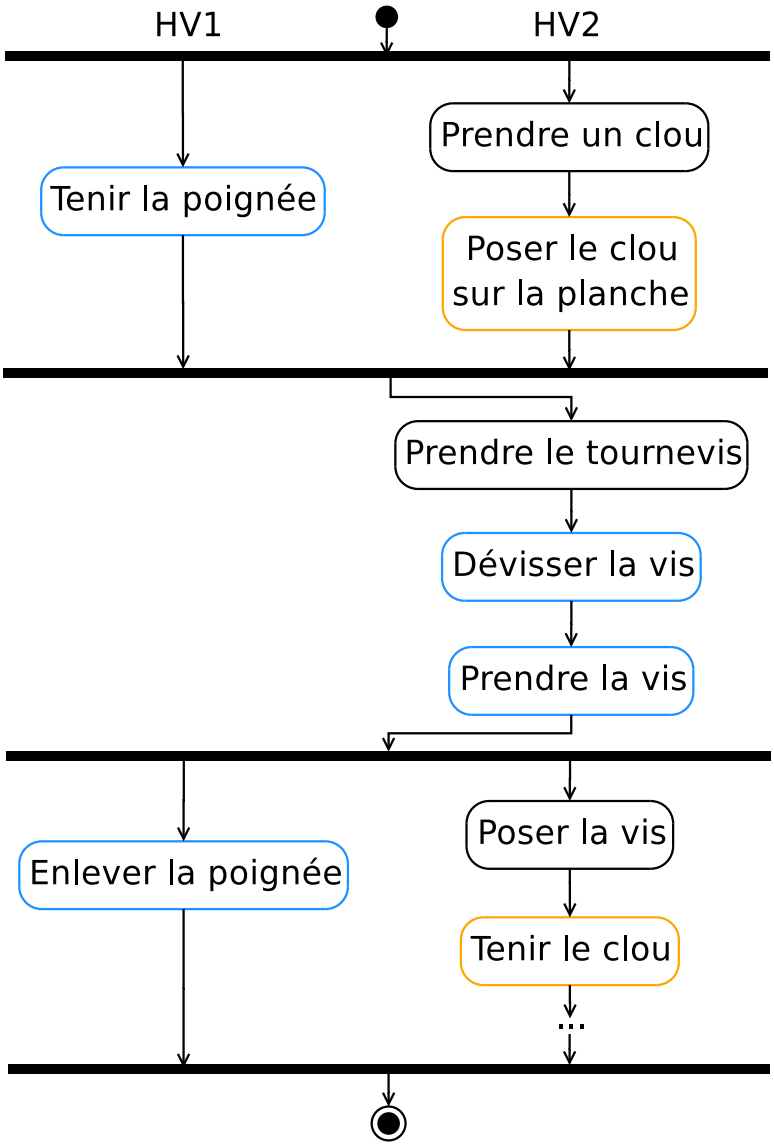


FIG. 5.3 – Diagramme d'activité - déroulement dans le noir avec collaboration implicite

sans assigner de rôle aux actions, en laissant plusieurs humains virtuels la réaliser et en voyant comment ils se sont répartis les tâches. Des retouches sont alors possibles de manière itérative en affinant la spécification des profils et des règles. Ces différentes utilisation du mécanisme de sélection d'action seront détaillées dans la section 5.3.

Dans cet exemple un humain virtuel aide spontanément l'autre qui est bloqué. Cette aide peut permettre à un apprenant novice de prendre conscience des erreurs qu'il a faites pour se retrouver bloqué, tout en lui permettant de poursuivre néanmoins la procédure. Il est bien entendu possible de désactiver cette aide en fonction de la stratégie pédagogique : il est ainsi possible d'imposer à un apprenant expérimenté de réaliser effectivement toutes les actions de la procédure qui sont de son ressort, pour cela il suffit d'adapter le profil collaboratif de ses partenaires virtuels. Le profil collaboratif des humains virtuels doit donc être adapté aux apprenants en fonction des attentes pédagogiques. Dans la section 5.4 nous allons détailler l'utilisation pédagogique que l'on peut faire du mécanisme de sélection d'action.

5.3 Avantages

Le mécanisme de sélection d'action permet de déterminer dynamiquement la personne qui va réaliser une action. Cette liberté sur l'assignation des personnes aux actions amène trois propriétés nouvelles dans les EVFC mais néanmoins fondamentales que nous allons maintenant détailler : l'adaptabilité (section 5.3.1), l'apparition de collaboration implicite (section 5.3.2) et la mise au point de procédures (section 5.3.3).

5.3.1 Adaptabilité

La procédure définie grâce aux rôles et aux fils d'activité garde un séquencement strict des actions, tout en permettant une souplesse quant à la répartition de celles-ci entre les acteurs participant au scénario. Ainsi la personne la mieux à même de réaliser une action donnée sera évaluée dynamiquement en fonction de critères paramétrables. Par exemple l'acteur qui réalisera l'action sera celui qui a déjà l'outil en main et qui est le plus près de l'objet avec lequel interagir. Le scénario sera donc plus optimisé (on pourra par exemple choisir de limiter les déplacements ou les échanges d'outils). De plus le scénario devient plus adaptatif à un contexte différent : si le nombre de personnes présentes change tout en gardant la même procédure, il sera possible de trouver une nouvelle répartition des actions. Il est également possible de mettre des contraintes supplémentaires pour voir la capacité d'adaptation des apprenants : un coéquipier qui ne peut pas se déplacer, un autre qui a une main occupée dont il ne peut plus se servir (il tient une lampe de poche et le scénario a lieu dans le noir ou il a une main blessée par exemple). Le scénario pouvant s'adapter, il devient possible d'enseigner aux apprenants cette capacité d'adaptation. Le fait de ne plus avoir à écrire les actions de prise et de pose d'objet contribue également à cette flexibilité du scénario puisqu'un apprenant peut conserver un outil pour une action future s'il le souhaite vu qu'on ne lui impose plus de le reposer systématiquement après chaque action.

5.3.2 Collaboration implicite

La procédure décrit la succession des actions, et donne des informations sur qui peut réaliser chaque action. Elle reste assez générique. Il est ensuite possible d'exploiter le mécanisme de sélection d'action pour créer des humains virtuels avec un profil collaboratif qui leur donne une propension à aider les autres. Ils vont alors se servir du mécanisme de sélection d'action pour détecter les acteurs qui sont en difficulté, c'est à dire qui sont bloqués et ne peuvent plus avancer dans le scénario. Il vont alors essayer de faire l'action suivante sur la branche entamée par l'acteur bloqué afin de tenter de le débloquer. Un humain virtuel pourra ainsi aider spontanément un apprenant novice pour débloquer ce dernier et lui permettre ainsi de poursuivre la procédure. C'est ce que nous nommons la collaboration implicite. Le mécanisme de sélection d'action permet en effet d'anticiper les actions possibles par les autres acteurs et il est donc possible de s'en servir pour mieux choisir l'action à réaliser.

5.3.3 Mise au point de procédures

Grâce à ce mécanisme, il est possible de mettre au point itérativement une procédure. Pour cela, il faut commencer par décrire les actions de la procédure à effectuer et leur agencement. Ensuite, il suffit de laisser des humains virtuels réaliser la procédure et d'observer l'influence de certains paramètres (nombre de personnes, outils disponibles, déplacement d'un objet, allocation des tâches, etc) sur le déroulement du scénario (répartition des tâches, les passages bloquants, etc). Si le concepteur du scénario souhaite optimiser les déplacements par exemple, il va donner beaucoup de poids au critère de proximité et ainsi les humains virtuels vont se déplacer le moins possible. Le concepteur pourra ensuite modifier la position de départ des humains virtuels ou de certains objets et voir l'influence sur le temps de réalisation de la procédure. Cette information sur le temps de réalisation ne pourra néanmoins servir qu'à comparer deux déroulements d'une même procédure puisqu'il n'y a pas de corrélation entre le temps de la procédure en virtuel et le temps mis pour faire la même procédure en réel.

5.4 Utilisation pédagogique

5.4.1 Adapter le comportement des coéquipiers

Le profil collaboratif doit être adapté en fonction de la finalité de la simulation : s'il s'agit d'une session de formation il faudra adapter les profils collaboratifs des humains virtuels en fonction du profil pédagogique de l'apprenant (novice, expert, etc). Si l'objectif est de mettre au point une procédure alors les humains virtuels auront des profils collaboratifs plus classiques leur indiquant de suivre la procédure. Certains profils vont favoriser les conflits sur les choix d'action d'autres vont avoir tendance à les éviter.

Le profil collaboratif permet d'adapter le comportement des humains virtuels à la situation pédagogique que l'on souhaite mettre en place. En effet, à des fins pédagogiques, il est intéressant d'avoir des humains virtuels avec des comportements différents et paramétrables. Le profil collaboratif d'un humain virtuel peut être défini de façon à lui faire jouer un rôle pédagogique particulier [Buc05]. Par exemple certaines règles pourraient permettre de créer un humain vir-

tuel jouant un rôle pédagogique de perturbateur. Cet humain virtuel choisirait alors des actions dans le but de déstabiliser l'apprenant. Mais grâce à la notion de profil, il est possible de créer, pour remplir un même rôle pédagogique, des profils avec certaines variantes. Deux humains virtuels jouant un rôle de perturbateur pourront ainsi avoir chacun leurs spécificités. L'un va par exemple réaliser des actions qui n'auront pas de lien avec la procédure et qui sont gênantes (comme déclencher une alarme). L'autre pourra plutôt prendre les outils dont l'apprenant a besoin. Ces deux humains virtuels auront donc le même rôle pédagogique (perturbateur) mais des profils collaboratifs différents (leurs règles de comportement seront différentes).

Il est donc important d'adapter les profils collaboratifs des humains virtuels à la situation de formation (en particulier le niveau pédagogique des apprenants) afin de profiter au mieux des possibilités collaboratives ou anti collaboratives que peuvent apporter les humains virtuels. En effet, un apprenant novice aura besoin de partenaires très coopératifs, pouvant même aller jusqu'à l'aider pour le débloquer alors qu'un apprenant expert pourra faire face à des perturbateurs ou des coéquipiers non coopératifs. La formation à un même scénario pourra ainsi être progressive en changeant les profils pédagogiques des partenaires de l'apprenant au fur et à mesure qu'il assimile la procédure.

5.4.2 Humain virtuel particulier : Merlin, le prof

Comme nous l'avons évoqué dans la section 4.6, dans notre scénario il peut arriver qu'un acteur se retrouve bloqué. Nous voulons alors proposer des solutions pour le débloquer pour ne pas avoir à recommencer la procédure. La première solution consiste à donner aux humains virtuels un profil collaboratif qui les incitent à débloquer un autre acteur s'ils le peuvent, pour cela une règle spécifique doit être activée (voir section 5.1.2.2). Cette solution est illustrée dans l'exemple de la section 5.2. Une autre solution (par exemple s'il n'y a pas d'humain virtuel qui réalise le scénario) consisterait à créer un humain virtuel particulier : Merlin, le prof. Merlin n'interviendrait que lorsqu'un acteur se retrouverait bloqué. Il apparaîtrait alors afin de réaliser les actions qu'il juge nécessaire pour débloquer l'acteur en difficulté. Puis il disparaîtrait, comme par enchantement. Merlin représente la métaphore du formateur ou du professeur de travaux pratiques qui vient débloquer un élève afin de lui permettre de poursuivre son TP. Merlin représente donc un joker qui vient aider quand aucune aide venant des acteurs déjà présents n'est possible. On pourrait attendre plusieurs cycles de sélection vide avant d'appeler Merlin pour voir si ce n'est pas juste un problème de ressources (outils par exemple) ou pour attendre de voir si un utilisateur ne va pas spontanément venir en aide à l'acteur bloqué.

5.4.3 Couplage avec le moteur pédagogique

Actuellement le moteur pédagogique ne communique pas avec les acteurs au niveau de leur module décisionnel. Pourtant il serait intéressant de les faire communiquer et de faire transiter des informations dans les deux sens. Le moteur pédagogique pourrait ainsi faire des suggestions d'actions auprès des humains virtuels (suite à une demande du formateur par exemple). Suite à une demande d'aide d'un apprenant, le moteur pédagogique pourrait aussi communiquer avec le module décisionnel de l'acteur concerné par cette demande d'aide et obtenir ainsi des informations sur l'action qu'il serait préférable de viser, les raisons de ce choix (valeur des

différents critères évalués) et les actions requises au préalable. La pédagogie serait ainsi à même de fournir plus d'informations à l'apprenant.

5.5 Gestion des conflits

Notre mécanisme de sélection d'action peut amener deux types de conflits : un conflit sur le choix d'action lorsque plusieurs acteurs visent la même action du scénario (section 5.5.1) et un conflit sur l'utilisation des outils ou objets de l'environnement (section 5.5.2).

Notre objectif n'est pas d'éviter systématiquement les conflits, en effet un humain virtuel perturbateur pourra au contraire les chercher. De plus, nous laissons aux utilisateurs réels la liberté sur leurs choix d'action, ils peuvent ainsi être à la source d'un conflit. Nous proposons simplement différentes méthodes pour gérer certains types de conflits et éviter qu'un humain virtuel soit la source d'un conflit non désiré. Le concepteur pourra ensuite choisir parmi les solutions proposées celles qui l'intéressent.

5.5.1 Conflits sur le choix d'action

En fonction du profil collaboratif défini pour les humains virtuels, il peut arriver que plusieurs humains virtuels décident de réaliser la même action. C'est pourquoi il est important de choisir des profils collaboratifs qui vont bien ensemble si l'on veut éviter les conflits sur les choix d'action. En effet, si un acteur se base sur son classement et qu'un autre ne se base que sur son score, les conflits seront alors plus fréquents. Par ailleurs un utilisateur réel et un humain virtuel peuvent viser la même action du scénario.

Si le conflit porte sur une action du scénario directement réalisable ce n'est pas gênant car ce sera l'acteur le plus rapide qui va alors réaliser cette action, les actions possibles du scénario vont ainsi changer et le conflit va disparaître. En effet il y a une atomicité dans la réalisation d'une action : une fois qu'un acteur a sélectionné une action, le processus de sélection d'action est bloqué pour les humains virtuels tant que la nouvelle répartition n'a pas eu lieu. Si le conflit porte sur une action qui nécessite des outils par exemple, cela peut être plus problématique. En effet, si chacun des acteurs veut le même outil et le vole constamment à l'autre, on arrive à une situation d'interblocage. Il en sera de même si chacun des deux acteurs a un des outils requis pour une action et refuse de s'en séparer. Le conflit sur le choix d'un action revient donc à un conflit sur l'utilisation d'un outil que l'on abordera dans la section suivante.

Toutefois, dans la plupart des cas les conflits sur les choix d'action s'effacent par eux même. En effet, imaginons que pour un cycle de sélection d'action donné deux humains virtuels soit classés premier ex æquo pour une action et la sélectionnent tous les deux. Le premier qui va décider de réaliser cette action va par exemple prendre un des outils requis. Dans le cycle de sélection d'action suivant cette action sera détectée comme dans la continuité de l'action qu'il vient d'effectuer (la prise d'outil). Il pourra de plus réaliser plus facilement l'action (car il possède déjà un outil) et aura donc un score plus élevé. L'acteur qui vient d'agir va ainsi prendre un avantage sur l'autre acteur, il aura un meilleur classement et cela peut suffire à faire disparaître le conflit.

De plus il est possible de mémoriser l'action visée par chacun des acteurs pour chaque cycle de sélection ce qui permettrait de repérer facilement les conflits. Pour les utilisateurs réels il

existe pour le moment un critère de continuité niveau action qui se base sur les actions de prise d'objet pour deviner si une action du scénario était visée. Il serait possible d'utiliser d'autres heuristiques pour tenter de mieux prédire les actions qu'un utilisateur réel envisage de faire.

Les conflits sur les choix d'action peuvent parfois se traduire par des actions inutiles. Par exemple, deux acteurs qui visent une action dévisser vont peut être chacun prendre un tournevis (s'il existe deux tournevis dans la scène) mais seul l'un va effectivement s'en servir. Mais cette situation fait juste perdre du temps à un acteur, elle ne provoque pas d'interblocage donc ce n'est pas trop gênant.

Pour éviter absolument les conflits il faut donc ne pas autoriser plusieurs acteurs ex æquo au niveau du module de répartition et activer pour tous les humains virtuels des règles qui ne sélectionnent qu'une action pour laquelle l'acteur est classé premier. Au contraire dans le but de perturber un apprenant expert par exemple, il est possible de faire en sorte qu'il y ait beaucoup de conflits sur les actions entre les participants au scénario ; pour cela on sélectionne des règles de sélection qui autorise à sélectionner une action pour laquelle l'acteur n'est pas classé premier.

5.5.2 Conflits sur les outils

Un conflit sur un outil a lieu si un acteur a besoin d'un outil pour l'action qu'il vise et que cet outil est déjà possédé par un autre acteur. Cette possibilité de conflit peut donc être détectée grâce au critère disponibilité évalué par le module de répartition des actions. Pour régler un conflit sur un outil, il faut qu'un acteur puisse demander ou prendre un outil à un autre. Pour cela nous devons déterminer si nous autorisons qu'un acteur prenne directement un outil à un d'autre (emprunt d'outil) ou si nous mettons en place une demande d'outil. La demande d'outil paraît une solution préférable mais elle peut amener à des situations d'interblocage. Imaginons que deux acteurs visent la même action et qu'ils possèdent chacun un des outils requis : aucun des deux acteurs ne va vouloir se séparer de son outil et la situation est donc bloquée. Nous avons donc choisi d'autoriser l'emprunt d'outil. Un conflit sur un outil peut alors se traduire par des emprunts d'outils successifs. Afin d'éviter qu'un outil passe alternativement d'un propriétaire à un autre, une règle de sélection d'action (décrite dans la section 5.1.2.2) permet à un humain virtuel de ne pas reprendre immédiatement un outil qu'il se serait fait voler. Il laisse ainsi à la personne qui a pris l'outil le temps de s'en servir.

Nous misons également sur le fait que l'acteur qui va acquérir un outil aura un meilleur score pour réaliser l'action qu'il vise et qu'il va donc prendre l'avantage sur les autres. De plus, les humains virtuels laissent un délai variable (aléatoire) entre la réalisation de deux actions. Il y a donc asymétrie entre les acteurs et il se peut qu'un acteur réalise deux actions à la suite sans que l'autre ne fasse aucune action. Les utilisateurs réels peuvent également enchaîner rapidement des actions s'ils le souhaitent. La non symétrie entre les différents acteurs permet ainsi, au bout d'un certain temps, d'arriver à une convergence c'est à dire à la disparition du conflit (lorsqu'un acteur aura réussi à utiliser l'outil).

Pour éviter d'avoir trop de conflits sur les outils, on peut donner un poids important au critère de disponibilité des outils. On peut également s'arranger pour ne pas avoir de conflit possible sur les outils en mettant à disposition suffisamment d'outils pour tout le monde.

5.6 Extensions

Le mécanisme de sélection d'action que nous proposons peut être étendu pour prendre en compte plus d'informations. Plusieurs extensions sont ainsi envisageables. On peut par exemple citer l'intégration de planificateurs de tâches plus complexes ou d'un planificateur de mouvement. Comme nous l'avons vu dans la section 4.5 nous proposons actuellement deux types de préconditions : les préconditions sur l'état d'une ressource et les préconditions sur l'état d'une relation. Pour le moment, seules les préconditions sur l'état d'une ressource font l'objet d'un raisonnement (mécanisme de gestion de ressources décrit dans la section 3.1.1.2), ce raisonnement permet de fournir des informations qui vont être prises en compte par le module de répartition. Il est envisageable de mettre en place d'autres mécanismes de raisonnement sur les actions (qui exploiteraient par exemple les préconditions sur l'état d'une relation) et de prendre en compte leur résultat comme critère pour la répartition (exemple : nombre d'actions intermédiaires requises, difficulté de la solution, etc). Nous pouvons donc greffer des nouveaux planificateurs de tâches, stocker leur résultat qui sera ensuite utilisé pour la partie réalisation effective de l'action (voir section 5.1.3) et prendre en compte la solution trouvée dans le module de répartition (en précisant une mesure de la complexité de la solution trouvée). Dans un premier temps on peut envisager d'intégrer un mécanisme capable de raisonner sur l'état des relations. Le problème qui se pose ici est de savoir de quelles actions il pourrait disposer : on ne veut pas lui donner toutes les actions possibles car sinon le scénario n'est plus respecté, il faudrait donc définir un sous ensemble d'actions exploitables par un tel mécanisme de raisonnement.

C'est également de cette manière que l'on pourrait intégrer un planificateur de mouvement pour gérer les déplacements de l'humanoïde, voire l'accessibilité des objets. Le planificateur pourrait alors nous donner des informations sur la distance du chemin à parcourir, sa simplicité (beaucoup d'obstacles potentiellement mobiles à éviter ou non), etc. Le problème ici est que le chemin dépend également des déplacements des autres acteurs et un chemin calculé à un instant peut devenir non valide assez rapidement. C'est pourquoi il vaudrait mieux pouvoir évaluer la difficulté potentiel du chemin (y a t il des acteurs qui seraient susceptibles de gêner) et une longueur de chemin approximative (ou par exemple la longueur minimale sans prendre en compte les obstacles mobiles) plutôt que des données précises valables seulement pour un instant t (comme la longueur exacte du chemin à parcourir).

Une autre extension envisageable serait de rajouter des passes d'optimisation au module de répartition une fois qu'une première répartition possible a été trouvée. Le but serait par exemple d'optimiser cette répartition pour faire en sorte que chaque acteur ait une action à réaliser et de revoir le classement en conséquence.

5.7 Analogie avec les théories sur le travail en équipe

Les théories de travail en équipe que nous avons évoquées dans notre état de l'art à la section 1.3.2.1 proposent que chaque agent gère parallèlement son propre état mental et l'état mental de l'équipe. L'état mental de l'équipe inclut les buts pour l'équipe et les intentions d'actions de chacun des membres. Dans un contexte de formation à la procédure une analogie

peut être réalisée entre les objectifs de l'équipe et le scénario ; ainsi l'état mental actuel de l'équipe est symbolisé par l'état actuel du scénario. Notre scénario se substitue ainsi à cette gestion locale. Les intentions de collaborer par exemple sont directement exprimées dans le scénario afin de pouvoir être comprises et effectuées à la fois par des humains virtuels et des utilisateurs réels. Cao [Cao05] nous indique que des études psychologiques sur le travail en équipe ont d'ailleurs montré que les équipiers ont des modèles mentaux partagés et qu'ils ont des attentes les uns envers les autres. Ces attentes sont de notre point de vue décrites dans le scénario et traduites par notre module de répartition qui modélise la répartition optimale pour l'équipe en sélectionnant les membres les plus aptes pour chacune des action à réaliser.

La principale différence entre nos modèles et ces théories réside dans le fait que nous avons décidé de centraliser la connaissance sur l'objectif de l'équipe (la procédure) et sur les attentes de chacun. Ce choix est justifié par plusieurs raisons. Tout d'abord nous sommes dans un contexte de formation à la procédure et il nous faut donc un système central qui contrôle le bon déroulement du scénario. Dès lors il serait redondant de dupliquer cette analyse au niveau de chaque acteur. Dans notre modèle le scénario est commun à tous les agents et ils ne disposent pas chacun d'une copie de ce scénario. Le moteur de scénario tient à jour l'état d'avancement du scénario pour tous les acteurs mais aussi pour le système (pour vérifier que la procédure se déroule correctement) et pour le formateur ce qui permet d'éviter toute erreur. En effet il est utile de déporter cette connaissance dans les agents dès lors qu'elle peut être partielle ou qu'elle peut inclure des erreurs (notion d'interprétation), il n'est pas utile de dupliquer la même connaissance et les mêmes traitements.

Dans SECUREVI [Que02] le modèle suppose que tous les agents ont une parfaite connaissance de la procédure, partagent la même volonté de la réaliser connaissent les rôles de chacun et sont omniscients en terme de réalisation des actions de chacun (ils savent tous ce que tous les autres ont fait). Dès lors, si toutes ces informations sont partagées par tous, pourquoi vouloir les dupliquer et dupliquer leur traitement au sein de chaque agent ? Il serait possible pour nous aussi, dans une logique d'agents autonomes, de déporter la connaissance sur la procédure ainsi que sur son évolution au sein de chaque acteur. Mais cela aurait pour conséquence des calculs redondants, des échanges de messages multiples pour signaler la réalisation des actions de chacun. Il nous est donc paru plus optimal de factoriser ce traitement.

De plus, nous avons deux types d'acteurs qui participent à ce travail d'équipe : des utilisateurs réels et des humains virtuels, il a donc fallu trouver un modèle de représentation commun et compréhensible par l'un comme par l'autre des tâches d'équipe à effectuer. C'est pour cela que des intentions de collaborer sont représentées dans le scénario et non pas directement échangées entre les acteurs.

Lorsque les théories de travail en équipe sont utilisées pour modéliser un travail impliquant à la fois des agents virtuels et des utilisateurs humains, certains systèmes [LRS99] tentent de proposer une reconnaissance de plan afin de deviner les intentions des utilisateurs sans avoir à leur demander explicitement leurs buts. La reconnaissance de plan a pour objectif d'inférer les intentions à partir des actions. Dans notre module de répartition des actions nous essayons également de proposer une reconnaissance de plan, en nous servant du scénario. Les deux critères de continuité (niveau scénario et niveau action) ont en effet pour but de détecter respectivement la branche du scénario choisie et l'action du scénario visée. Cette détection est intéressante lorsqu'il s'agit de deviner les intentions d'utilisateurs réels puisque, pour des humains virtuels,

ces informations sont en réalité déjà connues. Nous supposons donc qu'un humain réel aura tendance à poursuivre une branche de scénario qu'il a entamée et que s'il prend un outil c'est dans le but de s'en servir. En donnant ensuite un poids important à ces critères nous incitons les humains virtuels à être cohérents dans leurs choix d'action ; de plus si un utilisateur réel demande de l'aide nous allons lui indiquer l'action la plus naturelle à viser étant donné ses actions précédentes.

Troisième partie

Application

Chapitre 6

Mise en œuvre et résultats

Dans les chapitres précédents nous avons présenté des modèles permettant de gérer à la fois la collaboration et les humains virtuels. L'intégration de ces modèles dans GVT a été accompagnée par des modifications au niveau de l'interface et des métaphores d'interaction notamment. Dans la section 6.1 nous détaillons donc ces modifications pour passer de la version 1 de GVT, permettant une formation individuelle à la version 2 permettant une formation à des procédures collaboratives où utilisateurs réels et humains virtuels collaborent. Puis nous présenterons dans la section 6.2 une application qui consiste à apprendre le montage à plusieurs d'un meuble de cuisine.

6.1 GVT

Dans le chapitre 2 nous avons présenté le contexte du projet GVT, ainsi que les modèles fondamentaux sur lesquels l'application repose. Rappelons que GVT est une plateforme permettant de créer des environnements virtuels de formation à des procédures industrielles, de type procédures de maintenance. GVT repose sur la plateforme de réalité virtuelle OpenMASK [LCAA08] développée au sein de l'équipe projet Bunraku[1] de l'IRISA. La dernière version de GVT, GVT 2.0 intègre à la fois la dernière version d'OpenMASK (OpenMASK4) et les modèles que nous avons présentés dans ce manuscrit. En effet tous les modèles présentés dans les chapitres précédents ont été intégrés à GVT au sein d'un prototype. Ce prototype permet à plusieurs utilisateurs et à des humains virtuels de se partager une scène. Le modèle d'acteur, décrit dans le chapitre 3, permet donc à la fois de représenter des utilisateurs réels et des humains virtuels. Il est possible de remplacer dynamiquement un humain virtuel par un utilisateur réel (et inversement), à n'importe quel moment de la session de formation. Un acteur est décomposé en ressources (actuellement deux mains), doté de capacités, se voit attribuer un rôle et définir un profil collaboratif. Le modèle STORM permet maintenant de gérer trois types d'entités : des objets, des relations et des représentations d'acteur. Notons toutefois que les relations et les représentations d'acteur sont également des objets STORM (une relation peut donc gérer des objets, d'autres relations et également des représentations d'acteur). Dans le cadre

[1] http ://www.irisa.fr/bunraku

des propositions contenues dans ce manuscrit, le langage de scénario a été étendu : il est maintenant possible de décrire, dans le scénario, les associations des acteurs aux actions grâce aux rôles ou aux fils d'activité, de préciser des pré et post conditions sur l'état des mains ainsi que des préconditions sur l'état d'une relation STORM, d'utiliser des types à la place des instances pour spécifier un objet. Le scénario devient collaboratif grâce à l'introduction de différents moyens pour représenter la collaboration entre les acteurs, de la coordination jusqu'aux actions collaboratives. Le moteur de scénario est ensuite capable d'interpréter et de traiter toutes ces informations présentées dans le chapitre 4. Le mécanisme de sélection d'action décrit dans le chapitre 5 a lui aussi été intégré. Un module de répartition des actions vient se greffer entre le moteur de scénario, le moteur d'interactions et les différents acteurs. Un module de prise de décision est ajouté à chaque acteur qui va leur permettre de sélectionner une action à réaliser ; ce module est complété par un module de réalisation de l'action qui permet de transformer, si besoin est, l'action visée en une action implicite, requise au préalable et directement réalisable.

L'intégration de ces nouveaux modèles a été accompagnée par des changements au niveau des métaphores pour représenter à la fois l'état des acteurs et leur activité (section 6.1.1) et également par l'ajout de nouvelles relations (section 6.1.2).

6.1.1 Représentation de l'activité

Bien que le cœur de nos contributions concerne des modèles pour représenter notamment l'activité d'un acteur, il fallait également rendre visible cette activité afin de la rendre compréhensible par les autres. Pour cela nous avons dû créer de nouvelles métaphores. Nous proposons donc des métaphores simples à mettre en œuvre mais qui ont l'avantage d'être à la fois porteuses de sens et génériques.

L'introduction de la dimension collaborative dans GVT a nécessité de revoir les différentes métaphores utilisées pour représenter l'état de l'utilisateur et son activité. En effet, dans la version 1 de GVT, l'apprenant était représenté par une main. Lorsque l'apprenant prenait un objet une animation déplaçait la main jusqu'à l'objet, puis l'objet revenait au premier plan et remplaçait la main. Lors d'actions plus complexes, un gif animé était parfois utilisé pour représenter l'action en cours de réalisation (par exemple une vis qui tourne pour une action dévisser). La figure 6.1 illustre une séquence d'actions pour positionner un anneau de levage dans GVT1. Au départ c'est la main qui est active, l'utilisateur sélectionne alors l'anneau sur le bureau et sélectionne l'icône associée à l'action prendre. L'anneau de levage vient alors remplacer la main en tant qu'objet actif. L'utilisateur sélectionne alors l'emplacement où placer l'anneau de levage. La main redevient alors l'objet actif et l'utilisateur s'en sert pour visser l'anneau. L'action de visser est traduite par un gif animé.

Dans un environnement collaboratif ces métaphores ne conviennent pas, en particulier l'animation pour prendre et poser un objet. En effet un utilisateur verrait des objets traverser la scène en volant sans comprendre ce qui se passe. De plus il faut représenter l'utilisateur par un avatar plutôt que par une simple main. Dès lors il serait possible d'envisager une animation pour la prise et la pose d'objet mais une telle animation nécessite de résoudre certains problèmes comme par exemple la manière de prendre l'objet de manière réaliste, la technique pour s'approcher de l'objet à attraper, etc. Des travaux de recherche portent spécifiquement sur ces problématiques (par exemple les travaux de Badawi [BD04]) mais il ne nous parait pas fon-

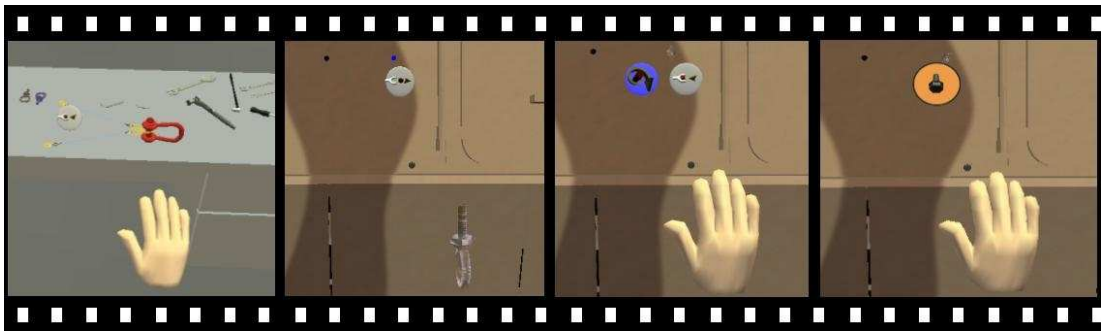


FIG. 6.1 – Séquence d'actions dans GVT1 pour prendre un anneau de levage, le poser puis le visser

damental de les intégrer ici. De plus nous cherchons des métaphores facilement généralisables, or si on représente de manière réaliste les gestes de prise et de pose, qu'en est-il pour les autres gestes comme le geste de vissage/dévissage par exemple ? Nous avons donc préféré privilégier la généricité au réalisme, tout en conservant l'aspect expressif de ces métaphores. Dans la section 6.1.1.1 nous allons donc décrire comment nous représentons un acteur et son état, dans la section 6.1.1.2 nous détaillerons les métaphores que nous avons choisies pour représenter l'activité des acteurs et dans la section 6.1.1.3 nous expliquerons les choix que nous avons fait en matière de déplacement des acteurs.

6.1.1.1 Avatar et visualisation de l'état d'une ressource

Pour représenter un acteur dans l'environnement virtuel nous avons choisi de remplacer la main par un avatar de forme humanoïde (voir figure 6.3). Cet avatar permet alors de savoir si un acteur est présent dans la scène, de l'identifier (grâce aux infobulles), de le localiser et de voir son état. Les avatars utilisés dans le projet GVT ont été réalisés avec le modeleur de personnages humanoïdes Quidam[2].

Le représentation de l'état des mains de l'acteur se fait grâce à des icônes. Pour représenter l'état des mains, il faut pouvoir représenter les trois état différents pour la main. Lorsque la main est *libre* l'icône affiche cette main (la main droite sur l'icône de droite et la gauche à gauche). Lorsque la main est dans l'état *objet attrapé* alors l'objet apparaît dans l'icône, sur fond gris. Cet objet est donc actif et l'acteur peut interagir avec. Lorsque la main est *occupée* l'objet qui l'occupe apparaît dans l'icône mais avec un fond orange pour symboliser l'impossibilité d'interagir. La figure 6.2 illustre la représentation des différents états des mains avec de gauche à droite, une main gauche libre, une main droite libre, une main qui a attrapé un clou et une main occupée par un clou.

Il faut ensuite distinguer le point de vue de l'utilisateur du point de vue des autres acteurs (voir figure 6.3). L'utilisateur voit au premier plan les icônes qui représentent l'état de ses mains, ce qui lui permet de facilement visualiser les objets qu'il possède. Les autres acteurs

[2]http ://www.n-sided.com/

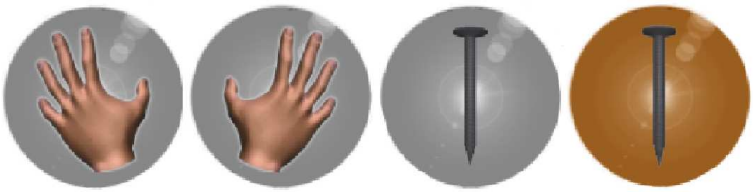


FIG. 6.2 – Les différents états pour les mains

voient ces icônes placées au dessus de la tête de l'avatar de l'acteur.

6.1.1.2 Métaphores d'interaction

Nous venons de décrire la manière de représenter visuellement un acteur et son état, il nous faut maintenant représenter son activité de manière compréhensible à la fois pour l'utilisateur qui fait l'action mais également par les autres utilisateurs qui observeraient cette action.

Nous avons choisi de conserver le menu avec les icônes pour représenter les actions possibles avec un objet et les gifs animés pour représenter l'action en cours. Lorsqu'un utilisateur réalise une action, le gif animé correspondant apparaît en premier plan afin de lui donner un retour immédiat sur l'action qu'il a réalisée. Ce gif animé apparaît également au dessus de l'avatar de l'acteur ayant réalisé l'action, entre les deux icônes représentant l'état de ses mains, afin que les autres utilisateurs comprennent à la fois quel acteur interagit et quelle est l'action réalisée. Sur la figure 6.3, l'utilisateur au premier plan effectue une action visser. En haut à gauche de l'image, on peut voir le point de vue d'un humain virtuel qui lui fait face, il voit alors le gif animé correspondant à l'action visser apparaître au dessus de la tête de l'avatar de l'utilisateur.

Lorsqu'un acteur interagit avec un objet, l'objet change de couleur pour montrer qu'une interaction est en cours. Si l'action a pour conséquence de changer l'état d'une main, l'icône correspondant à cette main s'agrandit, est remplacée par l'icône correspondant au nouvel état, puis reprend sa taille normale. La figure 6.4 illustre ce changement de l'état des mains de l'utilisateur suite à l'action *prendre le tournevis*. Ainsi l'objet source de l'interaction ainsi que la conséquence de l'interaction sur l'état des mains sont bien visibles. Si l'action est une prise d'objet, l'objet disparaît alors de la scène et n'est plus représenté que dans les icônes de son possesseur. Les actions qui ne modifient pas l'état d'une ressource sont plus difficiles à visualiser par un utilisateur extérieur, c'est pourquoi il est utile d'associer à ces actions un gif animé.

Ces métaphores, à la fois pour représenter l'état d'un acteur mais également l'action en cours sont très génériques puisqu'elles s'adaptent à tous les types de relation. Il suffit de fournir une icône associée à un objet et une animation associée à une relation et le reste est automatique.

Pour mieux visualiser l'action d'un coéquipier il est possible d'avoir un viewport qui nous donne la vue d'un partenaire, ou une autre vue de la scène si on le désire (voir figure 6.3).

Les différentes métaphores que nous avons proposées pour représenter l'activité d'un acteur

FIG. 6.3 – Deux acteurs se font face et agissent dans la scène

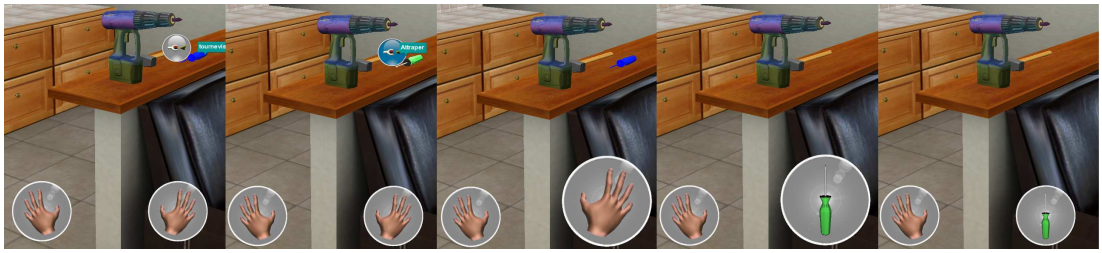


FIG. 6.4 – Animation suite à l'action prendre un tournevis

nous ont servi à réaliser un prototype. Une fois le prototype finalisé il sera possible de réaliser une étude ergonomique afin de valider ces choix ou de les remettre en cause.

6.1.1.3 Animation et déplacements

Pour gérer l'animation des avatars et planifier leurs déplacements nous avons simplement intégré des travaux existants.

L'animation des avatars dans GVT est gérée par la bibliothèque d'animation MKM[3] (Manageable Kinematic Motion) [MKMA04]. MKM est un moteur d'animation temps-réel d'humanoïdes synthétiques, basé sur la cinématique, qui synchronise, mélange et adapte des mouvements automatiquement.

Dans l'équipe Bunraku de l'IRISA des travaux existent permettant de gérer la navigation d'un personnage virtuel dans un environnement 3D complexe (exemple [Par07]). De tels travaux sont en cours d'évaluation en vue d'une intégration dans GVT.

En attendant la fin de cette intégration la navigation peut se faire de deux manières, de manière libre ou par points de vue. Avec la manière libre il est possible de parcourir librement l'environnement, sans se soucier des problèmes de collision notamment. La navigation par points de vue était déjà adoptée dans GVT 1. Elle permet de guider l'apprenant en lui offrant un choix limité de points de vue intéressants pour interagir et évite ainsi qu'il passe trop de temps à se positionner dans la scène. Un utilisateur dispose d'une vue à la première personne et son avatar va suivre le déplacement de la caméra.

6.1.2 Nouvelles relations

Nous avons profité des nouvelles possibilités offertes par nos modèles pour ajouter des nouveaux types de relation et offrir ainsi une plus grande variété d'interactions. Voici des exemples de relations que nous avons ajoutées.

Des relations qui rendent la main occupée Nous avons proposé de gérer un nouvel état pour la main qui est l'état *occupée* qui indique que la main ne peut plus interagir, ni directement ni par l'intermédiaire de l'objet qui l'occupe. Un type particulier de relation, le type *relationOccupante* est utilisé pour faire transiter la main vers l'état occupée et l'en faire sortir. Une main est donc considérée comme occupée lorsqu'elle est impliquée dans une relation de type *relationOccupante*. Il est alors possible de savoir l'objet qui occupe la main. Nous avons modélisé par exemple une relation qui permet à l'acteur de tenir un objet (un clou dans le but de le clouer, une poignée pour ne pas qu'elle tombe lorsqu'elle sera détachée de son support).

Des relations qui utilisent les deux mains de manière différentes L'acteur possède maintenant deux mains avec lesquelles il peut interagir, nous avons donc créé des relations permettant d'utiliser les deux mains à la fois, de manière différente. C'est le cas par exemple pour la relation clouer. Cette relation gère également une relation de type *relationOccupante* qui permet à l'acteur de tenir le clou. Une fois le clou tenu il est alors possible, si l'acteur a un marteau en

[3] http ://www.irisa.fr/bunraku/MKM/

main, de réaliser l'action clouer. Rien n'empêche au niveau de la relation d'avoir deux acteurs différents pour clouer : l'un qui tient le clou l'autre qui tape avec le marteau. En revanche si l'on veut que ce soit le même acteur qui tienne le clou et qui se serve du marteau, il est possible de le contraindre au niveau du scénario en utilisant par exemple un fil d'activité initialisé sur l'action tenir le clou puis utilisé en mode imposé pour l'action clouer.

Des relations qui utilisent les deux mains de manière identique L'acteur peut également se servir de ses deux mains en même temps et de manière symétrique, par exemple pour attraper un objet à deux mains. Nous avons créé une relation *attraper à deux mains* qui s'utilise sur des objets qui ont la capacité *saisissable2Main* au lieu de *saisissable*. Chaque acteur possède sa propre instance de cette relation. Cette relation surveille les mains de l'acteur et lorsque l'une d'elle n'est plus libre elle se désactive. Le principe d'utilisation de la relation attraper à deux mains est le même que pour la relation attraper simple, mis à part que les deux mains sont mises à contribution. Une fois l'objet attrapé, chaque main est considérée comme tenant le même objet. La pose se fait exactement de la même manière que pour reposer un objet attrapé avec une seule main.

Des relations qui impliquent plusieurs personnes Nous avons également créé une relation qui implique deux acteurs. Cette relation consiste à porter un objet lourd à deux. Cette action collaborative nécessite une phase préparatoire durant laquelle chacun des participants se déclare prêt à collaborer sur l'objet en réalisant l'action soulever. Une fois que chaque acteur a réalisé l'action soulever alors l'objet est considéré comme porté par les deux acteurs. Il suffit alors que l'un des deux décide de le reposer pour que l'objet soit reposé.

6.2 Scénario de montage de meuble

Afin de valider nos modèles nous avons choisi de sortir du domaine applicatif de la maintenance pour nous intéresser au montage collaboratif d'un meuble.

6.2.1 Description du scénario

Le scénario consiste à assembler un meuble de cuisine livré en kit puis à le fixer dans la cuisine. Le meuble à monter est un élément haut. La procédure écrite en LORA est réalisée à partir de la notice de montage du meuble (la notice est donnée en annexe B). La procédure de montage est intéressante car elle contient des actions collaboratives de plusieurs types. Il faut par exemple se mettre à deux pour porter la planche comme le préconise la notice (cf figure 6.5). Durant la phase de fixation au mur, là aussi il y a des actions collaboratives : une personne doit plaquer le meuble contre le mur avec ses deux mains pendant que l'autre, situé à l'autre bout du meuble, maintient le meuble avec une main et fixe le meuble grâce à une vis avec son autre main.

Nous rajoutons un passage qui n'est pas décrit dans la notice de montage et qui concerne la mise en place de la poignée. Pour cela deux mains sont nécessaires : une qui tient la poignée

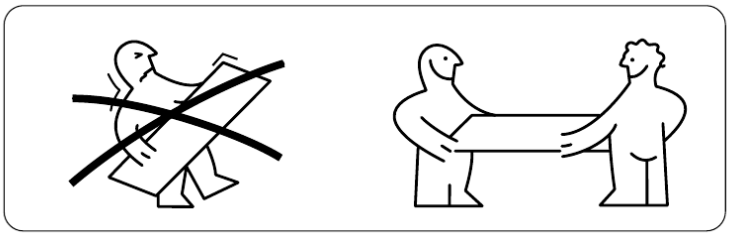


FIG. 6.5 – Extrait de la notice de montage : il faut se mettre à deux pour porter les planches

et l'autre qui visse. Ces deux actions peuvent être réalisées par la même personne ou par deux personnes différentes.

La procédure contient plusieurs séquences qui peuvent être réparties de différentes manières entre les personnes présentes. Le scénario nécessite au moins deux personnes mais il peut y avoir plus de participants. Nous définissons donc deux rôles : le monteur et l'assistant. Le monteur est celui qui utilise prioritairement les outils comme par exemple la perceuse, l'assistant doit aider le monteur lors des tâches collaboratives. Mais la répartition des actions entre eux n'est pas figée. Si plus de deux personnes participent au montage du meuble alors on peut attribuer le rôle d'assistant à deux personnes par exemple.

La notice décrit la procédure de montage d'une manière très linéaire, nous avons choisi de la représenter de manière moins stricte en permettant la parallélisation de certaines opérations. Par exemple, les opérations à réaliser sur le meuble en lui-même et celles à réaliser sur la porte du meuble peuvent être réalisée en parallèle, nous avons pour cela créé deux branches.

La figure 6.6 représente graphiquement la procédure complète écrite en LORA ainsi qu'un extrait où trois branches sont possibles : placer les fixations pour le compas sur le meuble, placer les fixations pour le compas sur la porte ou fixer la poignée sur la porte du meuble.

6.2.2 Résultats

Ce scénario de montage de meuble nous a permis d'illustrer certaines bonnes propriétés de nos modèles, nous allons donc détailler ces propriétés et les extraits du scénario correspondants.

6.2.2.1 Utilisation des fils d'activité

Comme nous l'avons vu la relation clouer nécessite qu'un acteur tienne le clou avant que l'action clouer soit possible. Or rien n'impose au niveau de la relation clouer que l'acteur qui tienne le clou et l'acteur qui plante le clou soit le même. Dans le cadre de ce scénario, nous considérons qu'il serait préférable que ça soit la même personne qui tienne le clou et tape dessus avec le marteau. Nous aimerions également que ce soit le même acteur qui plante tous les clous. Pour cela on va utiliser un fil d'activité. Ce fil d'activité sera initialisé lors de la première action *tenir un clou*. Ensuite ce fil d'activité sera utilisé en mode préférentiel pour les action *tenir* et *clouer* suivantes.

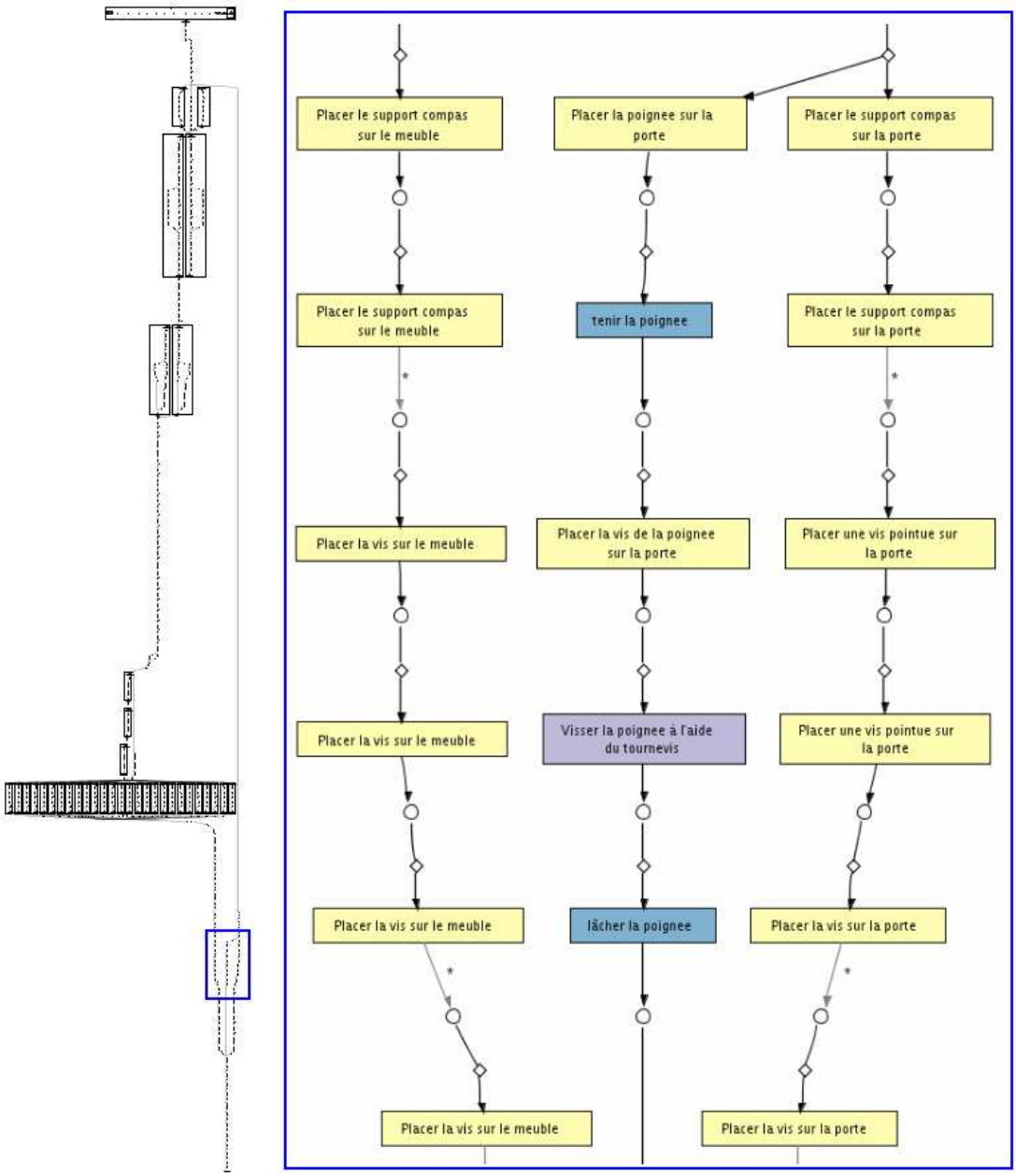


FIG. 6.6 – A gauche la procédure complète de montage du meuble, sur la droite un extrait

Un apprenant et un humain virtuel avec un profil l'incitant à suivre la suggestion du module de répartition réalisent cette portion de scénario. Si l'apprenant réalise le premier l'action *tenir un clou* alors l'humain virtuel laissera l'apprenant réaliser les actions *clouer* et *tenir* suivantes. En revanche si l'humain virtuel est le premier à réaliser l'action *tenir un clou* il va également réaliser les actions suivantes. Cependant, si l'apprenant décide de planter un clou tenu par l'humain virtuel, rien ne l'en empêche.

6.2.2.2 Intérêt des branches dans le scénario

La branche entamée par un acteur est également un critère pour la répartition. A certains endroits du scénario, plusieurs branches sont possibles en parallèles et les humains virtuels vont ainsi avoir tendance à réaliser chacun une branche différente. Par exemple dans ce scénario certaines actions sont à réaliser sur le meuble, et d'autres sur la porte, ce qui se traduit par deux branches d'actions distinctes dans le scénario. Un humain virtuel qui s'engage sur l'une de ces branches va ainsi avoir tendance à l'effectuer jusqu'au bout.

6.2.2.3 Passages collaboratifs

Dans le scénario de montage, les différentes planches doivent être portées à deux. Chaque acteur qui souhaite porter la planche va alors réaliser l'action *soulever* afin de signifier son intention de collaborer. Cette action place chacune des deux mains de l'acteur dans l'état *occupée*, en effet il ne peut plus interagir tant qu'une autre personne ne vient pas l'aider. L'acteur est ainsi bloqué et s'il a un coéquipier dont le profil collaboratif contient la règle aide, ce dernier va venir l'aider aussitôt. Ensuite la planche est considérée comme portée, et n'importe lequel des acteurs peut décider de la poser.

Un autre passage collaboratif plus long consiste à fixer le meuble au mur. Pour cela deux acteurs doivent soulever le meuble, ensuite ils doivent l'appuyer contre le mur. Une fois cette dernière action réalisée, un acteur peut alors lâcher une main (voir figure 6.7). Avec sa main libre il va pouvoir attraper une vis, la placer à l'intérieur de l'équerre de fixation du meuble, placer également la plaque de fixation, puis prendre un tournevis et visser cette vis. Ces mêmes opérations sont réalisés en parallèle et de manière automatique par le moteur de scénario pour l'autre côté du meuble. Ensuite les deux acteurs pourront lâcher le meuble.

6.2.2.4 Adaptation au contexte

Nous avons choisi de nous intéresser à l'étape 1 de la procédure afin d'illustrer la propriété d'adaptabilité au contexte de nos modèles. Au cours de cette étape, après avoir positionné des équerres de fixation sur chacune des faces du meuble, il est nécessaire de positionner deux vis sur chaque équerre puis de les visser. Le scénario est alors composé de deux branches : l'une pour positionner les vis et l'autre pour les visser. Cette portion de scénario est réalisée par deux acteurs : l'acteur 1 est joué par un utilisateur tandis que l'acteur 2 est un humain virtuel avec un profil collaboratif lui indiquant de suivre la suggestion du module de répartition. Le module de répartition va se fonder ici essentiellement sur le critère de facilité, c'est à dire le nombre d'outils requis par l'action à prendre par l'acteur, ainsi que le nombre d'objets que l'acteur va devoir reposer. La valeur de ce critère pour un acteur donné va ainsi varier entre 4 si aucune

FIG. 6.7 – Les deux acteurs soutiennent le meuble contre le mur et l'un deux va lâcher une main

action de prise et de pose n'est requise (l'acteur possède déjà les bons outils) et 0 s'il doit poser les deux objets qu'il a en main et en prendre deux autres. Une passe d'optimisation est effectuée à la fin de la répartition. En effet, une révision de la répartition est faite dans le cas où un acteur A est l'unique premier pour une action 1 et est premier ex-aequo avec un acteur B pour une action 2. Si l'acteur B n'est classé unique premier sur aucune action, on classe l'acteur A deuxième pour l'action 2 afin que l'acteur B devienne l'unique premier pour cette action.

L'acteur 1 commence par prendre le tournevis.

Le tableau suivant illustre alors les scores et les classements des deux acteurs à l'issue de la répartition. L'état courant de chaque acteur figure dans le tableau par un couple (état de la main droite, état de la main gauche). Les scores figurant dans le tableau tiennent compte uniquement du critère facilité avec un poids de 1. Les modifications de classement dues à la passe optimisation figurent en rouge dans le tableau.

	Visser la vis 1		Placer la vis 2	
	score	classement	score	classement
Acteur 1 (tournevis,libre)	4	1	3	2
Acteur 2 (libre,libre)	3	2	3	1

L'acteur 2 va ainsi placer les vis et laisser l'acteur 1 les visser. Admettons que l'utilisateur (acteur 1) décide alors de reposer le tournevis après avoir visser la première vis et décide de placer les vis suivante. Il prend alors une vis. Les nouveaux scores sont les suivants :

	Visser la vis 2		Placer la vis 3	
	score	classement	score	classement
Acteur 1 (vis,libre)	3	2	4	1
Acteur 2 (libre,libre)	3	1	3	2

La situation est alors inversée, l'acteur 1 a déjà une vis en main ce qui le favorise pour l'action placer la vis ; l'acteur 2 se retrouve alors classé premier pour l'action visser, il va donc prendre le tournevis et laisser l'acteur 1 placer les vis.

Cet exemple montre les capacités d'adaptation d'un humain virtuel aux actions d'un apprenant. On voit ainsi que le module de répartition aide les différents acteurs à se positionner les uns par rapport aux autres pour chacune des actions du scénario, et propose une répartition optimale. Ici, nous n'avons considéré que le critère facilité qui permet d'avoir une vision à un instant t sur la répartition optimale. Mais le module de répartition tente également de deviner les intentions des différents acteurs, grâce aux actions passées de chaque acteur prises en compte par les critères de continuité, afin d'effectuer la meilleure répartition possible en respectant les intentions de chacun.

6.2.2.5 Collaboration implicite

Une branche du scénario consiste à fixer la poignée de porte du meuble. Cette branche peut être réalisée en parallèle avec des opérations de montage de la partie principale du meuble. Les actions à réaliser sont donc les suivantes (voir figure 6.6) :

– placer la poignée
– tenir la poignée
– placer la vis
– visser la vis
– lâcher la poignée

Pour cette portion de scénario nous aurons également deux acteurs : un apprenant et un humain virtuel avec un profil collaboratif le tendant à aider l'apprenant.

Au cours du scénario, l'apprenant fait une mauvaise manipulation et se blesse. Il ne peut donc plus se servir de sa main droite. L'apprenant décide de réaliser la branche de fixation de la poignée. Il va donc prendre la poignée, la positionner et la tenir, pendant que son coéquipier virtuel réalise une autre branche du scénario (par exemple fixer les équerres de fixation du meuble). Une fois qu'il a réalisé l'action *tenir la poignée* l'apprenant se retrouve alors avec ses deux mains occupées donc inutilisables (l'une est blessée et l'autre tient la poignée), il ne peut alors plus avancer dans le scénario et est donc bloqué (voir figure 6.8). Le module de répartition détecte que l'apprenant ne peux plus réaliser aucune action et indique à l'humain virtuel l'action suivante sur la branche entamée par l'apprenant : en effet cette action pourrait permettre de débloquer l'apprenant. L'humain virtuel qui a dans son profil collaboratif la règle *aide* activée, va alors aider l'apprenant et tenter de le débloquer en interrompant sa branche pour réaliser l'action suivante sur la branche de l'apprenant, à savoir *placer la vis*. L'apprenant est toujours bloqué alors l'humain virtuel va réaliser également l'action *visser la vis*. L'apprenant est alors débloqué puisque le scénario l'autorise à réaliser l'action *lâcher la vis*. L'apprenant va donc libérer sa main qui tenait la poignée ce qui va clore cette branche pendant que l'humain virtuel va pouvoir reprendre la branche qu'il avait laissée en suspend.

6.2.3 Conclusion

Ce scénario collaboratif réel nous a permis de valider nos modèles et leur intégration au sein du prototype GVT 2. Il nous a permis d'illustrer les propriétés de nos modèles dans un cas concret d'utilisation. Le scénario s'est également révélé plus facile à écrire grâce aux actions implicites et à l'utilisation des types au lieu des instances d'objets dans le scénario. Ce scénario dans lequel la répartition des actions entre les participants n'est pas figée à priori tire le meilleur profit du module de répartition. En effet ce dernier permet de proposer à chaque instant une répartition optimale des actions en fonction du contexte de réalisation. Les humains virtuels se révèlent ainsi capables de s'adapter aux actions de l'apprenant, en réalisant des actions complémentaires ou en leur venant même en aide si besoin est. Le scénario a également démontré l'utilité des branches et des fils d'activité.

FIG. 6.8 – L'apprenant a une main blessée et tient la poignée : il se retrouve bloqué

Quatrième partie

Conclusion et perspectives

Conclusion et perspectives

Dans cette dernière partie nous allons dresser un bilan des travaux réalisés pendant cette thèse. Nous donnerons ensuite quelques pistes et perspectives pour le futur.

Bilan

L'objectif de cette thèse était de faire évoluer GVT pour proposer une formation à des procédures collaboratives où des utilisateurs réels collaborent entre eux et avec des humains virtuels. Nos contributions portent ainsi sur trois éléments : la modélisation de l'activité d'un acteur, la modélisation du scénario collaboratif ainsi que la mise en place d'un mécanisme de sélection d'action.

L'activité d'un acteur dans GVT consiste à réaliser des actions dans l'environnement en tenant compte de ses possibilités (son état, ses capacités), du rôle qu'il joue, de l'état d'avancement du scénario et de l'activité de ses différents partenaires. Le modèle d'acteur présenté dans cette thèse permet de représenter de manière identique l'activité d'un utilisateur réel et celle d'un humain virtuel ; il est ainsi possible de remplacer dynamiquement un apprenant par un humain virtuel et inversement. Les interactions offertes à l'acteur sont facilement paramétrables, il suffit de définir les ressources dont il dispose, d'attribuer des capacités globales communes à tous les acteurs et de définir les capacités associées à un rôle. L'acteur qui va jouer un rôle se verra donc doté des capacités d'interaction associées. L'acteur est également capable d'effectuer des raisonnements, notamment sur ses ressources (il peut calculer une séquence d'actions de base pour amener ses ressources dans un état donné) et sur les actions à réaliser (grâce à son module décisionnel). Le comportement d'un humain virtuel peut être paramétré grâce à son profil collaboratif, en précisant des règles de comportements vis à vis du scénario et de ses coéquipiers. De nouvelles interactions sont désormais accessibles à l'apprenant, comme des actions nécessitant deux mains ou des actions à plusieurs.

Pour modéliser le scénario nous avons proposé une extension du langage graphique de description de scénario LORA afin de lui ajouter une dimension collaborative. Nous y avons introduit la notion de rôle permettant de décrire l'assignation des personnes aux actions. L'association des personnes aux actions peut se faire de deux manières différentes : grâce aux rôles pour des actions ponctuelles ou indépendantes et grâce aux fils d'activité pour des séries d'actions liées. L'assignation peut être multiple afin de pouvoir représenter de manière réaliste certaines procédures collaboratives où la répartition des actions entre les acteurs n'est pas fixée à priori. Le candidat qui réalisera effectivement l'action sera alors choisi dynamiquement.

Nous avons également simplifié la spécification du scénario en rendant implicites des actions basiques comme la prise et la pose d'objets. Le scénario se rapproche ainsi davantage de la description de la procédure provenant des cartes de travail. Ces actions basiques sont remplacées par une description de l'état des mains requis en précondition d'une action. Le scénario gagne ainsi en adaptabilité. Nous avons également introduit dans le scénario de nouvelles actions : les actions collaboratives précédées de déclarations d'intention de la part des collaborateurs potentiels.

Enfin nous présentons dans cette thèse un mécanisme de sélection d'action dont l'objectif est double : permettre à un humain virtuel de sélectionner une action à réaliser et donner des conseils à l'apprenant sur l'action à choisir. Pour cela nous avons factorisé l'analyse du scénario au sein d'un module de répartition des actions chargé de déterminer, pour chacune des actions possibles dans le scénario, le meilleur candidat en fonction de critères paramétrables et évalués dynamiquement. Ce module va ainsi mutualiser l'analyse des actions du scénario et prévoir une répartition idéale des actions entre les différents acteurs. Parmi les critères proposés, certains ont pour objectif d'effectuer de la reconnaissance de plans, c'est à dire de tenter de deviner les intentions des acteurs et notamment celles des utilisateurs. Ce module, central, est complété par des modules décisionnels dont dispose chaque acteur. Un acteur possède, en plus, un profil collaboratif décrivant son comportement par un ensemble de règles. A partir de la suggestion du module de répartition et de son profil collaboratif un acteur va donc pouvoir sélectionner un objectif à attendre. Cet objectif peut être une action directement réalisable ou une action nécessitant des actions implicites préalables. Ce mécanisme apporte des avantages majeurs :

- le scénario est plus flexible et peut s'adapter à des contextes de réalisation différents (nombre de participants variable, états de départ différents, etc) sans avoir à modifier sa description
- il est possible pour un humain virtuel d'anticiper les besoins de ses coéquipiers ce qui peut se traduire par exemple par l'apparition de collaboration implicite
- il est également possible de détourner l'usage de GVT afin de faire de la mise au point incrémentale de procédure. Pour cela on laisse des humains virtuels réaliser la procédure et on affine progressivement le réglage de certains paramètres. Il est aussi possible de tester l'adaptabilité de la procédure à un contexte de réalisation différent et ainsi de déterminer ses limites

Le mécanisme de sélection d'action est paramétrable à plusieurs niveaux : au niveau global en précisant les critères à considérer pour la répartition et le poids à apporter à chacun et au niveau local en ajustant le profil collaboratif des humains virtuels pour leur donner un comportement particulier, à des fins pédagogiques par exemple.

Ces différents modèles ont été intégrés à GVT au sein d'un prototype. Dans cette thèse nous avons présenté un scénario applicatif réalisé dans GVT qui consiste à monter collaborativement un meuble livré en kit. Ce scénario permet à la fois de valider nos modèles mais également de montrer que GVT peut sortir du cadre de la maintenance industrielle et être utilisé pour des applications plus grand public.

Le projet GVT continue et différentes perspectives sont envisageables pour poursuivre les travaux présentés dans ce manuscrit de thèse.

Perspectives

La première perspective concerne la révision de la plateforme de GVT afin de tirer profit des nouveaux modèles proposés. Il serait par exemple intéressant de réaliser un couplage entre le mécanisme de sélection d'action et le moteur pédagogique. En effet il est possible d'utiliser un profil standard pour les utilisateurs réels et ainsi d'obtenir un choix d'action justifiable (pour le moment la pédagogie propose la première action trouvée). Le moteur pédagogique, en accédant au résultat du mécanisme de sélection d'action pourrait donner des conseils pédagogiques plus précis aux apprenants. Il pourrait par exemple informer l'apprenant des actions intermédiaires à effectuer avant de pouvoir réaliser une action donnée du scénario. Il serait également possible de définir des profils collaboratifs à associer automatiquement aux humains virtuels partenaires d'un apprenant, en fonction de son cursus pédagogique. Nous pourrions aussi définir des profils collaboratifs plus variés et notamment définir un profil de perturbateur qui pourrait réaliser des actions spécifiques identifiées comme perturbantes (par exemple mettre en route une sirène), qui pourrait sélectionner des actions de manière contre intuitive (aucune continuité dans ses choix d'action) ou alors des actions pour gêner directement l'apprenant (par exemple prendre les outils dont il a besoin). Il faudrait également faire évoluer les outils auteurs pour prendre en compte l'aspect collaboratif, en permettant, dans un premier temps, de décrire le scénario collaboratif grâce à l'outil auteur nommé "construire en faisant" (décrit dans la thèse de Nicolas Mollet [Mol05]) qui permet de créer dynamiquement un scénario en le réalisant dans l'environnement de GVT. Il faudrait alors préciser les rôles ou fils d'activité associés à chacune des séquences d'actions réalisées dans l'environnement. Dans un second temps il serait même possible d'envisager un outil auteur collaboratif qui permettrait à plusieurs personnes de décrire ensemble le scénario (construire en faisant collaboratif). Chacun réaliserait alors des parties du scénario et le post-traitement consisterait à préciser les points de synchronisation et à affiner l'assignation des personnes aux actions.

La seconde perspective concerne l'intégration de mécanismes de raisonnement plus avancés et leur prise en compte par le module de répartition des actions. Comme nous l'avons déjà évoqué dans le chapitre 5, il serait intéressant d'intégrer un planificateur de mouvements et/ou un planificateur de tâches plus complexes. Le résultat de ces planificateurs pourrait ensuite être intégré comme critère pour la répartition et pris en compte par les acteurs lors de leur choix d'action. Il faudrait alors trouver les critères à considérer pour décrire une solution trouvée (qui reflètent la difficulté de la solution proposée). La répartition serait alors fonction non seulement de critères logiques (l'action est-elle faisable théoriquement) mais également de critères d'accessibilité physique des objets avec lesquels il faut interagir par exemple. Dans sa thèse qui s'inscrit dans le domaine de la robotique, Fabien Gravot [Gra04] propose une méthode permettant de coupler fortement un planificateur de mouvements avec un planificateur de tâches pour la résolution de problèmes par des robots manipulateurs. Il prend également l'exemple de l'assemblage collaboratif d'un meuble IKEA, mais par des robots qui doivent prendre en compte l'environnement pour planifier leurs actions et manipuler les objets. Un autre raisonnement possible concerne l'utilisation des outils. Il serait possible d'intégrer un mécanisme qui tenterait de "deviner" (grâce à des heuristiques) si un outil est susceptible de (re)servir à l'acteur ou non (regarder si l'outil a déjà servi depuis sa prise, si l'action visée par la prise est toujours possible, voire analyser le scénario plusieurs actions en avant). Il serait ainsi possible

de rajouter des règles de sélection permettant de reposer un outil qui serait devenu inutile. Nous avons indiqué dans ce manuscrit que certains critères du module de répartition des actions ont pour objectif de tenter de prédire les actions des apprenants. Pour le moment nous tentons juste de prédire la prochaine action du scénario envisagée grâce aux objets attrapés. Il serait possible d'aller plus loin dans la reconnaissance de plans en se basant par exemple sur l'attention de l'apprenant (les objets sur lesquels il clique par exemple) ou sur ses déplacements.

La troisième perspective pourrait être de séparer le mécanisme de sélection d'action de GVT pour l'intégrer à un autre EVFC avec une architecture différente. Cet EVFC pourrait alors profiter des possibilités offertes par la répartition dynamique des actions entre les acteurs. Le mécanisme de sélection d'action nécessite trois briques pour sa mise en place : des ajouts au langage de scénario, un module de répartition des actions et un module décisionnel pour les acteurs. La brique de spécification peut être ajoutée à n'importe quel langage de scénario, puisqu'il suffit de rajouter des informations sur l'association d'une action avec des rôles ou des acteurs. La deuxième brique sert d'interface entre le scénario et un modèle d'activité de l'acteur, elle prend en entrée des actions et rend en sortie une liste d'actions possibles pour chaque acteur, avec des informations supplémentaires sur chaque action. Il est envisageable d'avoir un modèle de scénario et un modèle d'activité de l'acteur différents. En revanche, l'évaluation de critères se fait par des mécanismes qui dépendent des modèles choisis pour modéliser l'environnement et les interactions possibles, il faudrait donc adapter cette partie évaluation de critères. La dernière brique se greffe sur un modèle de l'activité de l'acteur (peu importe le modèle utilisé) et permet de lui apporter de l'autonomie sur ses choix d'action. Cette brique prend en entrée la liste d'actions provenant de la brique précédente et éventuellement d'autres actions provenant d'autres modules et fournit en sortie une action. Les règles de sélection qui s'appuient sur des critères provenant de la brique de répartition n'ont pas besoin d'être changés. La partie qui transforme cette action sélectionnée en une demande d'action (l'action demandée peut être différente de l'action sélectionnée lorsqu'une action implicite est requise) peut également être conservée. En revanche la demande d'action aura un format qui dépendra de l'EVFC. Le mécanisme de sélection d'action étant essentiellement basé sur le transit d'actions, il pourrait donc être adapté à une autre architecture. Les gains apportés par le mécanisme de sélection dépendrait alors des propriétés de la représentation du scénario utilisée. En effet, dans cette thèse c'est le langage LORA que nous avons adapté et nous avons donc profité de ses caractéristiques qui étaient de pouvoir décrire un scénario strict. Nos modèles apportent ainsi de la flexibilité sur la répartition des actions entre les participants tout en conservant le déroulement précis qui correspond à la procédure de référence. Nos modèles peuvent très bien être adaptés à des langages de scénario permettant de décrire de manière plus souple les scénarios. Les scénarios ainsi obtenus seraient donc très flexibles dans leur déroulement (aussi bien en terme d'agencement des actions que de répartition).

Nous pouvons également donné quelques pistes de modifications de nos modèles.

- Les critères pour la répartition des actions sont évalués dynamiquement mais ils sont définis de manière statique pour toute la durée d'une session de formation. Il serait possible de préciser des critères à privilégier localement pour certaines actions, comme c'est par exemple le cas dans l'architecture CAST où les critères permettant de sélectionner l'agent qui va accomplir une action sont déterminés seulement pour un plan donné.
- Il serait possible dans une logique d'agents autonomes de déporter la connaissance sur la

procédure ainsi que sur son évolution au sein de chaque acteur (notion de modèle mental partagé). Mais cela aurait pour conséquence des calculs redondants et des échanges de messages multiples pour signaler la fin des actions de chacun. Il est plus optimal de centraliser ce traitement d'autant plus qu'il faut bien un module qui ne dépend d'aucun acteur et qui gère l'évolution de la procédure pour vérifier son bon déroulement puisqu'on se situe dans un cadre de formation.

– Pour le moment nous ne gérons qu'une équipe d'acteurs avec une seule procédure à réaliser. Il pourrait être intéressant de considérer plusieurs équipes ayant chacune une procédure à effectuer mais qui doivent se partager l'environnement et éventuellement se synchroniser par moment.

Enfin nous pouvons donner des pistes à explorer pour exploiter nos modèles.

– Il serait possible de développer les analyses hors ligne du scénario afin par exemple de détecter des situations particulières, de vérifier des propriétés intéressantes ou de rajouter automatiquement des branches pour revenir en arrière.
– Nous avons évoqué dans ce manuscrit la possibilité de se servir de nos modèles pour mettre au point des procédures. Cette utilisation de GVT, bien qu'envisagée, n'a pas encore été testée.

La dernière perspective serait de valider nos modèles grâce à une campagne d'évaluation du prototype collaboratif de GVT.

Cinquième partie

Annexes

Annexe A

Extrait d’une carte de travail Nexter

MA1 4259/4

INTERVENTION	DESIGNATION DE L'INTERVENTION	NTI 1 2
55XXXX XXXX	CHARGEMENT DES MODULES REACTIFS	X

DOMAINE D'APPLICABILITE		
DESIGNATION	**IDENTIFICATION**	
APPLIC. SERIE 3	FCONF SERIE 3	
ENVIRONNEMENT :	**QUANTITE OPERATEURS** :4	**TMI** : 4H00
QUALIFICATIONS MILITAIRES		
2 PERSONNELS HABILITES ET 2 MECANIQUES		

N° ORDRE	INTERVENTION	DESIGNATION DE L'OPERATION
		OPERATIONS PRELIMINAIRES
10	ND	DEMONTAGE DES CAPOTS GALIX GAUCHE ET DROIT
20	N° inter à venir	DEPOSE DES COFFRES LOT DE BORD CENTRAUX ET ARRIERES
		OPERATIONS PROPREMENT DITES
30		DEPOSE D'UN COUVERCLE
40		DEPOSE D'UN LEST
…		…

LISTE DES OUTILLAGES CUMULES		
DESIGNATION	**BLOC IDENTIFICATEUR**	**QTE**
ANNEAU DE LEVAGE MALE M10X150	F6477 5021B	2
CISAILLES POUR FEUILLARDS	F6984 1425M15	1
CLE DYNAMOMETRIQUE ¼ DE 6 A36 NM	F0541 R.203A	1
CLIQUET REVERSIBLE COURT DE ½	F0541 S.151	1
CLIQUET REVERSIBLE DE ¼	F0541 R.151	1
…	…	…

LISTE DES INGREDIENTS CUMULES		
DESIGNATION	**BLOC IDENTIFICATEUR**	**QTE**
CHIFFONS	F5309 236521	SB
FREIN FILET NORMAL	F7121 243	SB
LOCTITE MSP 5062	F7121 5062	SB
SOLVANT DEGRAISSANT	F7121 LOCTITE7063A400ML	SB
…	…	…

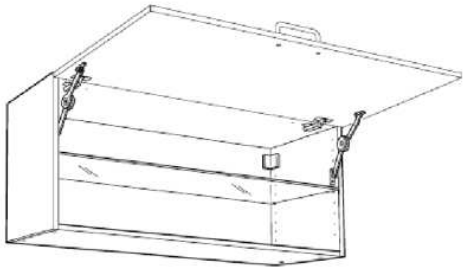
LISTE DES RECHANGES CUMULES	
DESIGNATION	**QTE**
FEUILLARD ASSEMBLE	
VIS FRAISEE HC M LG10 CL10.9 ZING. BICHRO.	
VIS H M16X200 30 A 10.9 ZING. CHRO.	
RONDELLES L 16 160HV30 ZING. CHRO	
...	

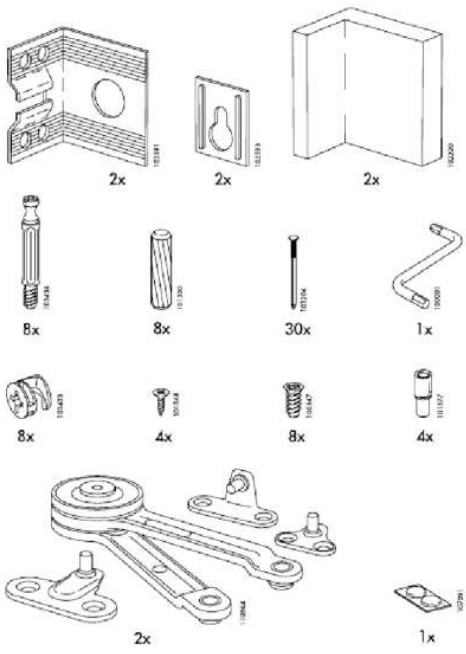
Localisation	**N°** **DESIGNATION DE L'OPERATION** - DESCRIPTIF DE LA TACHE OUTILLAGES STANDARDS OUTILLAGES SPECIFIQUES RECHANGES INGREDIENTS
	40 **DEPOSE D'UN LEST** **ATTENTION** : LE PORT DE LUNETTES EST OBLIGATOIRE - SECTIONNER LES FEUILLARDS DU BAS ET LES PLIER A L'EXTERIEUR DU CAISSON - RAFRAICHIR LES TARAUDAGES DE MISE EN PLACE DES ANNEAUX DE LEVAGE FRONTAL ET SUP. TARAUD M10X150 - VISSER DEUX ANNEAUX DE LEVAGE SUR LA PARTIE FRONTALE DU LEST ANNEAU DE LEVAGE MALE M10X150 - SECTIONNER LES FEUILLARDS ET LES PLIER A L'EXTERIEUR DU CAISSON CISAILLES POUR FEUILLARDS - DEVISSER , DEPOSER ET REBUTER UNE DES DEUX VIS INFERIEUR CLIQUET ¾ DOUILLE ¾ DE 24 - DEPLACER , EN EFFECTUANT UNE ROTATION MAITRISEE, LE LEST ET VISSER L'ANNEAU SUP - ELINGUER L'ANNEAU SUP. ELINGUE 2 BRINS NON REGLABLE DE 2000KG **ATTENTION DANGER:LES OPERATEURS NE DOIVENT PAS ETRE SITUES EN FACE DU LEST LORS DES OPERATIONS SUIVANTES** - DEVISSER , DEPOSER ET REBUTER LA 2EME VIS INF CLIQUET ¾ DOUILLE ¾ DE 24 - DEPLACER A NOUVEAU LE LEST ET VISSER L'ANNEAU SUP - ELINGUER L'ANNEAU SUPERIEUR - EXTRAIRE LE LEST A L'AIDE DU PONT - DEPOSER LE LEST, L'ELINGUE ET LES ANNEAUX - GRAISSER LES TROUS TARAUDES DE FIXATION DE ANNEAUX SUR LE LEST GRAISSE ISO 3790-AEROSOL - DEBLOQUER AU TOURNEVIS A FRAPPER LES VIS DE FIXATION DES BRIDES (SI POINT DUR RENCONTRER , PASSER AUX VIS SUIVANTES) PUIS, DEVISSER ET REBUTER LES VIS DE FIXATION DES BRIDES SUR LE CAISSON EQUIPE CLIQUET ¼ DOUILLE ¼ POUR VIS 6 PANS CREUX DE 3 TOURNEVIS A FRAPPER DOUILLE IMPACT PORTE EMBOUT EMBOUT IMPACT POUR VIS 6 PANS CREUX DE 3 MASSETTE - DEPOSER LES FEUILLARDS ET LES BRIDES

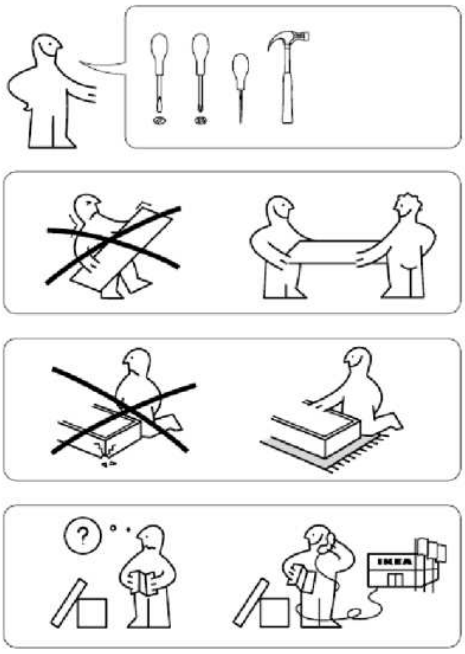
Annexe B

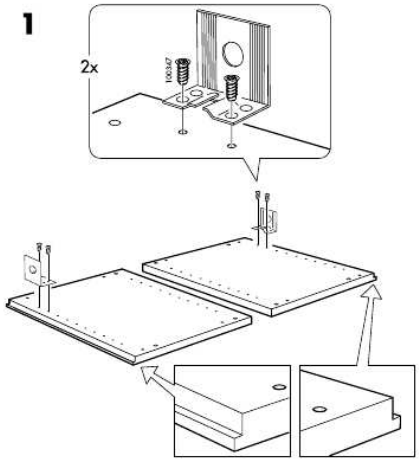
Notice de montage d’un meuble de cuisine : élément haut

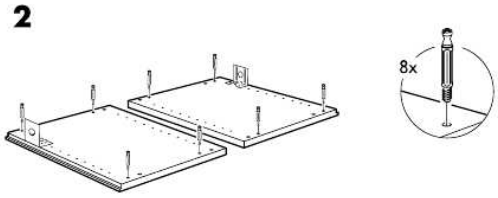
FAKTUM

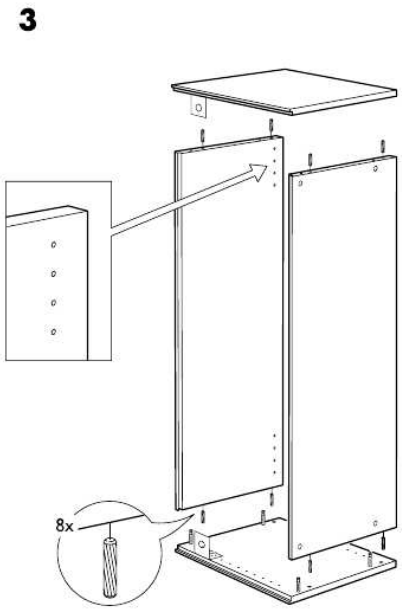


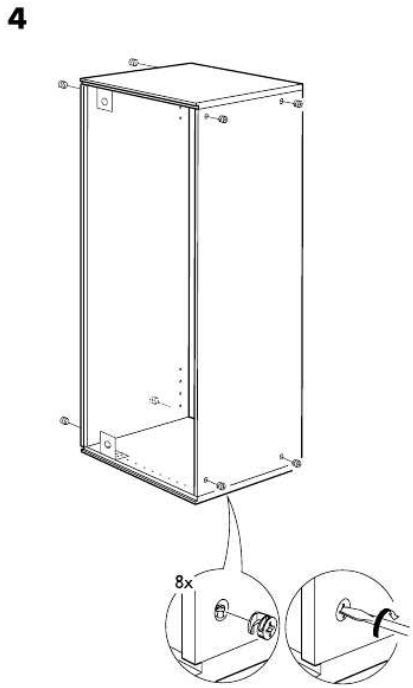


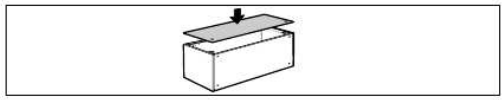


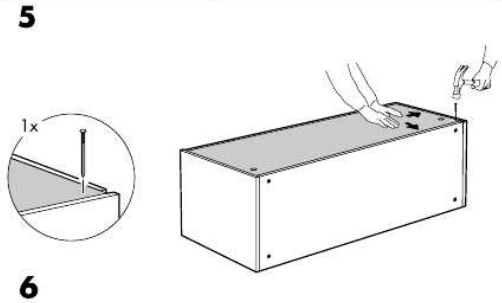

1
2x

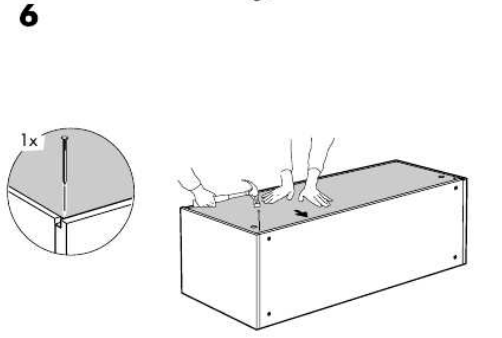

2
8x


3
8x

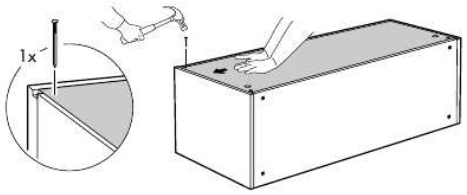

4
8x



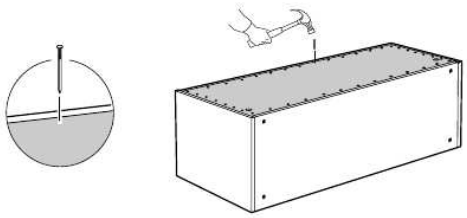

5
1x


6
1x

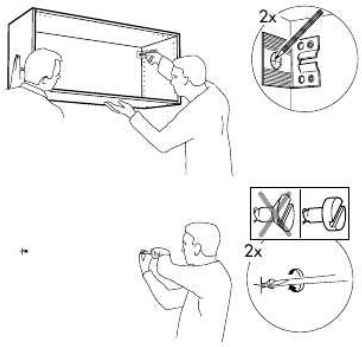
7

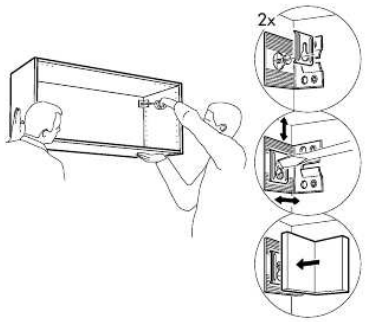

1x

8

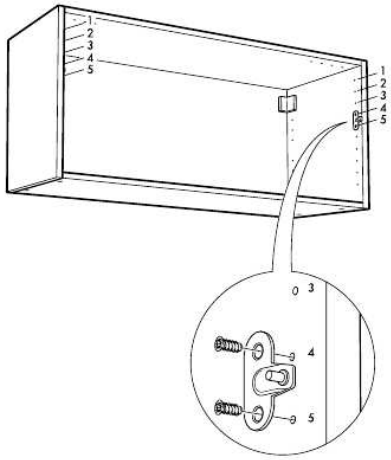


9

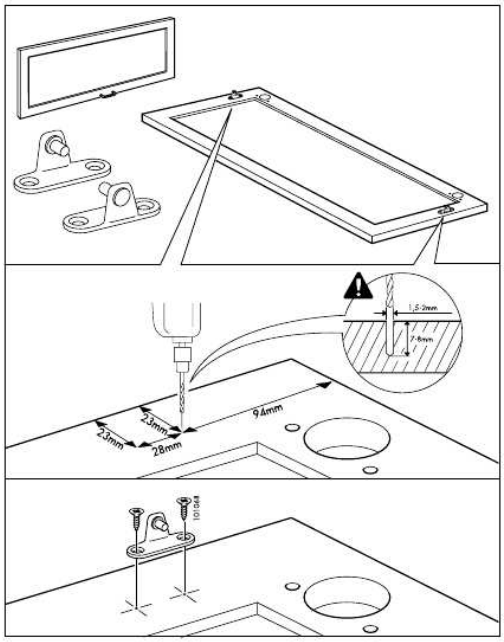

2x
2x


2x

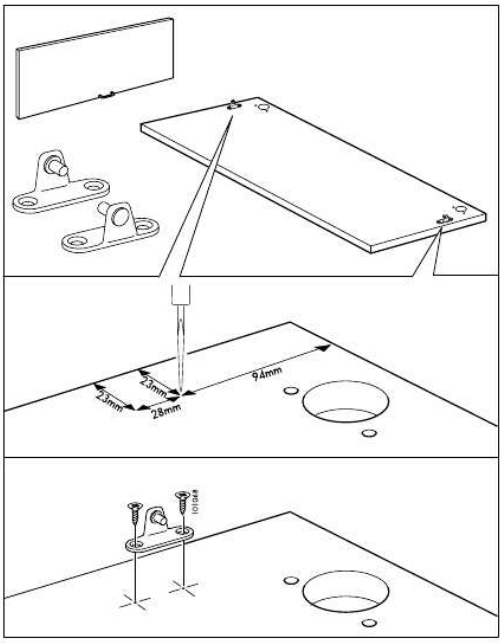
10


1
2
3
4
5
1
2
3
4
5
0 3
4
5

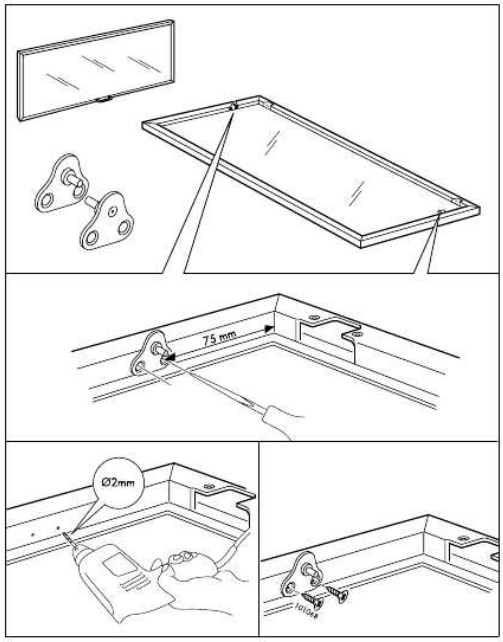
11


1,5-2mm
7-8mm
94mm
23mm
23mm
28mm
10mm

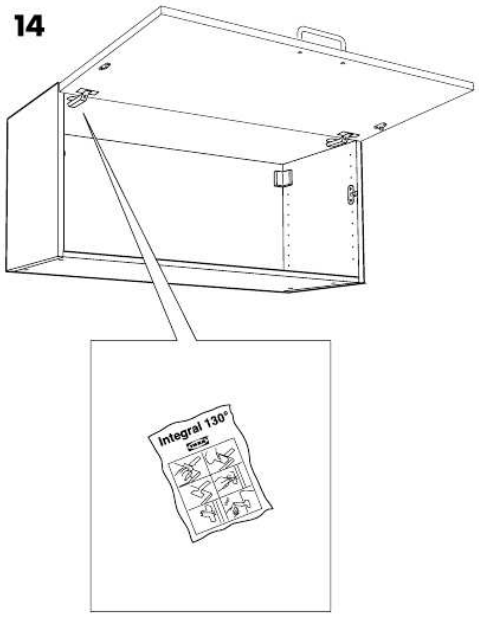
12


23mm
94mm
33mm
28mm

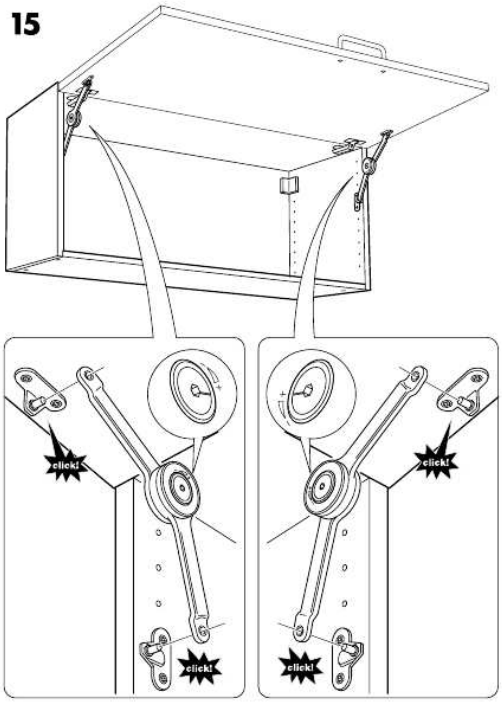
13


75 mm
Ø2mm

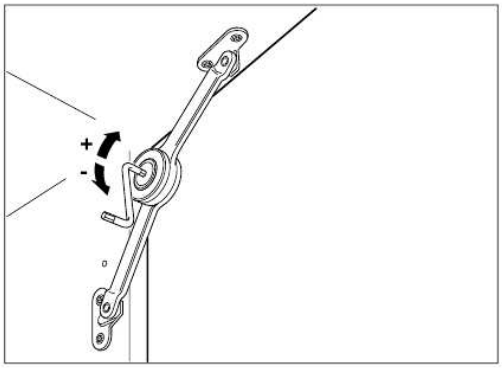
14

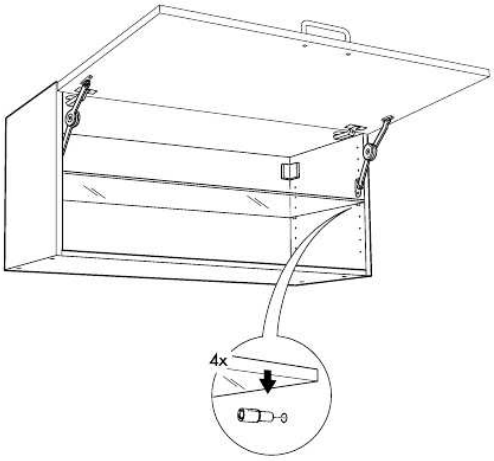

Integral 130°

15


click!
click!
click!
click!

16





Bibliographie

[AFG06] Bruno Arnaldi, Philippe Fuchs, and Pascal Guiton. *Le traité de la Réalité Virtuelle : Les applications de la Réalité Virtuelle*, volume 4, chapter Introduction à la réalité virtuelle, pages 3–30. Les Presses de l'Ecole des Mines de Paris, 3rd edition, 2006.

[All83] James F. Allen. Maintaining knowledge about temporal intervals. *Commun. ACM*, 26(11) :832–843, 1983.

[And83] John R. Anderson. *The Architecture of Cognition*. Harvard University Press, Cambridge, MA, USA, 1983.

[And95] John R. Anderson. *Learning and memory : An integrated approach*. New York : Wiley, 1995.

[BBA+00] Norman I. Badler, Rama Bindiganavale, Jan Allbeck, William Schuler, Liwei Zhao, and Martha Palmer. Parameterized action representation for virtual human agents. *Embodied conversational agents*, pages 256–284, 2000.

[BBF+95] Steve Benford, John Bowers, Lennart E. Fahlen, Chris Greenhalgh, and Dave Snowdon. User embodiment in collaborative virtual environments. In *CHI '95 : Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 242–249, New York, NY, USA, 1995. ACM Press/Addison-Wesley Publishing Co.

[BCBS97] E. Blickensderfer, J. A. Cannon-Bowers, and E. Salas. Theoretical bases for team self-correction : Fostering shared mental models. *Advances in interdisciplinary studies in work teams*, 4 :249–279, 1997.

[BD04] Marwan Badawi and Stéphane Donikian. Autonomous agents interacting with their virtual environment through synoptic objects. In *CASA 2004*, pages 179–187, 2004.

[BEL02] Norman I. Badler, C. A. Erignac, and Y. Liu. Virtual humans for validating maintenance procedures. *Commun. ACM*, 45(7) :56–63, 2002.

[BLMd06] Jean Marie Bukhardt, Domitile Lourdeaux, and Daniel Mellet-d'Huart. *Le traité de la Réalité Virtuelle : Les applications de la Réalité Virtuelle*, volume 4, chapter La conception des environnements virtuels pour l'apprentissage, pages 43–103. Les Presses de l'Ecole des Mines de Paris, 3rd edition, 2006.

[BQLC03] Cédric Buche, Ronan Querrec, P. De Loor, and Pierre Chevaillier. Mascaret : Pedagogical multi-agents system for virtual environment for training. In *CW*

'03 : Proceedings of the 2003 International Conference on Cyberworlds, pages 423–430, Washington, DC, USA, 2003. IEEE Computer Society.

[BT66] Bruce J. Biddle and Edwin J. Thomas. *Role theory : concepts and research*. New York : Wiley, 1966.

[Buc05] Cédric Buche. *Un système tutoriel intelligent et adaptatif pour l'apprentissage de compétences en environnement virtuel de formation*. PhD thesis, Université de Bretagne Occidentale, Centre Européen de Réalité Virtuelle, November 2005.

[Cao05] Sen Cao. *Role-based and agent-oriented teamwork modeling*. PhD thesis, Texas A&M University, 2005.

[CBSC90] J. A. Cannon-Bowers, E. Salas, and S. A. Converse. Cognitive psychology and team training : Training shared mental models and complex systems. *Human Factors Society Bulletin*, 33 :1–4, 1990.

[CCL03] Chih-Yueh Chou, Tak-Wai Chan, and Chi-Jen Lin. Redefining the learning companion : the past, present, and future of educational agents. *Computers & Education*, 40(3) :255–269, April 2003.

[CDD+08] Sue Cobb, Mirabelle D'Cruz, Andrew Day, Philippe David, Frédéric Gardeux, Egon L. van den Broek, Masha C. van der Voort, Frank Meijer, José Luis Izkara, and Dimitris Mavrikios. How is vr used to support training in industry ? : The intuition network of excellence working group on education and training. In *VRIC'08 : Proceedings of the 10th Virtual Reality International Conference*, pages 75–83, Laval, France, 2008.

[CKP95] James Cremer, Joseph Kearney, and Yiannis Papelis. Hcsm : a framework for behavior and scenario control in virtual environments. *ACM Trans. Model. Comput. Simul.*, 5(3) :242–267, 1995.

[CL91] Philip R. Cohen and Hector J. Levesque. Teamwork. *nous : Special Issue on Cognitive Science and Artificial Intelligence*, 25(4) :487–512, 1991.

[DA94] Stéphane Donikian and Bruno Arnaldi. Complexity and concurrency for behavioral animation and simulation. In *Fifth Eurographics workshop on animation and simulation*, pages 101–113, 1994.

[DF02] Thierry Duval and Aurélien Fenals. Faciliter la perception de l'interaction lors de manipulations coopératives en environnements virtuels 3d. In *annex of the Proceedings of IHM 2002*, pages 29–32, Poitiers, France, November 2002.

[DPPeJ04] Julie Dugdale, Bernard Pavard, Nico Pallamin, and Mehdi el Jed. Emergency fire incident training in a virtual world. In *Proceedings of the International workshop on Information Systems for Crisis Response and Management (ISCRAM 2004)*, May 2004.

[FBHH99] Mike Fraser, Steve Benford, Jon Hindmarsh, and Christian Heath. Supporting awareness and interaction through collaborative virtual interfaces. In *UIST '99 : Proceedings of the 12th annual ACM symposium on User interface software and technology*, pages 27–36, New York, NY, USA, 1999. ACM Press.

[FG98] Jacques Ferber and Olivier Gutknecht. A meta-model for the analysis and design of organizations in multi-agent systems. In *ICMAS '98 : Proceedings of the 3rd International Conference on Multi Agent Systems*, page 128, Washington, DC, USA, 1998. IEEE Computer Society.

[FGBB07] Guillaume François, Pascal Gautron, Gaspard Breton, and Kadi Bouatouch. Anatomically accurate modeling and rendering of the human eye. In *SIGGRAPH '07 : ACM SIGGRAPH 2007 sketches*, page 59, New York, NY, USA, 2007. ACM.

[FGH+02] Geoffrey A. Frank, C.I. Guinn, R.C. Hubal, M.A. Stanford, P. Pope, and D. Lamm-Weisel. Just-talk : An application of responsive virtual human technology. In *Proceedings of the Interservice/Industry Training, Simulation and Education Conference*, December 2002.

[FGV+00] Mike Fraser, Tony Glover, Ivan Vaghi, Steve Benford, Chris Greenhalgh, Jon Hindmarsh, and Christian Heath. Revealing the realities of collaborative virtual reality. In *CVE '00 : Proceedings of the third international conference on Collaborative virtual environments*, pages 29–37, New York, NY, USA, 2000. ACM Press.

[GA08] Stéphanie Gerbaud and Bruno Arnaldi. Scenario sharing in a collaborative virtual environment for training. In *VRST '08 : Proceedings of the 2008 ACM symposium on Virtual reality software and technology*, pages 109–112, Bordeaux, France, 2008.

[GHD08] Franck Ganier, Charlotte Hoareau, and Frédéric Devillers. Assessment of procedural learning with gvt, a virtual training environment for learning maintenance operations in industrial settings. In *10th Virtual Reality International Conference (VRIC'08)*, Laval, France, April 2008.

[GK96] Barbara J. Grosz and Sarit Kraus. Collaborative plans for complex group action. *Artif. Intell.*, 86(2) :269–357, 1996.

[GMA07] Stéphanie Gerbaud, Nicolas Mollet, and Bruno Arnaldi. Virtual environments for training : from individual learning to collaboration with humanoids. In *Edutainment*, pages 116–127, Hong Kong, China, June 2007.

[GMG+08] Stéphanie Gerbaud, Nicolas Mollet, Franck Ganier, Bruno Arnaldi, and Jacques Tisseau. Gvt : a platform to create virtual environments for procedural training. In *IEEE Virtual Reality Conference. VR '08*, pages 225–232, Reno, NV, USA, March 2008.

[Gra04] Fabien Gravot. *Fondation d'un planificateur robotique intégrant le symbolique et le géométrique*. PhD thesis, Université Paul Sabatier de Toulouse, 2004.

[HBSS00] Mahdi Hannoun, Olivier Boissier, Jaime S. Sichman, and Claudette Sayettat. Moise : An organizational model for multi-agent systems. In *Proceedings of the International Joint Conference, 7th Ibero-American Conference on AI, 15th Brazilian Symposium on AI (IBERAMIA/SBIA'2000)*, pages 156–165, Atibaia, SP, Brazil, November 2000.

[HEHV03] M. Hildebrand, A. Eliëns, Z. Huang, and C. Visser. Interactive agents learning their environment. In *Intelligent virtual agents*, pages 13–17. Springer, 2003.

[HG01] Mojtaba Hosseini and Nicolas D. Georganas. Collaborative virtual environments for training. In *MULTIMEDIA '01 : Proceedings of the ninth ACM international conference on Multimedia*, pages 621–622, New York, NY, USA, 2001. ACM Press.

[HGD07] Charlotte Hoareau, Franck Ganier, and Frédéric Devillers. Évaluation de l'apprentissage de procédures avec gvt, environnement virtuel d'apprentissage d'opérations de maintenance. In *deuxièmes journées de l'AFRV*, Marseille, France, November 2007.

[Ish02] Toru Ishida. Q : A scenario description language for interactive agents. *Computer*, 35(11) :42–47, 2002.

[Jen92] Nick R. Jennings. Towards a cooperation knowledge level for collaborative problem solving. In *ECAI '92 : Proceedings of the 10th European conference on Artificial intelligence*, pages 224–228, New York, NY, USA, 1992. John Wiley & Sons, Inc.

[JGM+03] Randall W. Hill Jr., Jonathan Gratch, Stacy Marsella, Jeff Rickel, William Swartout, and David R. Traum. Virtual humans in the mission rehearsal exercise system. *KI special issue on Embodied Conversational Agents*, 17(4) :5–, 2003.

[JR97] W. Lewis Johnson and Jeff Rickel. Steve : an animated pedagogical agent for procedural training in virtual environments. *SIGART Bull.*, 8(1-4) :16–21, 1997.

[JRL+98] Andrew E. Johnson, Maria Roussos, Jason Leigh, Christina Vasilakis, Craig R. Barnes, and Thomas Moher. The nice project : Learning together in a virtual world. In *VRAIS '98 : Proceedings of the Virtual Reality Annual International Symposium*, page 176, Washington, DC, USA, 1998. IEEE Computer Society.

[Kal01] M. Kallmann. *Object Interaction in Real-Time Virtual Environments*. PhD thesis, École Polytechnique Fédérale de Lausanne, 2001.

[KT98] M Kallmann and D Thalmann. Modeling Objects for Interaction Tasks. In *Eurographics Workshop on Animation and Simulation, 1998*, pages 73–86, 1998.

[LCAA08] Xavier Larrodé, Benoît Chanclou, Laurent Aguerreche, and Bruno Arnaldi. Openmask : an open-source platform for virtual reality. In *IEEE VR workshop on Software Engineering and Architectures for Realtime Interactive Systems (SEARIS)*, Reno, NV, USA, 2008.

[LD02] Fabrice Lamarche and Stéphane Donikian. Automatic orchestration of behaviours through the management of resources and priority levels. In *AAMAS'02 : Proceedings of Autonomous Agents and Multi Agent Systems*, Bologna, Italy, July 15-19 2002.

[LNR87] John E. Laird, Allen Newell, and Paul S. Rosenbloom. Soar : an architecture for general intelligence. *Artificial Intelligence*, 33(1) :1–64, 1987.

[Lou01] Domitile Lourdeaux. *Réalité virtuelle et formation : conception d'environnements virtuels pédagogiques*. PhD thesis, ENSMP, 2001.

[LRS99] Neal Lesh, Charles Rich, and Candace L. Sidner. Using plan recognition in human-computer collaboration. In *UM '99 : Proceedings of the seventh international conference on User modeling*, pages 23–32, Secaucus, NJ, USA, 1999. Springer-Verlag New York, Inc.

[LSF99] M. Leitao, A. A. Sousa, and F. N. Ferreira. A scripting language for multi-level control of autonomous agents in a driving simulator. In *DSC'99*, pages 339–351, July 1999.

[LSMC04] R. Bowen Loftin, Mark W. Scerbo, Frederic D. McKenzie, and Jean M. Catanzaro. Training in peacekeeping operations using virtual environments. *IEEE Comput. Graph. Appl.*, 24(4) :18–21, 2004.

[MA06] Nicolas Mollet and Bruno Arnaldi. Storytelling in virtual reality for training. In Zhigeng Pan, Ruth Aylett, Holger Diener, Xiaogang Jin, Stefan Göbel, and Li Li, editors, *Edutainment*, volume 3942 of *Lecture Notes in Computer Science*, pages 334–347. Springer, 2006.

[Mal97] Hanspeter A. Mallot. Behavior-oriented approaches to cognition : Theoretical perspectives. *Theory in Biosciences*, pages 196–220, 1997.

[Man01] F. Mantovani. *VR Learning : Potential and Challenges for the Use of 3D Environments in Education and Training*. IOS Press, Amsterdam, 2001.

[MAP99] David Margery, Bruno Arnaldi, and N. Plouzeau. A general framework for cooperative manipulation in virtual environments. In *Virtual Environments'99 : Proceedings of the Eurographics Workshop*, pages 169–178, Vienna, Austria,, May 31-June 1 1999.

[Mar06] Ronan Marion. Gaspar : Gestion aviation sur porte-avions par la réalité virtuelle. In *1ères journées de l'Association Fran caise de Réalité Virtuelle*, November 2006.

[Md04] Daniel Mellet-d'Huart. *De l'intention à l'attention. Contributions à une démarche de conception d'environnements virtuels pour apprendre à partir d'un modèle de l'(én)action*. PhD thesis, Université du Maine, 2004.

[MdMS+05] Daniel Mellet-d'Huart, G. Michel, D. Steib, B. Dutilleul, B. Paugam, M. Dasse, and R. Courtois. Usages de la realite virtuelle en formation professionnelle d'adultes : entre tradition et innovation. In *Proceedings of the First International VR-Learning Seminar*, Laval, France, April 20 & 21 2005.

[MGA07a] Nicolas Mollet, Stéphanie Gerbaud, and Bruno Arnaldi. An operational vr platform for building virtual training environments. In *Edutainment*, pages 140–151, Hong Kong, China, June 2007.

[MGA07b] Nicolas Mollet, Stéphanie Gerbaud, and Bruno Arnaldi. Storm : a generic interaction and behavioral model for 3d objects and humanoids in a virtual environment. In *IPT-EGVE the 13th Eurographics Symposium on Virtual Environments*, pages 95–100, July 2007.

[MKB08] Franck Multon, Richard Kulpa, and Benoit Bideau. Mkm : A global framework for animating humans in virtual reality applications. *Presence : Teleoperator and Virtual Environments*, 17(1) :17–28, 2008.

[MKMA04] Stéphane Menardais, Richard Kulpa, Franck Multon, and Bruno Arnaldi. Synchronization for dynamic blending of motions. In *SCA '04 : Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 325–335, New York, NY, USA, 2004. ACM Press.

[Mol05] Nicolas Mollet. *De l'Objet-Relation au Construire en Faisant : Application a la specification de scenarios de formation a la maintenance en Realite Virtuelle*. PhD thesis, INSA Rennes, 2005.

[MSBQ07] Nicolas Marion, Cyril Septseault, Alexandre Boudinot, and Ronan Querrec. Gaspar : Aviation management on an aircraft carrier using virtual reality. In *Cyberworlds, 2007. CW '07. International Conference on*, pages 15–22, 2007.

[MV01] Michael S. Miller and Richard A. Volz. Training for teamwork. In *The 5th World Multi-Conference on Systemics, Cybernetics and Informatics*, July 2001.

[Nar98] Alexander Nareyek. A planning model for agents in dynamic and uncertain real-time environments. In *In Proceedings of the 1998 AIPS Workshop on Integrating Planning, Scheduling and Execution in Dynamic and Uncertain Environments*, pages 7–14, Menlo Park, Ca, 1998.

[OHS+00] Jauvane C. Oliveira, Mojtaba Hosseini, Shervin Shirmohammadi, M. Cordea, Emil M. Petriu, Dorina C. Petriu, and Nicolas D. Georganas. Virtual theater for industrial training : A collaborative virtual environment. In *CSCC 2000 : Proceedings of 4th WORLD MULTICONFERENCE on Circuits, Systems, Communications & Computers*, Greece, July 2000.

[OSG00a] Jauvane C. Oliveira, X. Shen, and Nicolas D. Georganas. Collaborative virtual environment for industrial training and e-commerce. In *Proceedings of the Workshop on Application of Virtual Reality Technologies for Future Telecommunication Systems, IEEE Globecom 2000 Conference*, San Francisco, 2000.

[OSG00b] Jauvane C. Oliveira, Shervin Shirmohammadi, and Nicolas D. Georganas. A collaborative virtual environment for industrial training. In *VR '00 : Proceedings of the IEEE Virtual Reality 2000 Conference*, page 288, Washington, DC, USA, 2000. IEEE Computer Society.

[Par07] Sébastien Paris. *Caractérisation des niveaux de services et modélisation des circulations de personnes dans les lieux d'échanges*. PhD thesis, Université de Rennes I, October 2007.

[PHM+03] M. Ponder, B. Herbelin, T. Molet, S. Schertenlieb, B. Ulicny, G. Papagiannakis, N. Magnenat-Thalmann, and Daniel Thalmann. Immersive vr decision training : telling interactive stories featuring advanced virtual human simulation technologies. In *EGVE '03 : Proceedings of the workshop on Virtual environments 2003*, pages 97–106, New York, NY, USA, 2003. ACM Press.

[Pia76] Jean Piaget. *Le comportement, moteur de l'évolution*. Gallimard, Paris, 1976.

[PPD07] Sébastien Paris, Julien Pettré, and Stéphane Donikian. Pedestrian reactive navigation for crowd simulation : a predictive approach. *Computer Graphics Forum, Eurographics'07*, 26(3) :665–674, 2007.

[PT94] Ken Pimentel and Kevin Teixeira. *Virtual Reality : Through the New Looking Glass*. McGraw-Hill, Inc., New York, NY, USA, 2^{nd} edition, 1994.

[QBMC03] Ronan Querrec, Cédric Buche, E. Maffre, and Pierre Chevaillier. SécuRéVi : virtual environments for fire-fighting training. In Simon Richir, Paul Richard, and Bernard Taravel, editors, *5^{th} virtual reality international conference (VRIC'03)*, pages 169–175, Laval, France, May 2003.

[QBMC04] Ronan Querrec, Cédric Buche, E. Maffre, and Pierre Chevaillier. Multiagents systems for virtual environment for training. application to fire-fighting. *International Journal of Computers and Applications (IJCA)*, 1(1) :25–34, juin 2004.

[QC01] Ronan Querrec and Pierre Chevaillier. Virtual storytelling for training : An application to fire-fighting in industrial environment. *International Conference on Virtual Storytelling ICVS 2001*, (Vol 2197) :201–204, September 2001.

[QRC01] Ronan Querrec, P. Reignier, and Pierre Chevaillier. Humans and autonomous agents interactions in a virtual environment for fire figthing training. In *VRIC'2001*, pages 57–64, 2001.

[Que02] Ronan Querrec. *Les Systèmes Multi-Agents pour les Environnements Virtuels de Formation. Application à la sécurité civile*. PhD thesis, Université de Bretagne Occidentale, October 2002.

[Rey87] Craig W. Reynolds. Flocks, herds, and schools : A distributed behavioral model. *Computer Graphics*, 21(4) :25–34, 1987.

[RGH+06] N Rezzoug, P Gorce, A Héloir, S Gibet, JF Kamp, Franck Multon, and C Pelachaud. Virtual humanoids endowed with expressive communication gestures : the hugex project. In *Proceedings of SMC '06 : IEEE International Conference on Systems, Man, and Cybernetics*. IEEE Press, 2006.

[RJ99a] Jeff Rickel and W. Lewis Johnson. Animated agents for procedural training in virtual reality : Perception, cognition, and motor control. *Applied Artificial Intelligence*, 13(4 - 5) :343–382, May 1999.

[RJ99b] Jeff Rickel and W. Lewis Johnson. Virtual humans for team training in virtual reality. In *Proceedings of the Ninth International Conference on Artificial Intelligence in Education*, 1999.

[RJ00] Jeff Rickel and W. Lewis Johnson. Task-oriented collaboration with embodied agents in virtual worlds. pages 95–122, 2000.

[RJ03] Jeff Rickel and W. Lewis Johnson. Extending virtual humans to support team training in virtual reality. *Exploring artificial intelligence in the new millennium*, pages 217–238, 2003.

[RJL+97] Maria Roussos, Andrew E. Johnson, Jason Leigh, Christina A. Vasilakis, Craig R. Barnes, and Thomas G. Moher. Nice : combining constructionism, narrative and collaboration in a virtual learning environment. *SIGGRAPH Comput. Graph.*, 31(3) :62–63, 1997.

[RMG+02] Jeff Rickel, Stacy Marcella, Jonathan Gratch, Randall W. Hill, David Traum, and William Swartout. Toward a new generation of virtual humans for interactive experiences. *IEEE Intelligent Systems*, 17(4) :32–38, 2002.

[RMP04] Andrew Robinson, Katerina Mania, and Philippe Perey. Flight simulation : research challenges and user assessments of fidelity. In *VRCAI '04 : Proceedings of the 2004 ACM SIGGRAPH international conference on Virtual Reality continuum and its applications in industry*, pages 261–268, New York, NY, USA, 2004. ACM.

[RS97] Charles Rich and Candace L. Sidner. Collagen : when agents collaborate with people. In *AGENTS '97 : Proceedings of the first international conference on Autonomous agents*, pages 284–291, New York, NY, USA, 1997. ACM.

[SGH+05] William Swartout, J. Gratch, R. Hill, Eduard Hovy, S. Lindheim, Jeff Rickel, and David Traum. Simulation meets hollywood : Integrating graphics, sound, story and character for immersive simulation. *Multimodal Intelligent Information Presentation*, 2005.

[SGH+06] William Swartout, Jonathan Gratch, Randall W. Hill, Eduard Hovy, Stacy Marsella, Jeff Rickel, and David Traum. Toward virtual humans. *AI Mag.*, 27(2) :96–108, 2006.

[Sto01] Robert Stone. Virtual reality for interactive training : an industrial practitioner's viewpoint. *Int. J. Hum.-Comput. Stud.*, 55(4) :699–711, 2001.

[Tam97] Milind Tambe. Towards flexible teamwork. *Journal of Artificial Intelligence Research*, 7 :83–124, 1997.

[TRGM03] David Traum, Jeff Rickel, J. Gratch, and S. Marsella. Negotiation over tasks in hybrid human-agent teams for simulation-based training. In *AAMAS'03 : Proceedings of the second international joint conference on Autonomous agents and multiagent systems*, pages 441–448, New York, NY, USA, 2003. ACM Press.

[TT94] Xiaoyuan Tu and Demetri Terzopoulos. Artificial fishes : physics, locomotion, perception, behavior. In *SIGGRAPH '94 : Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, pages 43–50, New York, NY, USA, 1994. ACM Press.

[TZ07] Shang-Ping Ting and Suiping Zhou. An architecture for rapidly reconfigurable mout simulations. In *ANSS '07 : Proceedings of the 40th Annual Simulation Symposium*, pages 144–154, Washington, DC, USA, 2007. IEEE Computer Society.

[vdP93] Michiel van de Panne. Sensor-actuator networks. In *SIGGRAPH '93 : Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, pages 335–342. ACM Press, 1993.

[WL04] H. Wang and R. Li. A desktop vr prototype for industrial training applications. *Virtual Real.*, 7(3-4) :187–197, 2004.

[XVMP03] Dianxiang Xu, Richard A. Volz, Michael S. Miller, and Jesse Plymale. Human-agent teamwork for distributed team training. In *ICTAI '03 : Proceedings of the 15th IEEE International Conference on Tools with Artificial Intelligence*, page 602, Washington, DC, USA, 2003. IEEE Computer Society.

[YYI+01] John Yen, Jianwen Yin, Thomas R. Ioerger, Michael S. Miller, Dianxiang Xu, and Richard A. Volz. Cast : Collaborative agents for simulating teamwork. In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence (IJCAI-01)*, pages 1135–1142, Seattle, WA, August 2001.

[YYWP05] Chunyan Yu, Dongyi Ye, Minghui Wu, and Yunhe Pan. A role-based and agent-oriented model for collaborative virtual environment. In *Systems, Man and Cybernetics, 2005 IEEE International Conference on*, volume 2, pages 1592–1597 Vol. 2, 2005.

[ZCV06] Yu Zhang, Sen Cao, and Richard A. Volz. Towards a role-based framework in agent teams. In *Workshop of Role-Based Collaboration, The 20th International Conference in Computer Supported Cooperative Work (CSCW'06)*, 2006.

[Zha07] Yu Zhang. Modeling role-based agent team. In *The 20th Canadian Conference on Artificial intelligence (Canadian AI'07)*, pages 1–13, Montreal, Quebec, Canada, May 2007.

[ZLGB03] Xiaowei Zhong, Peiran Liu, Nicolas D. Georganas, and Pierre Boulanger. Designing a vision-based collaborative augmented reality application for industrial training. *it - Information Technology*, 45(1) :7–19, 2003.

[ZT06] Suiping Zhou and Shang-Ping Ting. A generic model framework for mout simulations. In *CW '06 : Proceedings of the 2006 International Conference on Cyberworlds*, pages 89–92, Washington, DC, USA, 2006. IEEE Computer Society.

Résumé

La formation industrielle est un domaine applicatif émergent pour la réalité virtuelle. GVT (Generic Virtual Training) est une plateforme de création d'environnements virtuels permettant une formation individuelle à des procédures industrielles de type maintenance. Cependant, les demandes industrielles évoluent et un nouveau besoin concerne la formation à des procédures collaboratives. Dans cette thèse nous proposons des modèles pour étendre les possibilités de GVT à la formation à des procédures collaboratives où des utilisateurs réels collaborent avec des humains virtuels.

Nous présentons un modèle de l'activité des acteurs permettant de remplacer dynamiquement un apprenant par un humain virtuel. Ce modèle permet à un acteur de réaliser des actions en tenant compte de ses caractéristiques, du scénario, de l'environnement ainsi que de l'activité de ses partenaires. Nous proposons également une extension du langage de scénario LORA afin de décrire un scénario collaboratif. Un tel scénario décrit l'assignation des personnes aux actions et intègre des actions collaboratives. Le scénario a également été simplifié en rendant implicites des actions basiques comme la prise et la pose d'outils. Enfin, nous présentons le mécanisme de sélection d'action que nous avons mis en place et dont l'objectif est double : permettre à un humain virtuel de sélectionner une action à réaliser et donner des conseils à l'apprenant sur l'action à choisir. Il se compose de deux parties : un module global de répartition des actions, chargé de déterminer le meilleur candidat pour chaque action du scénario, et des modules décisionnels dont dispose chaque acteur. Un acteur va ainsi se servir de son profil collaboratif (ensemble de règles de comportement) et de la suggestion du module de répartition pour choisir la prochaine action qu'il souhaiterait réaliser.

Ces différents modèles ont été intégrés à GVT au sein d'un prototype. Dans cette thèse nous présentons un scénario applicatif réalisé dans GVT qui consiste à monter collaborativement un meuble livré en kit.

Abstract

Industrial training is an emerging applicative domain for virtual reality. GVT (Generic Virtual Training) is a platform to create virtual environments for individual training on industrial procedures such as maintenance procedures. Nevertheless, industrial needs keep on evolving and a new request concerns training on collaborative procedures. In this thesis, we propose models to extend GVT possibilities to training on collaborative procedures where real users and virtual humans collaborate.

We present an activity model for the actors which allows the dynamic substitution of a real user by a virtual human. This model makes an actor perform actions depending on his characteristics, on the scenario, on the environment and also on his partners' activity. We also propose an extension to the scenario language LORA in order to describe collaborative scenarios. Such a scenario describes the assignation of people to actions and integrates collaborative actions. The scenario has been simplified while making implicit some basic actions such as to take or to put back a tool. Finally, we present the action selection mechanism we have set up. Its aim is dual : to enable a virtual human to select an action to perform and to give a trainee some pedagogical advice about the best action to choose. This mechanism has two parts : a global module of action distribution in charge of determining the best candidate for each scenario action and one decisional module per actor. Indeed, an actor uses his collaborative profile (a set of behavioral rules) and the suggestion made by the action distribution module to choose the next action to perform.

These models have been integrated to GVT in a prototype. In this thesis we present an applicative scenario which consists in collaboratively assembling a piece of furniture delivered in a kit.