Échange de Connaissances entre Utilisateurs et Agents Autonomes dans les EVFC

Mukesh Barange, Rozenn Bouville Berthelot, Pierre Chevaillier, Camille de Keukelaere, Valérie Gouranton, Alexandre Kabil, Thomas Lopez, Florian Nouviale, Bruno Arnaldi

Échange de Connaissances entre Utilisateurs et Agents Autonomes dans les EVFC

Mukesh Barange[1], Rozenn Bouville Berthelot[2], Pierre Chevaillier[1], Camille De Keukelaere[1], Valérie Gouranton[2], Alexandre Kabil[1], Thomas Lopez[2], Florian Nouviale[2], and Bruno Arnaldi[2]

[1]ENIB (UEB), Lab-STICC, France
[2]INSA-Rennes (UEB), Irisa, France

RÉSUMÉ

Cet article propose une approche innovante de la gestion et de l'échange de connaissances entre utilisateurs et agents autonomes dans les Environnements Virtuels de Formation Collaboratifs. Afin de faciliter à la fois le dialogue et la réalisation de tâches collaboratives au sein d'une équipe pouvant être composée d'utilisateurs et d'agents autonomes, nous introduisons une entité commune à ces derniers : le *Shell*. Cette entité regroupe la gestion des connaissances et le contrôle du mannequin qui lui est attaché. Les agents autonomes, basés sur une architecture nommée C-BDI, exploitent les connaissances du *Shell* auquel ils sont liés afin d'alimenter leur processus de prise de décision et d'enrichir leur dialogue, basé sur une approche Information-State. Les utilisateurs peuvent également accéder aux connaissances de leur *Shell* respectif à travers une interface utilisateur.

Keywords: Virtual Actors, Collaboration, Conversational Behaviour, Decision-Making

1 INTRODUCTION

Les Environnements Virtuels de Formation Collaboratifs (EVFC) permettent à des utilisateurs d'apprendre à réaliser une procédure collaborative. Dans ces environnements, les utilisateurs sont amenés à interagir et à travailler avec d'autres utilisateurs mais aussi avec des agents autonomes, chacun remplissant un rôle spécifique dans la procédure. Dans ce papier, nous utiliserons le terme *acteur* pour désigner indifféremment les agents autonomes ou les utilisateurs et le terme *mannequin* pour leur représentation dans le monde virtuel.

La collaboration et la coordination entre plusieurs acteurs nécessite la mise en place de deux éléments : premièrement, une connaissance et un contexte communs [5], deuxièmement, une perception et une compréhension des actions des autres [24]. Ces éléments font appel à deux notions de psychologie. Tout d'abord, le *common grounding* qui est le processus par lequel les membres d'une équipe parviennent à partager un point de vue commun sur la situation courante [8]. D'autre part, la conscience mutuelle, ou *mutual awareness*, qui signifie que les membres de l'équipe agissent de façon à obtenir des informations sur l'activité de leurs partenaires et à leur fournir des informations sur leur propre activité [6]. Afin de mettre en œuvre ces deux notions, les acteurs doivent être capables d'acquérir des connaissances sur l'évolution de l'environnement virtuel et sur l'activité des autres acteurs par la perception, la recherche active d'information ou encore le dialogue. Une façon efficace de rendre tangible pour l'apprenant les processus de *commmon grouding* et de conscience mutuelle est de permettre à l'apprenant d'échanger son rôle avec un autre acteur [15] ; et ainsi de permettre une mise en commun de connaissances entre l'utilisateur-apprenant et un partenaire de l'activité collaborative.

Par exemple, dans les EVFC, on peut imaginer qu'à un moment clé de la procédure, un agent tuteur choisisse d'échanger son rôle avec celui d'un apprenant, amenant ce dernier à enrichir son point de vue sur la situation courante et à décider d'une action pour son nouveau rôle. Cet échange de connaissance entre un utilisateur et un autre acteur, agent autonome ou autre utilisateur, requiert une architecture spécifique. Elle doit permettre à la fois le changement de contrôle des mannequins mais également proposer une gestion adaptée des connaissances préalablement acquises par les acteurs de façon à assurer la continuation de l'activité collaborative. Il convient alors de définir le type et l'organisation des connaissances associées aux acteurs, afin de supporter de manière cohérente la réalisation coordonnée de la tâche collective et les communications verbales entre les acteurs, tout en permettant de changer de rôle à tout instant.

La solution que nous proposons, nommée le *Shell*, permet (1) un contrôle identique des mannequins par les utilisateurs et les agents autonomes, (2) l'accès, par un utilisateur, aux connaissances associées au mannequin qu'il contrôle, (3) la séparation des capacités cognitives d'un agent et de la représentation de ses connaissances, (4) le dialogue en langue naturelle entre acteurs pour la coordination et la collaboration.

Dans la section 2, nous présentons l'état de l'art concernant la collaboration, les modèles de connaissance et la gestion du dialogue entre agents autonomes et utilisateurs dans le cadre des environnements de formation. La section 3 décrit les différents composants de notre architecture ainsi que leur fonctionnement. La section suivante met en situation notre solution dans un scénario d'apprentissage réel. Enfin, la section 5 résume notre contribution.

2 ÉTAT DE L'ART

Dans la majorité des Environnements Virtuels de Formation Collaboratifs, les agents autonomes sont capables de réaliser des tâches indépendantes mais visant à réaliser un objectif commun [22, 25]. Cependant, la collaboration et les interactions entre agents autonomes et utilisateurs sont souvent limitées. Certaines plateformes, comme la version collaborative de *Generic Virtual Training* (GVT) [7] prennent en compte ces problématiques de collaboration et d'interaction entre agents et utilisateurs. C'est aussi le cas de certains modèles, comme MASCARET qui permet la réalisation d'applications pédagogiques pour l'entraînement en équipe, même si l'implémentation actuelle ne gère pas ces problématiques [14].

Les agents autonomes sont généralement composés de deux parties spécifiques : le corps (*body*) et l'esprit (*mind*). Le corps peut agir et percevoir dans le monde virtuel tandis que l'esprit prend des décisions et influence l'état mental de l'acteur [10, 11]. Ce style de séparation est ainsi utilisé dans le domaine du *Storytelling* afin

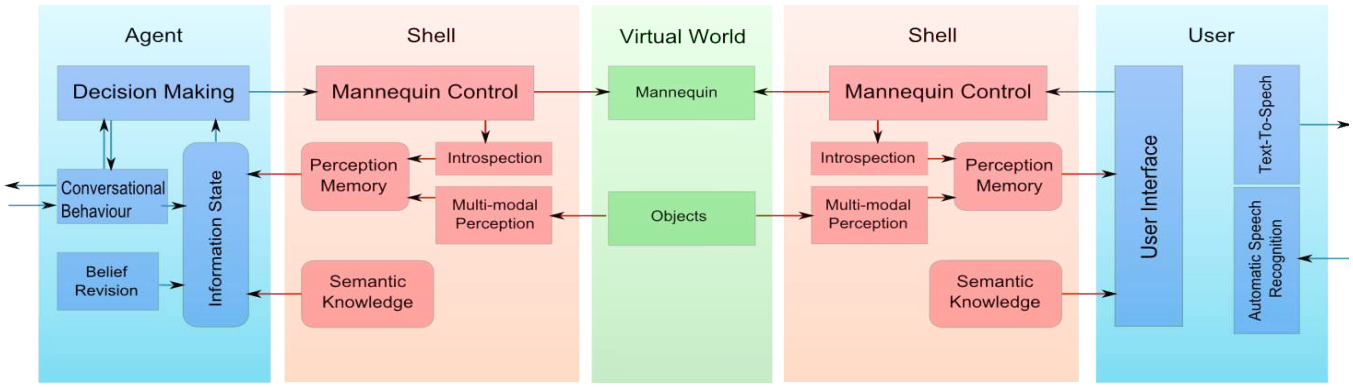


FIGURE 1: Architecture et flux de données.

de permettre le contrôle d'un personnage aussi bien par un utilisateur que par le système. Différentes méthodes ont proposé une interface unifiée pour le contrôle de personnage par les agents autonomes et les utilisateurs [13]. Cependant, les agents autonomes sont majoritairement considérés comme étant omniscients et la connaissance associée à un personnage est rarement accessible par un utilisateur qui le contrôlerait. Les représentations des connaissances d'un acteur peuvent être séparées selon deux aspects : la représentation sémantique du monde virtuel et l'état mental de l'acteur d'un côté, la représentation de la planification de la tâche de l'autre. Pour les agents conversationnels, deux approches existent pour représenter les connaissances. STEVE [22] utilise des règles de production pour représenter les connaissances sur le monde alors que Max [12] utilise des *logical frames* [16]. Ils utilisent tous les deux une structure hiérarchique pour la représentation des tâches. Aucune de ces architectures ne fournit une représentation unifiée des connaissances qui permettrait à un acteur, réel ou virtuel, de collaborer et communiquer avec d'autres acteurs afin de réaliser leur objectif.

Dans les EVFC, il est nécessaire d'intégrer des modèles de dialogue [27, 21] au modèle de tâche pour la coordination et le partage de connaissances. Plusieurs systèmes de dialogue comme TrindiKit [27] sont basés sur la mise à jour de l'*Information-State* (IS). L'IS contient l'information contextuelle sur les dialogues. Ces derniers peuvent être analysés comme des actes destinés à atteindre un but, comme pour tout comportement intentionnel. [3] a proposé une taxonomie des actes de dialogue (DIT++) et a définit leur sémantique pour la mise à jour de l'IS.

Pour guider leurs actions, les agents doivent acquérir des connaissances ou des croyances sur l'environnement et sur leurs buts, notamment par le dialogue. Pour cela, [17, 18, 28] ont proposé des modèles de gestion du dialogue basés sur l'approche BDI (*belief, desire, intention*). Ces modèles s'appliquent à des conversations orientées tâches (par exemple pour un assistant de courrier électronique [17], l'entraînement au commandement [18], le *storytelling* [28]). Cependant, dans ces modèles, les utilisateurs ne sont pas activement engagés dans la réalisation de tâches collaboratives avec d'autres partenaires virtuels.

D'autres modèles, tels que COLLAGEN [21] et GRATE* [9], sont basés sur la théorie des plans partagés [8] ou des intentions conjointes [6]. Ils sont appliqués à des systèmes multi-agents artificiels et reposent également sur les concepts de BDI. [1] adopte la théorie des plans partagés et proposent une extension du modèle BDI pour supporter la coopération entre agents, c'est à dire la capacité des agents à s'entraider pour concrétiser leurs buts. La plupart de ces approches ont mis l'accent sur la planification des communications en ignorant les autres aspects de la collaboration évoqués ci-dessus [17, 18, 28], ou ont considéré que la collaboration est guidée par les plans d'action de l'agent sans prise en compte de l'activité des autres agents [21].

L'approche BDI a donc été utilisée à la fois pour des agents conversationnels et pour la coordination des comportements au sein de systèmes multi-agents, mais de manière séparée. Le développement d'EVFC nécessite de disposer d'une architecture qui traite à la fois du comportement délibératif des agents (c'est à dire la planification des actions) et du comportement conversationnel de ces agents. Ces deux types de comportements sont guidés par des buts et doivent être contextualisés par l'activité collaborative. Ainsi, les agents peuvent échanger des connaissances avec d'autres partenaires afin d'établir un point de vue partagé sur la situation courante, ce qui leur donne la capacité de collaborer.

3 ARCHITECTURE

L'architecture que nous proposons vise à (1) séparer les acteurs, réels ou virtuels, par l'utilisation d'une entité commune, le *Shell* et à (2) fournir des comportements conversationnels pour la collaboration à travers l'utilisation d'agents *C-BDI*, issus de l'approche BDI. Comme le montre la Fig. 1, le *Shell* gère différents flux de connaissances et divise ses connaissances en deux parties : la mémoire perceptive (*Perception Memory*) et la connaissance sémantique (*Semantic Knowledge*). Ces connaissances peuvent être exploitées par les agents virtuels pour alimenter leur processus de décision et permettre le dialogue orienté tâche. De leurs côté, les utilisateurs réels peuvent parcourir les connaissances du *Shell* via une interface graphique. Une interface de dialogue, utilisant la reconnaissance vocale et la synthèse vocale, est également prévue afin de permettre la communication avec les autres acteurs.

3.1 Le *Shell*

Le *Shell* permet, en fournissant une interface commune pour les agents autonomes et les utilisateurs, d'abstraire la nature des acteurs de l'environnement virtuel. Dans ce but, le *Shell* est composé de différents modules (Fig. 1) : le module *Mannequin Control*, le module *Semantic Knowledge* et le module *Perception Memory*.

Le module *Mannequin Control* peut recevoir indifféremment des commandes provenant d'agents virtuels ou d'utilisateurs et contrôle en conséquence le mannequin qui interagit directement avec le monde virtuel. Dans [23], nous décrivons le modèle d'interaction collaboratif dont le *Shell* hérite. Ce modèle définit à la fois la façon dont un acteur contrôle le *Shell* et la façon dont le mannequin, piloté par le *Shell*, interagit avec le monde virtuel. Des acteurs de différente nature peuvent ainsi collaborer et co-manipuler des objets.

Le module *Semantic Knowledge* contient les connaissance générales et a priori, comme que le contenu de la procédure.

Le module *Perception Memory* contient des connaissances issues du monde virtuel et acquises pendant le déroulement de la procédure, à travers un système de perception multi-modale et un module d'introspection alimenté par les commandes reçues par le *Shell*.

Les connaissances du *Shell* sont variées et ont pour but d'épauler l'apprenant durant la procédure. On y retrouvera ainsi des informations sur les étapes de cette procédure, le fait qu'elles soient collaboratives ou non, mais aussi l'état et les caractéristiques du mannequin contrôlé par l'apprenant. Seront également présentes des connaissances concernant le monde virtuel, les objets, leurs positions et caractéristiques et également concernant les autres acteurs afin de favoriser la collaboration et la synchronisation.

Les connaissances du *Shell* sont divisées en quatre catégories : *Ego*, *Task*, *Team* et *World*. L'introspection, c'est à dire l'acquisition de connaissances sur l'état et les actions du mannequin lié au *Shell* permet de remplir l'*Ego*. La perception, qui peut être multimodale (dans notre cas, la vision et le dialogue) permet d'enrichir les connaissances à la fois sur le monde virtuel (*World*) lorsqu'il concerne des objets de ce dernier mais aussi sur les autres *Shells* présents (*Team*). Enfin, la catégorie *Task*, située dans la base de connaissance sémantique possède le savoir qui concerne la procédure.

Puisque les connaissances portées par le *Shell* et ses capacités d'interaction sont indépendantes de l'acteur qui le contrôle, nous avons pu définir un protocole d'échange de rôle, et donc de *Shell*, entre utilisateurs et agents autonomes.

3.2 L'architecture C-BDI

Notre architecture d'agent, nommée *C-BDI*, repose sur les principes de l'architecture BDI (*Belief, Desire, Intention*) [20]. Elle traite de manière uniforme les comportements délibératifs et conversationnels des agents, en les considérant tous deux comme des activités collaboratives dirigées par des buts. L'originalité de la solution tient en premier au rôle du dialogue, qui modifie à la fois les croyances, les désirs et les plans des agents, et en second, au caractère collaboratif de l'activité des agents. *C-BDI* repose sur les théories des croyances mutuelles et des intentions conjointes. Tout d'abord, pour coordonner leurs activités, les agents doivent partager leur intention de réaliser un but collectif. Ainsi, les agents ont la croyance mutuelle que chaque membre de l'équipe a la même intention collective d'atteindre ce but collectif. Le partage des intentions n'est pas suffisant car il ne garantit pas que les agents vont appliquer la même procédure pour atteindre le but. Pour répondre à ce problème, *C-BDI* reprend les principes de la théorie des plans partagés : les agents ont la même connaissance sémantique sur le plan global à réaliser. Cette connaissance donne aux agents des informations sur les inter-dépendances des plans individuels : définition des rôles, séquencement (partiel) des actions à réaliser par chacun, ressources à utiliser. Ainsi, chaque agent peut suivre la progression de l'activité des autres. Cette approche permet de formaliser le comportement conversationnel des agents *C-BDI* relatif à la coordination de l'activité : (1) les agents communiquent pour établir leurs croyances mutuelles, connaissance nécessaire à la réalisation de leurs actions et (2) la connaissance du plan global informe sur les dépendances entre les activités individuelles, ce qui déclenche les actions de communication.

Le processus de décision d'un agent est globalement le suivant : sur la base des connaissances du *Shell* et de ses croyances, désirs et intentions, l'agent décide de construire un plan d'actions, de réagir à la situation courante ou d'échanger des informations avec ses partenaires. La décision repose sur le contexte courant de la tâche et sur les croyances et désirs de l'agent. Les paragraphes suivants décrivent les composants de l'architecture.

3.2.1 Représentation des connaissances à base de frame.

Les différentes bases de connaissance de l'agent sont composées de *chunks*, ou *éléments de connaissances*, représentés sous forme de *frames* [16]. Cette représentation permet de traiter de manière unifiée les informations perçues, les connaissances sémantiques sur la réalisation de l'activité collaborative et les informations échangées par le dialogue. La structure des *frames* dépend du type d'information qu'elles supportent. Les concepts manipulés par les agents étant liés les uns aux autres, les *frames* définissent aussi ces liens entre éléments de connaissance, par exemple les relations type/sous-type, tout et parties, activité/actions... Des annotations linguistiques utiles à la génération des énoncés en langue naturelle à partir des connaissances sémantiques de l'agent sont également associées aux *frames* [2].

Afin de construire cette base de connaissance, les agents traitent 3 types d'information. Tout d'abord les informations concernant les ressources disponibles et potentiellement mobilisables pour la réalisation de la tâche collaborative : le type des objets, leurs propriétés, leur état et les inter-relations. Le deuxième type d'information concerne les actions réalisables par les agents, c'est à dire leurs compétences. Le troisième type permet de représenter les connaissances sur l'organisation de l'activité collaborative : les procédures et le rôle de chaque membre de l'équipe. Par exemple (Table 1), l'information sur une action réalisable par un agent est de la forme *action(name, object-resource, target-resource, mean-resource, body-resource, patient, feasability-condition, post-condition)*. *object-resource* est l'objet qui subit une transformation suite à la réalisation de l'action ; *target-resource* est l'objet qui guide l'action, ce vers quoi l'action est dirigée ; *mean-resource* est l'objet utilisé pour réaliser l'action ; *body-resource* est la partie du corps de l'agent mobilisée pour la réalisation de l'action ; *patient* est la personne pour qui l'action est réalisée ; *feasability-condition* et *postcondition* jouent leurs rôles habituels pour le raisonnement sur l'action : condition à satisfaire pour l'exécuter (réactivité) et effet attendu (pro-activité).

Semantic Knowledge	action(Lock-the-Door,door,null,key,left-hand, precondition(Bel(state(door,locked,F,any)) ; Bel(state(door,closed,T,any))),null)

TABLE 1: Représentation sous forme de *frame* de l'action "Lock-the-Door"

Le modèle de la tâche définit les connaissances de l'agent à propos du plan global d'action à réaliser et sur la structuration des conversations. Ces plans d'action sont structurés hiérarchiquement. Ces connaissances sont dans la mémoire sémantique et sont donc partagées par tous les agents.

3.2.2 Information State.

C-BDI intègre les deux processus mis en œuvre dans une activité collaborative centrée sur la réalisation conjointe d'une tâche : le processus de décision orienté-but (caractère délibératif des agents) et la communication pour la coordination de l'activité. En situation naturelle, ces deux processus sont bien entendu étroitement liés. Leur couplage s'opère via les informations sémantiques utilisées et produites par ces processus sur la base des connaissances stockées dans le *Shell* et des connaissances initiales des agents. Le schéma d'organisation des connaissances de l'agent est une extension de l'Information State (IS) de [27]. Dans [27, 3], l'IS sert au contrôle du dialogue en langue naturelle entre un utilisateur et un agent conversationnel. Nous en avons étendu l'usage afin qu'il constitue une source de connaissance partagée entre le processus de prise de décision et le comportement conversationnel : l'Information State constitue le modèle de contexte de l'agent.

Nous avons repris les différentes catégories d'information de [3] pour la gestion du dialogue. Il s'agit d'informations sur : la conversation courante (but et intention communicatives, locuteur, interlocuteurs etc.), le contexte cognitif (croyances sur soi, et sur les autres), le contexte sémantique (connaissances sur le monde et sur la tâche à réaliser), le contexte perceptuel (informations sur les propriétés des objets du monde), et le contexte social.

Dans *C-BDI*, le contexte sémantique de l'Information State est instancié à partir des connaissances sémantiques provenant du *Shell* en fonction de l'avancement de la réalisation de la tâche. Le contexte sémantique inclut également l'*agenda* qui contient les buts du dialogue. Ces buts sont ajoutés à l'agenda, entre autre, suite aux intentions communicatives générées par la réalisation collaborative de la tâche et les obligations sociales. Dans [3], l'IS définit le modèle de contexte, mais n'intègre pas les informations contextuelles sur la tâche en cours de réalisation par l'agent. De plus, dans cette proposition, les informations que l'agent peut percevoir sur son environnement, indépendamment du dialogue, ne sont pas prises en compte. Dans notre cas, ceci peut conduire à une incohérence entre le modèle du dialogue et celui de la tâche. Nous avons donc étendu le modèle de contexte de [3] en y ajoutant le contexte de la tâche et le contexte perceptif. Le contexte de la tâche (*taskContext*) contient les désirs de l'agent, ces buts relatifs à la tâche et la pile de ses intentions (*taskFocus*).

Le contexte de la tâche permet aux agents de mettre à jour leurs croyances sur l'avancement de la réalisation de la tâche collaborative par chaque membre de l'équipe. Ici, les agents ont la connaissances des buts et sous-buts qui doivent être atteints par le groupe, et des plans d'actions associés. Les agents doivent aussi savoir si les autres agents (ou un sous-ensemble de ces derniers) partagent ces mêmes buts, désirs et intentions. Nous faisons donc la distinction entre, d'une part, les buts collectifs (*group-goal*) les plans collectifs (*group-desire*), les intentions collectives (*group-intention*), et, d'autre part, les buts partagés (*joint-goal*), les plans partagés (*joint-desire*), les intentions partagées (*joint-intention*). Pour un agent, un but collectif devient un but partagé quand les agents impliqués dans sa réalisation ont fait part de leur croyance mutuelle à ce propos, c'est à dire quand l'agent sait que ce but est partagé par les autres. Le but collectif indique que l'agent sait que tous les membres de l'équipe souhaitent atteindre ce but à un moment ou à un autre. La sémantique de l'intention partagée indique que tous les membres de l'équipe ont la même intention collective et en plus, que chacun le sait. De manière similaire, les sémantiques des plans partagés et des buts partagés sont analogues.

Le partage d'une intention n'est pas suffisant pour qu'un agent s'engage dans la réalisation du plan d'action correspondant. Il faut aussi qu'il s'assure de l'engagement des autres à le réaliser. Les agents doivent donc communiquer leurs engagements partagés (*joint-commitments*). On passe alors d'un engagement individuel à réaliser un plan d'action à un engagement mutuel à agir [19]. A partir de ce moment, les agents gardent leur intention courante jusqu'à ce que le but soit atteint, à moins qu'ils n'expriment explicitement leur abandon, ce qui conduit à l'échec de la réalisation du plan d'action. Pour garantir la stabilité des intentions partagées, les agents conservent l'information sur les engagements partagés dans le *taskContext*.

Le *contexte perceptuel* contient les informations sur ce sur quoi l'agent porte son attention au cours de la conversation et de la réalisation de la tâche. Contraitement à [27], l'agent ne met pas à jour ces informations qu'à partir du dialogue, mais aussi en utilisant les informations acquises par le *Shell* dans sa mémoire perceptuelle. Ces informations sont nécessaires à la compréhension des énoncés des actes de dialogue, en particulier pour le résolution des pronoms, et pour l'instanciation contextualisée des connaissances sémantiques de l'agent.

En conclusion, dans *C-BDI*, l'Information State contient la base de croyances de l'agent ainsi que ses intentions. L'IS contient donc les informations relatives au contexte courant du dialogue, mais aussi à celui de la tâche collaborative, c'est à dire les croyances potentiellement utiles à l'agent pour sa prise de décision ainsi que sur l'*agenda* des buts de l'agent relatifs au dialogue et à l'activité collaborative.

3.2.3 Prise de décision

Le processus de décision est défini par deux fonctions : la révision des croyances et la génération des buts. Notre fonction de révision de croyance spécialise le mécanisme classique de l'architecture BDI. Elle assure la cohérence de la base des croyances alimentées par la perception de l'environnement et par le traitement des actes de dialogue reçus. De plus, lors du changement de rôle, elle permet à l'agent d'adopter de nouvelles croyances ou de les renforcer.

La génération des buts est dirigée par les informations sur les buts courants, le plan d'action global et les connaissances de l'agent (Information State et connaissances sémantiques). En particulier, les agents doivent être capable de gérer les changements de contexte. L'algorithme vérifie si l'agenda de l'IS n'est pas vide ou si la pile des *taskFocus* contient des intentions communicatives. Si c'est le cas, le contrôle passe au comportement conversationnel qui prend en charge les dialogues avec les autres membres de l'équipe. Sinon, l'agent choisit le plan à effectuer ; si l'activité nécessite une coordination avec d'autres agents, l'agent cherche à s'assurer de l'engagement des autres dans ce plan d'action. L'agent est dans une situation collaborative quand (1) il n'a pas les compétences (capacités d'action et connaissances) pour atteindre tout seul le but collectif, (2) il a les compétences nécessaires pour atteindre seul le but, mais il n'en a pas l'intention. Ce qui peut conduire un agent à adopter cette dernière posture est en dehors du cadre de ce travail.

Les situations collaboratives génèrent des intentions communicatives dans l'agenda, ce qui amène l'agent à interagir avec les autres membres de l'équipe afin d'établir leurs croyances partagées. Une fois que l'intention est générée, l'agent sélectionne les actions à réaliser et, en retour, met à jour la pile des *taskFocus* afin de maintenir sa connaissance sur le contexte de tâche.

3.2.4 Comportement conversationnel

Le comportement conversationnel permet à l'agent de partager des connaissances avec les autres agents et d'assurer la coordination de l'activité. Les productions langagières des agents sont fonction de l'activité en cours, des besoins d'information sur l'usage des ressources et sur les règles qui définissent les obligations sociales des agents (par exemple lorsqu'un agent doit utiliser une ressource partagée, il en informe les autres agents si ceux-ci sont à proximité).

La sémantique des actes de dialogue est basée sur celle de DIT++ [3]. Le comportement conversationnel des agents étant orienté vers la coordination de l'activité collaborative, ceci nous permet d'aller plus loin dans l'interprétation des énoncés, tout en utilisant moins de règles que pour un agent "généraliste". Prenons l'exemple du traitement de la question *Que vas-tu utiliser pour vérouiller la porte ?*. L'agent identifie qu'il s'agit d'une question à propos d'une action (*Que ... vérouiller la porte*), que l'on fait référence à une action à venir (temps = futur), que l'action fait partie du contexte de la tâche de l'agent qui traite le message (actor-in-focus = *tu*), et que la question porte sur la ressource utilisée pour la réaliser (*utiliser pour* + action-name). La sémantique de l'énoncé est interprété, ce qui permet d'identifier la *frame* à rechercher dans l'Information State et de construite une réponse à partir de l'élément pertinent de la *frame* : *Je vais utiliser la clé*. La génération des énoncés suit une approche pilotée par les modèles [2]. Elle combine des traits linguistiques (par exemple le genre, le nombre, le temps) avec les connaissances sémantiques relatives aux objets perçus et aux plans d'action.

Comme nous voulons que les agents aient à la fois des comportements réactifs et proactifs, nous avons défini trois protocoles de conversation collaboratifs (PCC). Ils s'expriment sous forme de règles de production contextualisées par les caractéristiques de l'activité, notamment les synchronisations entre les actions des agents et les changements de contexte. Les protocoles constituent des *désirs conversationnels* qui, lorsqu'ils sont activés, se traduisent en *intentions conversationnelles*. Les trois protocoles correspondent aux intentions suivantes :

1. L'agent qui initie une nouvelle activité collaborative, informe les autres de son intention de réaliser le but de cette activité qui devient un objectif commun pour l'équipe.
2. L'agent qui termine la dernière action d'une activité collaborative, informe ses partenaires que l'activité est maintenant terminée.
3. Quand l'agent a réalisé toutes les actions qu'il a planifiées, et que l'activité collaborative n'est pas terminée, il demande à ses partenaires de l'informer lorsqu'ils auront terminé.

Les protocoles sont activés quand le processus de décision identifie des situations collaboratives qui satisfont les conditions nécessaires à leur réalisation.

Ces trois protocoles sont complémentaires. Le premier assure que tous les membres de l'équipe construisent bien la même intention partagée au même moment ; les protocoles 2 et 3 assurent que les membres de l'équipe sont toujours engagés dans la réalisation du but partagé. Pour conclure, on voit que les intentions communicatives conduisent les agents à produire des informations utiles à la coordination de l'activité. Ces informations sont utilisées par les autres agents pour mettre à jour le contexte de la tâche. L'intérêt de la définition de ces protocoles est que l'agent peut déclencher de manière autonome des dialogues qui n'ont pas besoin d'être scriptés dans la définition des plans d'action ni de faire l'objet de règles de production ad hoc.

3.3 Protocole pour l'échange de rôle

La structure du *Shell* permet de supporter l'échange de rôle entre deux acteurs. Une première implémentation basique de ce protocole a été présentée dans [23]. Le protocole permet à deux acteurs d'échanger le mannequin qu'ils contrôlent ainsi que les connaissances qui leurs sont associées. Ce protocole présente plusieurs avantages. Premièrement, les deux acteurs engagés dans l'échange de rôle n'ont pas à se déplacer, l'échange peut se faire à distance et être observé par une tierce personne. Deuxièmement, le protocole assure un élément clé du changement de rôle en conservant la cohérence de la représentation des connaissances associées aux deux mannequins.

Pour illustrer les mécanismes qui supportent ce changement de rôle, nous présentons un petit scénario avec deux acteurs ayant respectivement les rôles R_1 et R_2. Il montre comment les connaissances embarquées dans le *Shell* de chaque agent évolue au cours du changement de rôle.

Au départ, un agent joue le rôle R_1 et un utilisateur le rôle R_2. Les deux acteurs se trouvent dans une pièce avec une porte et doivent travailler conjointement sur un moteur. Ils partagent des connaissances sémantiques sur la tâche à réaliser et sur les objets de l'environnement. En cours d'exécution, l'agent R_1 acquière des informations sur les objets dans son champ de vision et les actions réalisées par les différents membres de l'équipe. Les divergences de perception entre les deux acteurs font que les deux acteurs n'ont pas nécessairement les mêmes informations sur la situation courante. Par exemple, il se peut que l'acteur R_1 ait perçu à la date t_3 que la porte était déverrouillée alors que la dernière fois que R_2 a perçu l'état de la porte, celle-ci était verrouillée. R_1 peut aussi avoir des connaissances sur l'état du moteur que n'a pas R_2. Enfin, les deux acteurs peuvent aussi avoir la même connaissance sur l'état d'un objet (la porte est fermée). Dans cet exemple, l'état des connaissances des deux acteurs est tel que dans la Table 2.

	Actor R_1 (agent)	Actor R_2 (user)
Perception memory	state(door, locked, F, t_3) ; state(door, closed, T, t_1) ; state(engine, on, T, t_1)	state(door, locked, T, t_2) ; state(door, closed, T, t_1)
Information State	Bel(state(door,locked,F,t_3)) ; TaskContext(taskFocus(Intention("Verify-Circuit") ; Intention("Replace-Mould")))	empty

TABLE 2: État de la base de connaissance des deux acteurs avant le changement de rôle.

Si l'utilisateur demande à R_1 si la porte est verrouillée, R_2 répondra *Non*. Si l'agent pose cette question à l'utilisateur, qui est supposé être honnête et de confiance, la réponse sera *Oui* et l'agent R_1 modifiera sa base de croyances en conséquence (croyance sur l'état de la porte et sur les connaissances de R_2). De la même façon si l'utilisateur demande *Quelle sera ta prochaine action ?*, l'agent consultera la pile des *task-focus* de son IS et répondra en fonction de ses intentions courantes. La réponse sera *Je vais vérifier le circuit*. Si l'utilisateur demande ensuite *Qu'est ce que je dois faire ?*, l'agent répondra en fonction de sa connaissance sémantique sur la tâche et sur ces croyances à propos des actions en cours de réalisation par les autres acteurs.

Lors du changement de rôle, l'utilisateur prend le contrôle sur le *Shell* associé au rôle R_1 et l'agent sur celui du rôle R_2. Après cet échange, l'utilisateur peut consulter les informations déjà acquises par son nouveau *Shell* et les combiner aux siennes, ce qui peut l'amener à avoir un autre point de vue sur la situation courante. De manière analogue, l'agent qui contrôle maintenant l'autre *Shell* met à jour son IS en fonction de ses croyances et des informations stockées dans le *Shell*. Ici, l'utilisateur croyait que la porte était verrouillée, alors que le *Shell* de l'agent a une information plus récente (soit *state(door, locked, F, t_3)* avec $t_2 < t_3$). L'agent croira vraie cette information plus récente qui est supposée avoir été acquise par un agent honnête et de confiance.

Comme l'agent a des croyances et des désirs différents de ceux qu'avaient l'utilisateur, et comme sa base de croyances a été mise à jour, l'agent révise ces intentions. De ce fait, son nouveau comportement (ses nouvelles intentions) ne sera ni exactement celui qu'il aurait eu dans son ancien rôle ni celui que l'utilisateur aurait eu en continuant de jouer le même rôle. Ceci serait également vrai pour un échange entre deux agents, ce que le protocole permet aussi. En conséquence, si l'utilisateur demande à nouveau à l'agent quel sera sa prochaine action, l'agent répondra en faisant référence à sa prochaine action planifiée, qui dépend de son nouveau rôle et de ses croyances sur l'état de l'environnement.

Grâce à ce protocole, la rationalité des agents est enrichie par l'expérience de l'agent ou de l'utilisateur qui était avant associé au *Shell*. Notons que la base de croyance de chaque agent reste cohérente, ce qui est bien évidemment indispensable pour préserver la rationalité du comportement des agents.

4 APPLICATION AU SCÉNARIO D'APPRENTISSAGE

Cette section montre comment la solution que nous proposons a été appliquée à un environnement virtuel collaboratif pour l'apprentissage d'une procédure de maintenance industrielle ainsi qu'un premier retour d'expérience d'utilisateurs du système. Cette application illustre comment les processus de décision et le dialogue permettent à un agent de coordonner ses actions à celles de l'apprenant. Nous nous intéressons ici plus particulièrement à ce qui se passe lorsqu'un utilisateur et un agent échange leurs rôles en cours de procédure.

4.1 Le scénario industriel

FIGURE 2: Réalisation collaborative de la procédure de maintenance.

Le scénario d'application utilisé est une procédure de changement de moule sur une presse dans un atelier de plasturgie (Fig. 2). Notre solution a été implémentée sur la plateforme GVT, commercialisée par Nexter Training [7]. Elle enrichit cette plateforme sur plusieurs points. Tout d'abord, chaque acteur est incarné par un mannequin contrôlé par un *Shell*. L'utilisateur agit dans l'environnement en pilotant son avatar grâce à un système de tracking de son corps et de ses mains. De plus, les utilisateurs peuvent à tout moment prendre le contrôle d'un autre mannequin piloté par un agent autonome. Par ailleurs, la plateforme a également été enrichie d'une interface vocale qui utilise le système de reconnaissance et de synthèse vocale de Microsoft. La reconnaissance des énoncés de l'utilisateur et la synthèse des messages produits par les agents en langue naturelle utilisent le moteur de dialogue NabuTalk. L'utilisateur est muni d'un casque-micro sans fil pour dialoguer avec ses partenaires virtuels.

La procédure requière une coordination précise entre deux intervenants : le régleur et l'opérateur de la machine. Certaines des actions du scénario nécessitent que les deux acteurs agissent en même temps sur le même objet ou qu'un acteur réalise une action pour l'autre. De plus, les acteurs sont physiquement séparés par la machine et, à certains moments, ils ne sont plus en contact visuel. Dans ces moments, ils n'ont pas accès aux mêmes informations sur l'environnement et n'ont pas toujours de connaissances sur la réalisation effective d'une action par leur partenaire. Un des rôles est joué par l'utilisateur-apprenant et l'autre par un agent autonome. Considérons la situation où l'agent autonome joue le rôle de régleur (R_1) et l'apprenant celui d'opérateur de la machine (R_2) et plaçons-nous au moment où ils terminent le remplacement du moule. Dans ce cas, les deux acteurs ont l'intention collective de réaliser le changement du moule, mais l'agent n'a pas de croyance partagée à propos de cette intention collective comme l'illustre la Table 3. Notons que le rôle R_2 étant joué par un utilisateur, son Information State est vide : il est "dans la tête de l'utilisateur".

	Actor R_1 (agent)	Actor R_2 (user)
Perception memory	state(door, locked, F, t_3) ; state(door, closed, T, t_1) ; state(engine, on, T, t_1)	state(door, locked, T, t_2) ; state(door, closed, T, t_1)
Information State	Semantic-context(Bel(state(door,locked,F,t3))) ; TaskContext(group-intention("Replace-mould"))	empty

TABLE 3: Vue partielle de la base de connaissances des deux acteurs au début du changement de moule.

Le processus décisionnel de l'agent identifie cette situation comme un contexte d'activation du protocole conversationnel collaboratif #1 et choisit donc d'informer l'utilisateur sur le but commun. Il dit alors : *Maintenant, nous devons remplacer le moule*. La production de cet acte de dialogue ajoute une intention communicative au sommet de l'agenda (contexte sémantique de l'IS). Les connaissances des acteurs sont maintenant telles qu'indiquées dans la Table 4.

	Actor R_1 (agent)	Actor R_2 (user)
Perception memory	state(door,locked,F,t_3) ; state(door,closed,T,t_1) ; state(engine,on,T,t_1)	state(door,locked,T,t_2) ; state(door,closed,T,t_1)
Information State	Semantic-context(agenda (inform-collective-obligation Replace-Mould) ; Bel(state(door,locked,F,t_3))) ; TaskContext(group-intention ("Replace-mould"))	empty

TABLE 4: Vue partielle de la base de connaissance des deux acteurs quand le protocole 1 est initialisé.

Si l'utilisateur répond *D'accord*, l'agent met à jour son Information State. Il a une croyance partagée sur l'intention du groupe de réaliser le changement de moule. Ceci le conduit aussi à s'engager à réaliser le plan d'action permettant d'atteindre ce but et de croire en l'engagement partagé de l'utilisateur. L'agent a maintenant l'intention de réaliser "Replace-mould" qui occupe le sommet de la pile des *taskFocus* (Table 5).

	Actor R_1 (agent)	Actor R_2 (user)
Perception memory	state(door,locked,F,t_3) ; state(door,closed,T,t_1) ; state(engine,on,T,t_1)	state(door,locked,T,t_2) ; state(door,closed,T,t_1)
Information State	Semantic-context(Bel(state (door,locked,F,t_3))) ; Cognitive-context(mutual-belief (group-intention("Replace-mould"))) ; TaskContext(joint-intention ("Replace-mould") ; joint-commitment("Replace-mould") ; taskFocus(Intention("Replace- Mould")))	empty

TABLE 5: Vue partielle de la base de connaissance des deux acteurs après l'activation du protocole 1.

A ce stade, l'intention courante de l'agent est alors de vérifier le circuit d'alimentation et celle de l'utilisateur est de verrouiller la porte (Fig. 4).

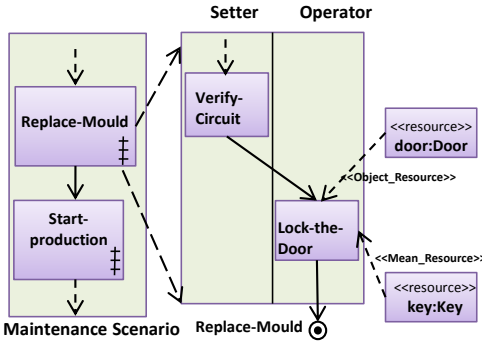
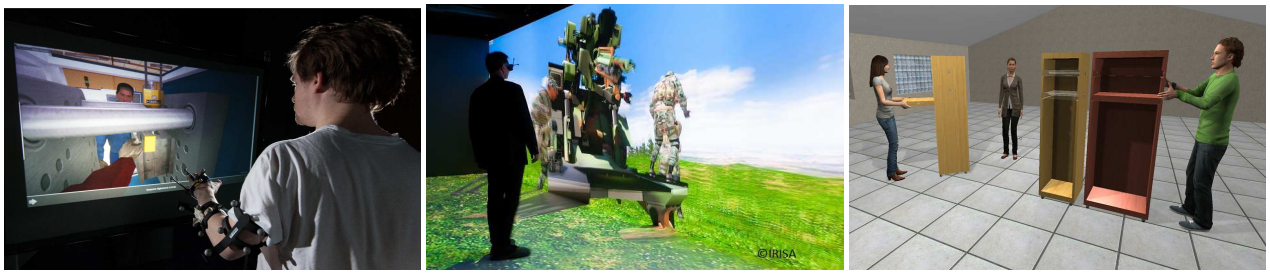


FIGURE 4: Vue partielle du plan d'action partagé entre le régleur et l'opérateur.



(a) (b) (c)

FIGURE 3: Les trois dispositifs développés dans le projet Corvette. (a) L'EVFC présenté Sect. 4.1. (b) L'apprenant se coordonne avec quatre agents autonomes. (c) Un utilisateur et trois agents autonomes organisent et planifient leurs activités.

	Actor R_1 (agent)	Actor R_2 (user)
Perception memory	state(door,locked,F,t_3) ; state(door,closed,T,t_1) ; state(engine,on,T,t_1)	state(door,locked,T,t_2) ; state(door,closed,T,t_1)
Information State	Semantic-context(Bel(state (door,locked,F,t_3))) ; Cognitive-context(mutual-belief (group-intention("Replace-mould"))) ; TaskContext(joint-intention ("Replace-mould") ; joint-commitment("Replace-mould") ; taskFocus(Intention("Verify-Circuit") ; Intention("Replace-Mould")))	empty

TABLE 6: Vue partielle de la base de connaissance des deux acteurs avant le changement de rôle pendant le remplacement du moule.

La Table 6 présente une partie des informations en mémoire des acteurs. A ce point, la conversation entre les acteurs est la suivante : *Utilisateur : Que vas-tu faire maintenant ?* ; *Agent : Je vais vérifier le circuit*.

Maintenant, les deux acteurs opèrent un échange de rôle en prenant chacun le contrôle du *Shell* de l'autre. Chacun met à jour ses croyances comme décrit en section 3.3 (les différents instants sont tels que $t_1 < t_2 < t_3$).

	Actor R_1 (user)	Actor R_2 (agent)
Perception memory	state(door,locked,F,t_3) ; state(door,closed,T,t_1) ; state(engine,on,T,t_1)	state(door,locked,F,t_3) ; state(door,closed,T,t_1) ; state(engine,on,T,t_1)
Information State	empty	Semantic-context(Bel(state (door,locked,F,t_3))) ; Cognitive-context(mutual-belief (group-intention("Replace-mould"))) ; TaskContext(joint-intention("Replace-mould") ; joint-commitment("Replace-mould") ; taskFocus(Intention("Lock-the-Door") ; Intention("Replace-Mould"))

TABLE 7: Vue partielle la base de connaissance des deux acteurs après le changement de rôle pendant le remplacement du moule.

La Table 7 donne une image des informations en mémoire chez les deux acteurs après le changement de rôle. Ici, l'apprenant obtient une information sur l'état de la machine. Il sait aussi que son partenaire avait cette information, ce qui lui apporte un éclairage nouveau sur l'activité de son partenaire. De son côté, l'agent décide de verrouiller la porte. Si l'utilisateur demande à nouveau *Que vas-tu faire ?* l'agent répond *Je vais verrouiller la porte*.

Les deux acteurs réalisent ensuite les actions qui correspondent à leurs intentions. A ce stade, l'agent n'a plus d'autre action à réaliser dans le contexte courant (Fig. 4) : l'agent a terminé la dernière action "Lock-the-Door" de l'activité *Replace-Mould*. L'application du protocole conversationnel collaboratif (protocole-2) ajoute à l'agenda de l'Information State de l'agent la production d'un acte de dialogue qui a pour fonction d'informer les autres agents de la fin de l'activité collaborative courante (Table 8). Il annonce alors *Nous avons terminé de remplacer le moule*.

	Actor R_1 (user)	Actor R_2 (agent)
Information State	empty	Semantic-context(agenda (Inform(goal-achieved,teammates))) ; TaskContext(joint-intention("Replace-mould") ; joint-commitment("Replace-mould") ; taskFocus(Intention("Replace-Mould")))

TABLE 8: Information State des acteurs après la dernière action du changement de moule.

4.2 Retour d'expérience

Dans le cadre du projet ANR Corvette, notre solution a été implémentée dans deux applications. La première est un EVFC qui implémente différents scénarios collaboratifs : celui présenté dans le section précédente (Fig. 3a) et un autre qui implique cinq acteurs devant collaborer pour manipuler un système mécanique complexe (Fig. 3b). La deuxième application est un dispositif expérimental pour l'analyse de la coordination entre des utilisateurs et des humains virtuels (Fig. 3c).

Le premier scénario éducatif (Fig. 3a et Sec. 4.1) a été présenté au salon Laval Virtual 2013. Pour favoriser l'immersion de l'utilisateur, nous avons utilisé un système ART et un casque audio sans fil. Ce scénario a été testé sur une vingtaine de personnes, expertes ou novices dans l'usage de dispositifs de réalité virtuelle. Nous avons eu des retours globalement très positifs des utilisateurs. Ils ont été capables de réaliser la procédure et ont perçu le dialogue en langue naturelle avec leur partenaire virtuel intuitif et pertinent, et par conséquent, utile. Le deuxième scénario (Fig. 3b) a montré la capacité de notre solution à traiter des scénarios multi-acteurs ainsi que le protocole d'échange de rôle.

La seconde application (Fig. 3c) met l'accent sur le comportement délibératif et conversationnel des agents. Ici, l'activité n'est pas complètement organisée et les agents doivent donc calculer dynamiquement leur plan d'action en fonction de ce que font les autres. L'utilisateur est libre de ses actions et les agents doivent mettre en jeu différents protocoles de conversation collaboratifs pour assurer la coordination de l'activité.

5 DISCUSSION ET CONCLUSION

L'architecture que nous avons développée fournit aux utilisateurs d'un environnement virtuel collaboratif, tel qu'un EVFC, la possibilité d'interagir en langue naturelle avec des humains virtuels autonomes. Elle permet également à l'utilisateur de changer de rôle

en cours de procédure grâce à une entité, nommée le *Shell*, et à un protocole d'échange qui maintient la cohérence de la base de connaissance des acteurs qu'ils soient contrôlés par un utilisateur ou bien qu'ils s'agissent d'un agent autonome. Ces capacités favorisent le partage de connaissances entre les membres d'une l'équipe et facilite la bonne coordination de l'activité collective.

Notre modèle repose sur une architecture *C-BDI* qui est ancrée à la fois dans la théorie des intentions conjointes et dans celle des croyances mutuelles. Cette architecture supporte deux processus : le comportement délibératif de réalisation de l'activité collective et la planification des dialogues. Ces processus utilisent les connaissances acquises par le *Shell* et les croyances de l'IS. De plus, les intentions des agents de réaliser leurs plans d'action génèrent des intentions conversationnelles qui, en retour, peuvent modifier la planification des actions. Nous assurons ainsi le couplage entre les deux processus.

Notre solution se base sur les fonctions communicatives des actes de dialogue définies dans DIT++. Nous n'avons pas repris l'ensemble des actes de dialogue de DIT++ mais nous nous sommes focalisé sur ceux correspondant aux fonctions de portée générale et à celles relatives aux obligations sociales. Nous avons étendu la bibliothèque des actes de dialogue relatifs au transfert d'information en les spécialisant pour coordination de l'activité collaborative. Ceci est possible car nous nous appuyons sur un modèle d'activité collaborative et que les agents agissent dans une environnement virtuel informé.

Nous avons obtenu un premier retour d'expérience positif, mais nous n'avons pas encore achevé la phase d'évaluation de la solution. Les implémentations réalisées montrent la faisabilité de notre proposition et permet déjà de traiter bon nombre de situations. Nous avons abordé le problème des dialogues multi-parties identifié dans [26]. Nous travaillons aussi sur l'enrichissement des protocoles conversationnels collaboratifs en introduisant des connaissances supplémentaires sur les ressources partagées et sur les règles organisationnelles. Ceci dotera les agents d'un répertoire plus étendu d'actions collaboratives et de prise d'information, capacités mises en œuvre dans les activités à buts partagés [4].

REMERCIEMENTS

Ces recherches ont été soutenues par l'Agence Nationale de la Recherche dans le cadre du projet CORVETTE (ANR-10-CONTINT-CORD-012).

RÉFÉRENCES

[1] D. Ancona and V. Mascardi. Coo-BDI : Extending the BDI model with cooperativity. In J. Leite, A. Omicini, L. Sterling, and P. Torroni, editors, *DALT-03*, LNAI, pages 109–134, Melbourne, 2004.

[2] M. Barange and P. Chevaillier. Model-based approach for natural language generation from semantic virtual environment. In *WACAI 2012*, pages 111–118, Grenoble, France, 2012.

[3] H. Bunt. The semantics of dialogue acts. In *Proc. of the 9th Int. Conf. on Computational Semantics*, IWCS '11, pages 1–13, Stroudsburg, PA, USA, 2011.

[4] W. J. Clancey. Simulating activities : relating motives, deliberation, and attentive coordination. *Cognitive Systems Research*, 3 :471–499, 2002.

[5] H. H. Clark and E. F. Schaefer. Contributing to discourse. *Cognitive Science*, 13 :259–294, 1989.

[6] P. Cohen and L. H.J. Teamwork. In *Nous*, volume 25, 1991.

[7] S. Gerbaud, N. Mollet, F. Ganier, B. Arnaldi, and J. Tisseau. GVT : a platform to create virtual environments for procedural training. In *IEEE Virtual Reality*, pages 225–232, Reno Etats-Unis, 2008.

[8] B. J. Grosz and S. Kraus. Collaborative plans for complex group action. *Artificial Intelligence*, 86(2) :269 – 357, 1996.

[9] N. R. Jennings. Controlling cooperative problem solving in industrial multi-agent systems using joint intentions. *Artif. Intell.*, 75(2) :195–240, June 1995.

[10] P. Kenny, A. Hartholt, J. Gratch, W. Swartout, D. Traum, S. Marsella, and D. Piepol. Building interactive virtual humans for training environments. In *The Interservice/Industry Training, Simulation & Education Conference*, 2007.

[11] J. Lee, D. DeVault, S. Marsella, and D. Traum. Thoughts on fml : Behavior generation in the virtual human communication architecture. *Proceedings of FML*, 2008.

[12] N. Leß mann, S. Kopp, and I. Wachsmuth. *Situated Interaction with a Virtual Human - Perception, Action, Cognition*, pages 287–323. 2006.

[13] I. Machado, P. Brna, and A. Paiva. Tell me a story. *Virtual Reality*, 9(1) :34–48, 2005.

[14] N. Marion, C. Septseault, A. Boudinot, and R. Querrec. Gaspar : Aviation management on an aircraft carrier using virtual reality. In *International Conference on Cyberworlds, CW'07.*, pages 15–22, 2007.

[15] M. Marks, M. Sabella, C. Burke, and Z. S.J. Performance implications of leader briefings and team-interaction training for team adaptation to novel environments. *Journal of Applied Psychology*, 87 :3–13, 2002.

[16] M. Minsky. A framework for representing knowledge. Technical report, MIT, USA, 1974.

[17] A. Nguyen and W. Wobcke. An agent-based approach to dialogue management in personal assistants. In *Proc. of the 10th int. conf. on Intelligent user interfaces*, IUI '05, pages 137–144, New York, NY, USA, 2005. ACM.

[18] J. Oijen, W. Doesburg, and F. Dignum. Goal-based communication using BDI agents as virtual humans in training : An ontology driven dialogue system. In *Agents for Games and Simulations II*, volume 6525 of *LNCS*, pages 38–52. Springer Berlin Heidelberg, 2011.

[19] P. Panzarasa, N. R. Jennings, and T. J. Norman. Formalizing collaborative decision-making and practical reasoning in multi-agent systems. *Journal of Logic and Computation*, 12(1) :55–117, 2002.

[20] A. S. Rao and M. P. Georgeff. BDI agents : From theory to practice. In *1st int. conf. on multi-agent systems*, pages 312–319, 1995.

[21] C. Rich, C. L. Sidner, and N. Lesh. Collagen : applying collaborative discourse theory to human-computer interaction. *AI Mag.*, 22(4) :15–25, Oct. 2001.

[22] J. Rickel and W. L. Johnson. *Virtual Humans for Team Training in Virtual Reality*. 1999.

[23] A. Saraos Luna, V. Gouranton, and B. Arnaldi. Collaborative Virtual Environments For Training : A Unified Interaction Model For Real Humans And Virtual Humans. In *Learning by Playing. Game-based Education System Design and Development*, pages 1–12, Allemagne, Sept. 2012.

[24] K. Schmidt. The problem with 'awareness' : Introductory remarks on awareness in CSCW. *Computer Supported Cooperative Work*, 11(3) :285–298, 2002.

[25] K. Sycara and G. Sukthankar. Literature review of teamwork models. Technical Report CMU-RI-TR-06-50, Robotics Institute, Pittsburgh, PA, 2006.

[26] D. Traum. Issues in multiparty dialogues. In F. Dignum, editor, *Advances in Agent Communication*, volume 2922 of *L.N.C.S*, pages 201–211. Springer Berlin, 2004.

[27] D. Traum and S. Larsson. The information state approach to dialogue management. In J. Kuppevelt and R. Smith, editors, *Current and New Directions in Discourse and Dialogue*, volume 22 of *Text, Speech and Language Technology*, pages 325–353. Springer Netherlands, 2003.

[28] W. Wong, L. Cavedon, J. Thangarajah, and L. Padgham. Flexible conversation management using a bdi agent approach. In *Proc. of IVA'12*, pages 464–470, Berlin, Heidelberg, 2012. Springer-Verlag.