MEMORIA, un Modèle de rEprésentation de la MémOire de l'appRenant pour les systèmes tutoriels Intelligents et Adaptatifs

Joanna Taoum

THESE DE DOCTORAT DE

L'ECOLE NATIONALE D'INGENIEURS DE BREST
COMUE UNIVERSITE BRETAGNE LOIRE

ECOLE DOCTORALE N° 601
Mathématiques et Sciences et Technologies de l'Information et de la Communication
Spécialité : *informatique*

Par
Joanna TAOUM

MEMORIA, un **M**odèle de r**E**présentation de la **M**ém**O**ire de l'app**R**enant pour les systèmes tutoriels **I**ntelligents et **A**daptatifs

Thèse présentée et soutenue à Plouzané, le 18 Décembre 2018
Unité de recherche : Lab-STICC, CNRS UMR 6285

Rapporteurs avant soutenance :

Bernard BLANDIN	Directeur de recherche, CESI Montpellier
David BAUDRY	Directeur de recherche, CESI Mont-Saint-Aignan

Composition du Jury :

Président :	Jean-Pierre JESSEL	Professeur des Universités, Université Toulouse III
Examinateurs :	Bernard BLANDIN	Directeur de recherche, CESI Montpellier
	David BAUDRY	Directeur de recherche, CESI Mont-Saint-Aignan
	Dominique LENNE	Professeur des Universités, Université de Technologie de Compiègne
Directeur de thèse :	Ronan QUERREC	Professeur des Universités, École Nationale d'Ingénieurs de Brest
Co-encadrant :	Elisabetta BEVACQUA	Maître de conférences, École Nationale d'Ingénieurs de Brest

Remerciements et Dédicace

Remerciements

« La gratitude est le secret de la vie. L'essentiel est de remercier pour tout ».
(*Albert Schweitzer*)

Je tiens à remercier tous ceux qui ont cru en moi dès le début et ceux qui ont partagé ainsi leurs connaissances et leurs expériences avec moi.

À tous ceux qui m'ont accompagné, aidé, corrigé, soutenu, encouragé, félicité et « supporté », de près ou de loin, je vous suis profondément reconnaissante pour ce que vous avez fait pour moi.

Une chose est sure : je n'oublierais jamais.

Dédicaces

À l'homme de ma vie, ma mère, ma sœur, Nicole, Landy et Anthony

Je vous suis redevable pour votre amour, votre soutien et vos encouragements. Sans vous, je n'aurais pas pu réussir.

Merci ...

Table des matières

I Introduction et État de l'art

1 Introduction

Sommaire

1.1 Contexte

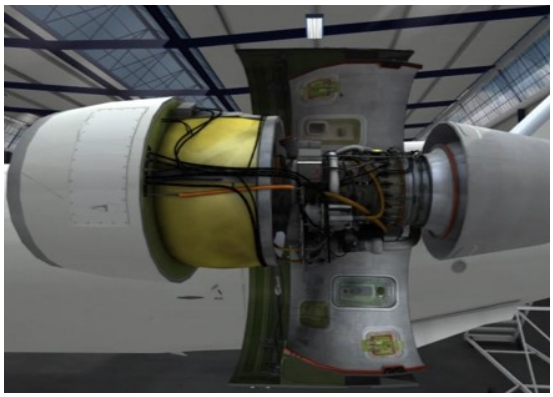
La réalité virtuelle offre de nombreuses opportunités pour le milieu éducatif. Cette technologie s'est positionnée comme l'une des techniques les plus prometteuses pour changer et améliorer l'éducation. Les technologies actuelles permettent d'imaginer un déploiement pour le grand public. Google, dans son Daydream Labs [1] a par exemple réalisé récemment une expérimentation afin de valider l'intérêt de la réalité virtuelle pour apprendre à utiliser une machine à espresso par la réalité virtuelle (Figure 1.1). Pour mieux comprendre l'impact de la réalité virtuelle sur l'apprentissage, l'équipe a mené une expérience en comparant deux groupes. Le premier groupe a utilisé un casque de réalité virtuelle et le deuxième groupe des vidéos YouTube. L'équipe a constaté que les utilisateurs apprenaient plus vite et mieux en réalité virtuelle. Le nombre d'erreurs commises et le temps nécessaire à la préparation d'un espresso étaient significativement plus faibles pour les utilisateurs formés en réalité virtuelle. Ceci est cohérent avec des recherches antérieures qui ont montré que les environnements virtuels dans les applications éducatives, également appelés les environnements virtuels d'apprentissage humain, semblent avoir une influence positive sur l'apprentissage (Bailenson et al. 2008).

1. https://www.blog.google/products/daydream/daydream-labs-teaching-skills-vr/

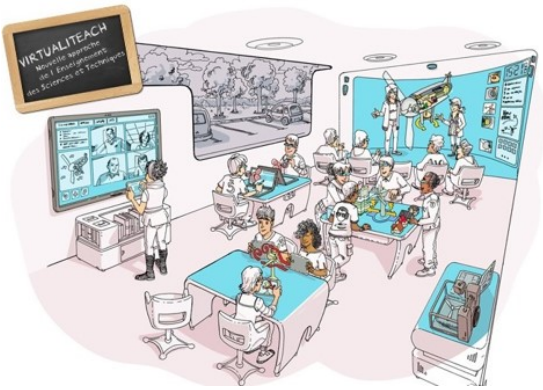
FIGURE 1.1 – Environnement virtuel construit par Daydream Labs pour apprendre à utiliser une machine à espresso.

Dans le domaine de la formation, les environnements virtuels pour l'apprentissage humain permettent l'acquisition de différents types de connaissances et de compétences. Ces environnements peuvent être utilisés dans le domaine professionnel, pour apprendre des procédures et des gestes techniques ou dans la formation initiale pour l'acquisition de savoirs tels que les langues et la physique.

De nos jours, de plus en plus d'entreprises commerciales proposent des logiciels liés à la formation en environnement virtuel. Par exemple, la société Virtualys [2] propose un outil de formation en environnement virtuel pour l'apprentissage de procédures de maintenance aéronautique (Figure 1.2a). La société Clarte [3] propose d'autres types de formations en environnement virtuel, liés à l'apprentissage initial à l'école (Figure 1.2b).



(a) Moteur d'avion en environnement virtuel.



(b) Configuration d'une classe utilisant des dispositifs de réalité virtuelle.

FIGURE 1.2 – Exemples de produits commerciaux pour la formation en réalité virtuelle.

Dans le cadre de ce travail de recherche, nous nous intéressons à l'apprentissage

2. https://www.virtualys.fr/
3. https://www.clarte-lab.fr/

dans le domaine industriel et plus particulièrement à l'apprentissage des activités humaines sur des systèmes techniques industriels tels que les procédures d'utilisation ou de maintenance.

1.2 Problématique

Nous considérons, comme (Blandin et Querrec 2014), que le seul fait de placer un apprenant dans un environnement virtuel ne suffit pas à son apprentissage, mais que le système doit proposer des fonctionnalités pédagogiques. La définition de ces fonctionnalités pédagogiques est une activité d'ingénierie, qui permet d'orchestrer les activités d'apprentissage et de définir les ressources utiles à l'atteinte d'un objectif pédagogique (Hotte et al. 2007). L'orchestration de ces activités est souvent organisée par des scénarios pédagogiques. Selon Koper, un scénario pédagogique est composé de cinq éléments principaux (Koper et Van Es 2004) :

- objectif pédagogique,
- pré-requis pédagogique,
- activité pédagogique,
- organisation pédagogique,
- environnement pédagogique.

En général, l'« objectif pédagogique » peut être exprimé selon deux niveaux. Le premier porte sur les compétences à atteindre lors de la réalisation du scénario pédagogique. Pour exprimer ces compétences, différentes taxonomies ont été proposées. Par exemple, la taxonomie de Bloom (Bloom et al. 1956) porte sur des compétences cognitives telles que comprendre, appliquer, évaluer, etc. Celle de Krathwohl (Krathwohl et al. 1964) porte sur des objectifs affectifs et sociaux tels que aider, assister, encourager, etc. Enfin, la taxonomie de Jewett (Jewett et al. 1971) porte sur les compétences psychomotrices telles que reproduire, régler, synchroniser, etc. Le deuxième niveau définit les objectifs de réalisation, c'est-à-dire l'état de la situation pédagogique que l'apprenant doit atteindre pour résoudre un problème particulier.

Les « pré-requis pédagogiques » peuvent être constitués d'une liste d'objectifs pédagogiques déjà atteints par l'apprenant ou son positionnement dans un curriculum prédéfini.

Les « organisations pédagogiques » décrivent les différents rôles intervenants dans la situation pédagogique. Dans notre contexte, les procédures à apprendre peuvent être collaboratives et donc faire intervenir plusieurs apprenants. Dans un scénario pédagogique, il est également possible de définir plusieurs rôles pédagogiques qui peuvent être déterminés en fonction des stratégies pédagogiques préférées des apprenants.

Dans notre travail, l'« environnement pédagogique » est essentiellement centré sur l'environnement virtuel qui représente le système industriel, objet de la formation. Cepen-

dant, afin d'améliorer l'apprentissage, il est possible de faire intervenir, dans cet environnement virtuel, des ressources pédagogiques externes telles que des vidéos présentant la réalisation d'une procédure ou des schémas décrivant la structure du système.

Les « activités pédagogiques » peuvent être de deux niveaux. Il peut s'agir de l'enchaînement d'activités macroscopiques, par exemple, l'enchaînement d'exercices, souvent décrit dans un curriculum. Mais il peut également s'agir de l'enchaînement des actions pédagogiques qu'un agent autonome, jouant un rôle de tuteur, peut réaliser afin de favoriser l'apprentissage au cours de l'exercice. Dans le cadre de notre travail, c'est ce dernier niveau qui nous intéresse.

Notre domaine d'application est l'apprentissage procédural pour les systèmes industriels. Selon Anderson (Anderson 1983), l'apprentissage de procédure est considéré comme un apprentissage complexe et nécessite l'utilisation de la pratique (répétition). Ceci signifie qu'un apprenant doit non seulement comprendre le système et les actions à réaliser dans la procédure mais il doit également répéter plusieurs fois l'exécution de la procédure. L'apprenant va ainsi réaliser plusieurs fois le même scénario pédagogique. Cependant, ce scénario reste général. Il peut être efficace au début de l'apprentissage (lors des premières répétitions), mais pas lors les répétitions suivantes. En effet, chaque apprenant évolue différemment lors des répétitions et applique différentes stratégies d'apprentissage. Par exemple, certains apprenants tentent de réaliser les actions sans demander de l'aide au tuteur au risque de commettre des erreurs. À l'inverse, d'autres apprenants préfèrent mémoriser complètement l'enchaînement des actions de la procédure et donc demander de l'aide au tuteur pour ne pas commettre des erreurs. Le même scénario pédagogique, avec le même objectif pédagogique, dans le même environnement pédagogique, avec la même organisation et avec le même enchaînement d'actions pédagogiques ne convient pas à tous les apprenants lors de toutes les répétitions. Il est donc important d'adapter l'exécution de ces scénarios pédagogiques en fonction de l'évolution de chaque apprenant.

1.3 Hypothèse et Approche

L'hypothèse que nous émettons dans ce travail de recherche est que l'adaptation de l'exécution d'un scénario pédagogique, en fonction de l'évolution des compétences de l'apprenant au cours des répétitions, devrait améliorer les performances d'apprentissage.

Notre objectif est ainsi de concevoir un tuteur virtuel adaptatif, capable d'adapter et d'enrichir l'exécution d'un scénario pédagogique en tenant compte de l'évolution des performances de l'apprenant en temps réel pour l'apprentissage de procédure.

Pour réaliser cette adaptation, le tuteur virtuel doit être capable de raisonner sur les connaissances représentées dans l'environnement virtuel. Ces connaissances portent non seulement sur les connaissances liées au métier mais également sur les connaissances liées aux scénarios pédagogiques. Différents modèles ont été proposés afin de

rendre explicites ces connaissances (Chevaillier et al. 2012 ; Claude, Gouranton, Berthelot et al. 2014 ; Lourdeaux et al. 2017). L'explicitation de ces connaissances permet au tuteur virtuel de raisonner sur la situation pédagogique afin de l'adapter à l'apprenant. L'adaptation en temps réel de l'interaction pédagogique entre un tuteur et un apprenant est un des objectifs majeurs des systèmes tutoriels intelligents (Nkambou, Mizoguchi et al. 2010).

Plusieurs méthodes existantes permettent la conception des systèmes tutoriels intelligents. Parmi ces méthodes certaines proposent de représenter explicitement les processus cognitifs utilisés par les utilisateurs humains lors de la résolution de problèmes. D'un point de vue informatique, ces méthodes ont été utilisées pour représenter le comportement intelligent des agents autonomes. Parmi les architectures informatiques proposées dans le cadre de ces méthodes, Act-r est une des architectures les plus connues. Cependant, dans Act-r, le traitement des informations en mémoire est une « boîte noire ». Elle ne permet pas de représenter le flux de connaissances au sein du modèle de l'apprenant. Pour cette raison, nous proposons notre propre architecture permettant d'expliciter les processus cognitifs de l'apprenant mis en jeu lors de l'apprentissage de procédure sur des systèmes techniques. Ces processus portent essentiellement sur la mémorisation. Nous nous appuyons donc sur la théorie de la mémoire d'Atkinson et Shiffrin (Atkinson et Shiffrin 1968), que nous instancions dans le contexte de l'apprentissage des procédures, tout en nous inspirant des concepts d'Act-r.

L'originalité de notre démarche est de permettre au tuteur d'inférer et de prendre en compte les contenus de la mémoire de l'apprenant, afin qu'il puisse adapter l'exécution du scénario pédagogique et individualiser ses interventions en choisissant des actions pédagogiques appropriées telles que la modification de l'environnement, poser des questions à l'apprenant et répondre aux siennes. De plus, afin de favoriser l'engagement de l'apprenant, nous considérons qu'il est important de maintenir une communication naturelle entre le tuteur et l'apprenant. Pour y parvenir, nous proposons d'incarner le tuteur dans l'environnement virtuel.

1.4 Contribution : MEMORIA

Notre contribution majeure porte sur la modélisation et l'implémentation d'une architecture permettant de représenter l'état de la mémoire de l'apprenant lors de ses interactions avec un tuteur adaptatif. Pour cela, nous proposons le modèle Memoria, un **M**odèle de r**E**présentation de la **M**ém**O**ire de l'app**R**enant pour les systèmes tutoriels **I**ntelligents et **A**daptatifs. L'apport majeur de ce modèle réside dans une formalisation et une implémentation du modèle de l'apprenant sous forme de mémoires qui stockent les informations perçues par l'apprenant dans un environnement virtuel. Ces informations sont constituées des instructions et des objets à manipuler dans le cadre de procédures sur des systèmes techniques. Elles sont organisées dans trois mémoires : la mémoire sensorielle, la mé-

moire de travail et la mémoire à long terme. L'enjeu majeur de cette thèse porte sur la formalisation de l'encodage des informations dans ces mémoires, ainsi que le flux d'information entre celles-ci. La Figure 1.3 montre une représentation simplifiée de notre modèle MEMORIA.

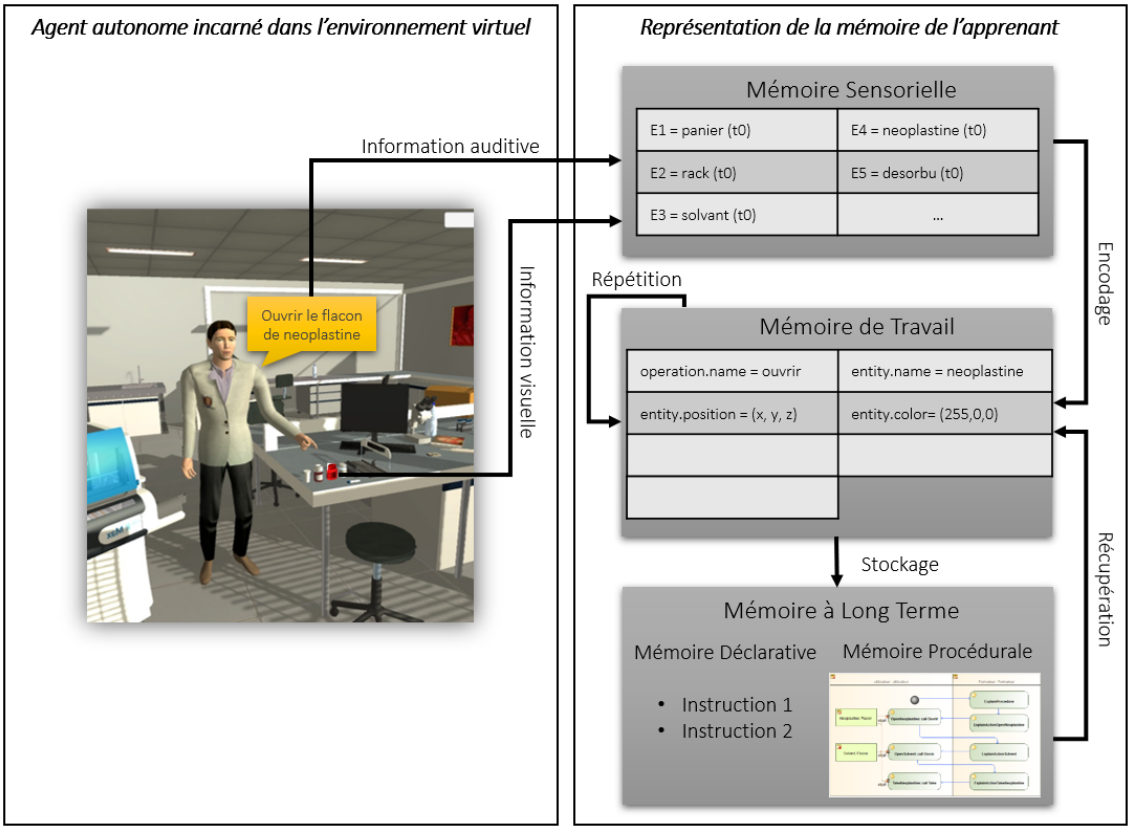


FIGURE 1.3 – Représentation du modèle MEMORIA.

Dans une perspective de valider notre modèle MEMORIA nous avons mené une expérience destinée à examiner l'impact de la présence d'un tuteur virtuel adaptatif sur les performances d'un apprentissage de procédure. Il s'agissait de mesurer les performances d'apprentissage des apprenants lorsqu'ils interagissent avec un tuteur adaptatif, en termes de temps de réalisation de la tâche, du nombre de demandes d'aides et du nombre d'actions incorrectes.

1.5 Organisation du mémoire

Ce mémoire est organisé autour de trois chapitres principaux. Le premier chapitre est dédié à l'état de l'art (Chapitre 2). Dans ce chapitre nous étudions tout d'abord, l'utilisation des environnements virtuels d'apprentissage et plus précisément dans le cas de l'apprentissage de procédure. Nous y présentons également, dans la Section 2.1, les architectures logicielles qui permettent d'expliciter les connaissances dans ces environnements virtuels. Ces connaissances portent non seulement sur les modèles métiers mais égale-

ment sur les modèles pédagogiques. L'utilisation de ces architectures permet d'une part aux experts métiers et aux formateurs de concevoir les situations pédagogiques et fournit d'autre part une base de connaissances aux agents pédagogiques. Les agents pédagogiques qui raisonnent sur ces connaissances afin d'adapter l'apprentissage à l'apprenant sont appelés des systèmes tutoriels intelligents. Nous présentons dans la Section 2.2 les différentes méthodes de conception de ces systèmes. Dans ce chapitre nous montrons également que l'incarnation du tuteur dans la situation pédagogique favorise les interactions avec l'apprenant et améliore ses performances d'apprentissage (Section 2.2.3).

À l'issue de cet état de l'art, nous proposons dans un deuxième chapitre un agent pédagogique adaptatif et incarné dans l'environnement virtuel (Chapitre 3). Pour cela, nous proposons un comportement de tuteur adaptatif, capable de s'adapter en fonction de l'évolution des apprenants, c'est-à-dire d'adapter l'exécution du scénario pédagogique et de personnaliser ses interventions en choisissant des actions pédagogiques telles que modifier l'environnement, poser des questions, répondre à des questions, etc. Le choix des actions pédagogiques réalisées par le tuteur est basé, essentiellement, sur une représentation du modèle de l'apprenant. Dans notre travail, la représentation du modèle de l'apprenant est formalisée autour des travaux en psychologie cognitive définissant la manière dont les informations données par un tuteur et l'environnement virtuel sont encodées et stockées dans la mémoire de l'apprenant. Le modèle informatique de l'encodage de ces informations constitue l'apport majeur de notre travail. Afin de permettre à l'apprenant d'interagir avec le système d'une manière naturelle, nous choisissons de représenter notre interface par un agent conversationnel animé.

Ensuite, dans un troisième chapitre, afin d'étudier l'impact de notre modèle, nous menons deux expériences (Chapitre 4). L'objectif de la première expérience est de s'assurer que la procédure utilisée peut réellement être apprise en environnent virtuel et d'identifier la configuration de réalité virtuelle (écran, casque ou CAVE) la plus adaptée à cet apprentissage. Dans la seconde expérience, nous comparons les performances d'apprentissage des apprenants entre deux conditions. Dans la première condition, nous intégrons un tuteur adaptatif utilisant notre modèle, qui adapte l'exécution du scénario pédagogique et dans la seconde condition, un tuteur non adaptatif qui réalise un scénario pédagogique figé.

Enfin, notre travail est conclu dans le Chapitre 5. Nous faisons un bilan sur ce que notre modèle est capable de réaliser et présentons des limites et des perspectives.

2

État de l'art

L'objectif de notre travail de recherche est de concevoir un environnement virtuel pour l'apprentissage de compétences techniques sur des systèmes industriels. Généralement, les situations d'apprentissage conçues dans ce domaine s'appuient sur la réalisation de scénarios pédagogiques qui définissent les assistances pédagogiques que le système propose à l'apprenant afin de favoriser son apprentissage.

L'intérêt des environnements virtuels pour l'apprentissage humain est que ces environnements permettent l'apprentissage de différents types de compétences (gestes techniques, savoirs et compétences techniques et non-techniques) dans des situations contrôlées, moins risquées et souvent moins coûteuses par rapport à des situations réelles. Cependant, la conception de ces environnements virtuels reste un enjeu majeur, parce qu'elle nécessite d'exprimer l'ensemble des connaissances à acquérir et les moyens de favoriser l'apprentissage. Dans la Section 2.1, nous présentons différents types de travaux en réalité virtuelle destinés à la formation et nous insistons plus précisément sur les méthodologies et les outils qui permettent aux experts métiers et aux formateurs de concevoir eux-mêmes les situations pédagogiques.

Grâce à certaines des méthodes présentées, les formateurs peuvent rédiger eux-mêmes les scénarios pédagogiques. Cependant, dans le cadre de l'apprentissage de procédure sur des systèmes techniques, l'acquisition de la procédure nécessite plusieurs répétitions. Lors de ces répétitions, les apprenants adoptent des stratégies d'apprentissage différentes. L'environnement virtuel et les scénarios pédagogiques doivent donc s'adapter aux apprenants. Cette problématique est l'objectif majeur des systèmes tutoriels intelligents. Nous définissons ces systèmes et nous présentons différents types de modèles qui permettent cette adaptation en Section 2.2.

Dans la Section 2.2.3, nous montrons l'intérêt de l'incarnation du tuteur afin d'améliorer les performances d'apprentissage, par exemple en terme d'engagement de l'apprenant et de prise en compte de ses caractéristiques affectives. Nous présentons également dans cette section, les bibliothèques génériques d'agents conversationnels animés qui

permettent d'incarner ces tuteurs dans un environnement virtuel.

Sommaire

2.1 Environnement Virtuel pour l'Apprentissage Humain

La réalité virtuelle offre de nombreuses opportunités pour le milieu éducatif. Cette technologie s'est positionnée comme l'une des techniques les plus prometteuses pour changer et améliorer l'éducation. Plusieurs applications éducatives de réalité virtuelle ont déjà été développées.

Par exemple, l'application « InMind VR [1] » permet aux utilisateurs d'expérimenter un voyage dans le cerveau d'un patient à la recherche des neurones qui causent les troubles mentaux. « MEL Chemistry VR [2] » est une application destinée aux cours de chimie en réalité virtuelle, adaptée au programme scolaire. En utilisant des jeux scientifiques et des outils immersifs, cette application vise à transformer l'étude de chimie en un processus divertissant pour apprendre les rudiments de la chimie. Enfin, l'application « Google Expeditions [3] », spécialement destinée au domaine de l'éducation, est un outil d'enseignement en réalité virtuelle qui permet de mener et de participer à des voyages virtuels immersifs dans le monde entier. Cette application permet de visiter la station spatiale internationale, d'escalader l'Everest, de vivre la Seconde Guerre Mondiale et de visiter le centre européenne pour la recherche nucléaire, etc. (Figure 2.1).

FIGURE 2.1 – Exemples d'expériences de réalité virtuelle dans Google Expeditions.

La réalité virtuelle offre donc la possibilité de simuler de nombreuses expériences au sein d'un environnement virtuel et de les scénariser. Elle permet, par exemple, de recréer des situations et des scénarios rarement répétables ou impossibles à simuler en conditions réelles. De plus, une formation en réalité virtuelle peut être utilisée et répétée autant de fois que souhaité. Elle permet donc un retour sur investissement plus rapide pour les organismes de formation.

2.1.1 Domaines d'application des environnements virtuels pour l'apprentissage

Les travaux que nous présentons dans cette section se focalisent particulièrement sur les environnements virtuels pour l'apprentissage humain (EVAH). Dans le domaine de la formation, les environnements virtuels pour l'apprentissage humain permettent l'acquisition de différents types de connaissances et de compétences :

1. https://luden.io/inmind/
2. https://melscience.com/vr/
3. https://edu.google.com/expeditions/

- apprentissage de savoirs,
- apprentissage de compétences non-techniques,
- apprentissage de gestes techniques,
- apprentissage de procédure.

La réalité virtuelle et la réalité augmentée ont été appliquées dans le domaine de l'« apprentissage de savoirs ». Par exemple, le projet « EVEIL-3D[4] » (Roy 2014) (Environnement virtuel pour l'enseignement immersif 3D) s'appuie sur des technologies de réalité virtuelle immersives pour l'apprentissage des langues étrangères. Grâce à la combinaison de la reconnaissance des gestes et de la parole, les utilisateurs peuvent parler, agir et interagir comme dans le monde réel. Ce projet considère que l'immersion visuelle et l'interaction gestuelle renforcent le sentiment de présence chez l'apprenant et que cela influe positivement sur la compréhension de l'oral en langue étrangère (Figure 2.2a).

(a) Eveil-3d (b) VirtualiTeach (c) Holo-Math

Figure 2.2 – Exemple d'applications pour l'apprentissage de savoirs.

Le projet « VirtualiTeach » (Loup-Escande et al. 2013) avait pour but de concevoir de nouveaux outils pédagogiques pour l'enseignement et l'apprentissage de concepts scientifiques abordés dans plusieurs formations techniques et professionnelles. L'approche originale de ce projet est d'avoir adopté une démarche de conception centrée sur la technologie et l'utilisateur. Cette démarche s'articule autours de quatre étapes : spécifier le contexte d'utilisation, spécifier les besoins, produire des preuves de concept et évaluer les preuves de concept.

« Holo-Math[5] » est un projet international dont l'objectif est de plonger les participants dans des expériences mathématiques immersives grâce à la technologie de la réalité augmentée. Ce projet initié par Cédric Villani, étudie comment la réalité augmentée peut être utilisée pour expliquer et explorer des concepts mathématiques. Les utilisateurs sont équipés d'un casque HoloLens de Microsoft et initiés aux mondes « mathématiquement augmentés » (Figure 2.2c).

Des systèmes de réalité virtuelle pour la formation aux « compétences non-techniques » telles que la prise de décision ont également été développés. Le système déci-

4. http://www.eveil-3d.eu/francais/
5. http://holo-math.org/

sionnel « GULLIVER » (FRICOTEAUX et al. 2014) a notamment été utilisé pour la formation à la prise de décision dans le domaine fluvial. Le principe de ce système est d'évaluer le niveau de compétence de l'apprenant, et en fonction de ce niveau, d'augmenter l'environnement virtuel par des métaphores qui facilitent la prise de décision de l'apprenant (Figure 2.3).

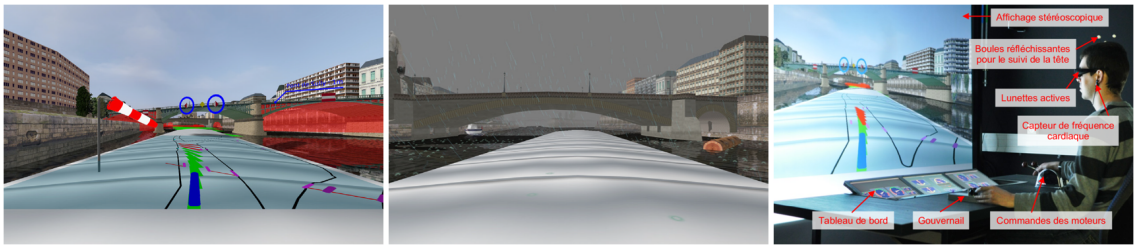


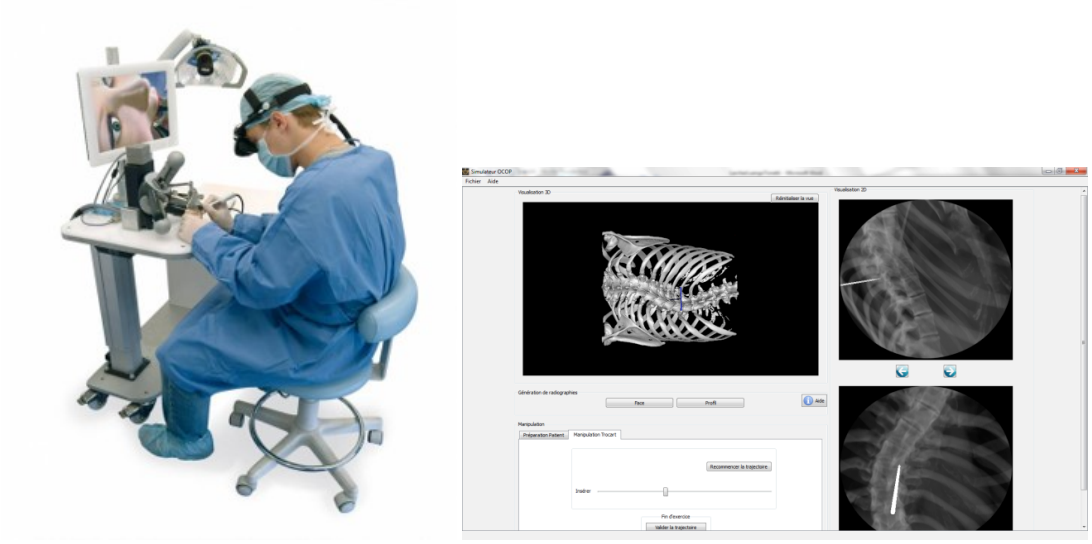
FIGURE 2.3 – Exemple de métaphores dans l'environnement virtuel en fonction du niveau de l'apprenant dans GULLIVER.

Le projet « MacCoy Critical » permet la formation aux situations critiques. Plus spécifiquement, ce projet adresse les situations de dilemme et d'ambiguïté dans le domaine de la sécurité routière et de la médecine (BENABBOU et al. 2017). L'idée de ce projet est de générer des situations dynamiques dans lesquelles l'apprenant doit faire un choix entre deux actions ayant une conséquence négative. Par exemple, un conducteur doit choisir entre freiner ou non en arrivant à un feu rouge en situation d'aquaplaning (Figure 2.4). L'idée de ce projet est de confronter l'apprenant à ce type de situations afin qu'il en prenne conscience et qu'il apprenne à les éviter.

FIGURE 2.4 – Exemple de situation dilemmatique en environnement virtuel dans le cadre du projet MacCoy Critical.

En ce qui concerne l'« apprentissage de gestes techniques », plusieurs projets ont été également proposés. L'objectif du projet « VirTeaSy » (CORMIER et al. 2011) était de former à l'implantologie dentaire en utilisant la réalité virtuelle. Le projet VirTeaSy est constitué d'un logiciel pour aider à la planification des opérations, d'un environnement virtuel 3D, et d'un bras à retour d'effort pour améliorer les compétences manuelles. L'offre de formation de VirTeaSy se déroule en deux phases : la prise de décision et la réalisation des interventions chirurgicales (Figure 2.5a).

Le projet « TELEOS » (LUENGO et al. 2011) (Technology Enhanced Learning Environment for Orthopaedical) était destiné à l'apprentissage de l'orthopédie percutanée. Il vi-



(a) (b)

FIGURE 2.5 – Apprentissage de gestes techniques en implantologie (a) et en orthopédie percutanée (b).

sait principalement à l'apprentissage de connaissances perceptivo-gestuelles. L'objectif du système était de permettre à l'apprenant de s'entraîner en autonomie. La problématique de ce domaine est de permettre au chirurgien de positionner correctement son outil dans le corps du patient. Pour cela, deux indicateurs sont disponibles : les radiographies et les pressions ressenties lors de l'introduction de l'outil dans le corps. Le projet TELEOS simule des radiographies que l'apprenant doit consulter lorsqu'il introduit son outil dans un corps virtuel à l'aide d'une interface haptique (Figure 2.5b).

L'utilisation de la réalité virtuelle pour l'« apprentissage de procédure » est un domaine très développé. Ce domaine constitue notre cadre de recherche. Nous détaillons par la suite quelques modèles de référence dans ce contexte.

2.1.2 Conception des environnements virtuels pour l'apprentissage

Même si le potentiel éducatif de la réalité virtuelle ne cesse de croître aux vues des nouvelles applications immersives publiées ces dernières années, la conception des environnements virtuels pour l'apprentissage peut s'avérer difficile. Une des difficultés de la conception des environnements virtuels pour l'apprentissage humain est de fournir aux formateurs un langage qui leur permet de concevoir eux-mêmes la situation pédagogique. Le flot de conception classiquement utilisé lors de la réalisation des environnements virtuels pour la formation est centré sur le *designer* ou l'informaticien. Ce dernier doit dialoguer avec les experts métiers et les formateurs métiers. L'objectif de ce dialogue est d'extraire les connaissances de ces experts afin de les implémenter dans l'environnement virtuel destiné à la formation (Figure 2.6). Ainsi l'informaticien doit implémenter :

- les géométries et les animations qui constituent l'environnement virtuel,
- les comportements des objets et des utilisateurs tels que les actions et les procédures issues du modèle métier,
- les scénarios pédagogiques.

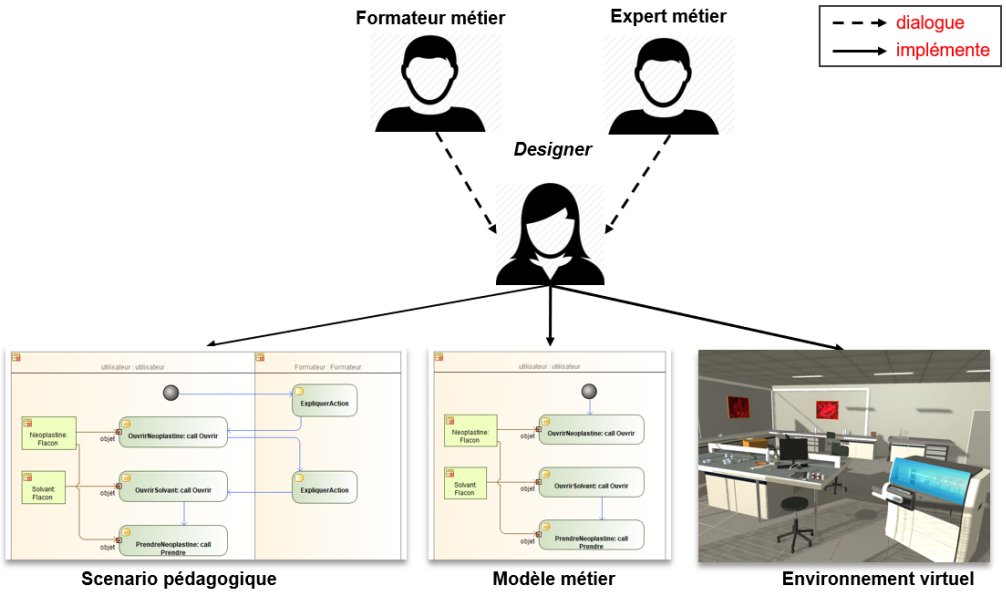


FIGURE 2.6 – Flot de conception classique d'un environnement virtuel pour la formation.

L'inconvénient de cette démarche est l'intervention incontournable et obligatoire des informaticiens lors de toutes les phases de réalisation et de modification des situations pédagogiques. Cette démarche impose donc à l'informaticien de comprendre le métier considéré (qui est un métier complexe dans le cas des systèmes techniques industriels) ce qui peut engendrer des mauvaises interprétations et approximations. De plus, en concevant les scénarios pédagogiques, l'informaticien induit de manière implicite sa propre représentation de la formation.

Afin de contourner ce défaut, l'externalisation de ces expertises est une solution proposée. C'est-à-dire, de transformer les connaissances métiers en données externes au programme informatique. Elles deviennent alors explicites lors de l'exécution de la simulation. Les modèles que nous présentons par la suite proposent de modifier le flot de conception classique afin que l'expert métier puisse par lui-même définir les concepts et les comportements dans l'environnement virtuel et que le formateur métier puisse lui-même également définir les scénarios pédagogiques (Figure 2.7).

Les architectures logicielles que nous présentons par la suite sont parmi les modèles de référence dans le cadre de l'apprentissage de procédure. Notre travail de recherche s'intéresse davantage à ce domaine d'apprentissage.

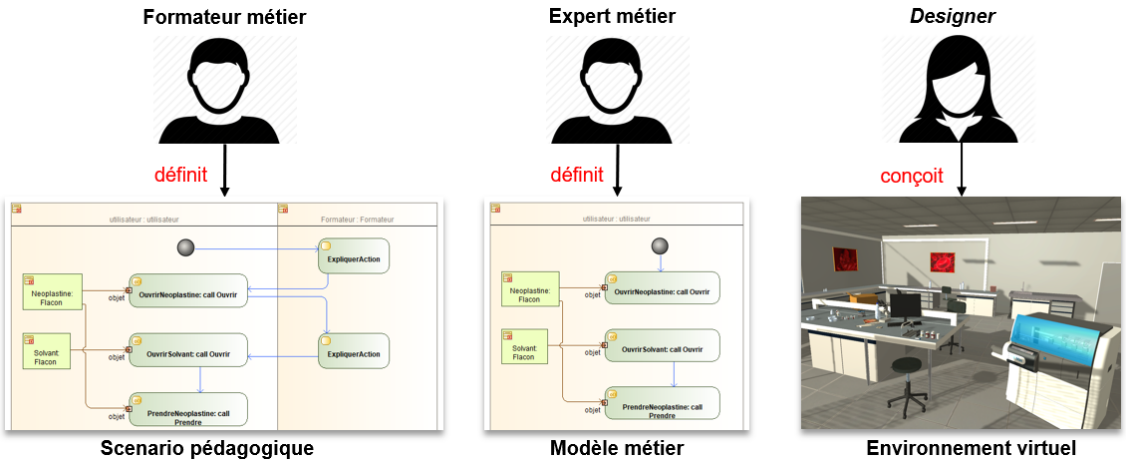


FIGURE 2.7 – Principe de modification du flot de conception classique.

2.1.2.1 #FIVE et #SEVEN

La suite logicielle #FIVE (Framework for Interactive Virtual Environments) (BOUVILLE BERTHELOT et al. 2015) est destinée au développement d'environnements virtuels interactifs et collaboratifs. #FIVE est une API (Application Programming Interface) pour les développeurs d'environnements virtuels. Cette API aide les développeurs lors de la conception des comportements des objets et des interactions entre les utilisateurs et les objets virtuels. L'objectif de #FIVE est de définir les comportements et les interactions indépendamment de la plate-forme de réalité virtuelle sur laquelle l'environnement virtuel s'exécute. L'API #FIVE est composée de trois parties :

- un moteur de relations,
- un moteur d'interactions collaboratives,
- des recommandations d'usage.

Le moteur de relations a pour objectif de permettre aux développeurs de sélectionner les objets interactifs de l'environnements virtuels et de définir leurs comportements. L'idée majeure de #FIVE est qu'un comportement est réalisé dans le cadre d'une relation entre plusieurs objets. Ainsi #FIVE définit la notion d'objet relationnel qui peut être impliqué dans une relation. Un objet relationnel possède des propriétés appelées « Type #FIVE ». Une relation fait intervenir des *patterns* d'objets. Un *pattern* d'objet est défini par un ensemble de « Type #FIVE ». Tout objet possédant au minimum les « Types #FIVE » définis par le *pattern* peut participer à la relation. Ce principe est illustré sur la Figure 2.8

Grâce à ce principe, le moteur de relations est capable de répondre à des questions du développeur telles que lister l'ensemble des objets qui peuvent participer à une relation, vérifier q'un objet peut participer à une relation, etc. Une « réalisation » est une instanciation d'une relation, c'est-à-dire l'affectation d'objet à tous les *patterns* définis dans la relation. L'exécution effective d'une réalisation est définie dans un code informatique spé-

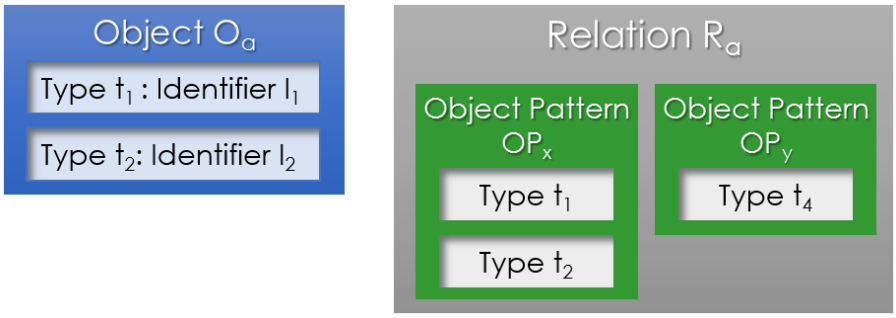


FIGURE 2.8 – Modèle du moteur de relations de #FIVE (BOUVILLE BERTHELOT et al. 2015).

cifique écrit par le développeur. Un objectif de #FIVE est de fournir une bibliothèque des codes informatiques permettant la réalisation des actions les plus courantes telles que prendre, tourner, pousser, etc.

Le deuxième moteur de #FIVE est le moteur d'interactions collaboratives. Le but de ce moteur est d'aider le développeur à définir les stratégies permettant l'interaction collaborative entre les utilisateurs et les objets de l'environnement virtuel. Ce moteur s'appuie sur deux concepts : les objets interactifs et les interacteurs. Un objet interactif est défini par un ensemble de paramètres contrôlables. Un paramètre contrôlable est définit, tout d'abord, par un contrôleur d'accès qui défini les règles de modification et d'observation du paramètre. Il est également défini par un type. Enfin, pour permettre l'interaction collaborative, un paramètre contrôlable est également défini par un *merger* qui définit la politique utilisée lorsque plusieurs utilisateurs souhaitent interagir de manière concurrente sur un même paramètre. Un interacteur est une entité de l'environnement virtuel qui permet de modifier un paramètre contrôlable. Un interacteur est défini par le type de paramètre qu'il peut modifier.

#SEVEN (Sensor Effector Based Scenarios Model for Driving Collaborative Virtual Environment) (CLAUDE, GOURANTON, BOUVILLE BERTHELOT et al. 2014), s'appuie sur #FIVE pour définir les scénarios pédagogiques que les utilisateurs doivent réaliser dans l'environnement virtuel. L'objectif de #SEVEN est de fournir un langage pour les formateurs afin qu'ils écrivent eux-mêmes le scénario pédagogique. Le langage proposé s'appuie sur le formalisme des réseaux de Petri. L'apport de #SEVEN à ce formalisme porte sur la notion d'événement qui est une extension des transitions des réseaux de Petri. Un événement possède des entrées nommées senseurs et des sorties nommées effecteurs. Ce principe est illustré par la Figure 2.9.

Les senseurs observent l'environnement virtuel et se déclenchent lorsqu'une condition est atteinte. Dans ce cas l'événement attaché à ce senseur est généré. L'événement est lui-même attaché à une transition du réseau de Petri. Lors de l'occurrence de l'événement, le système réalise la transition si toutes les places en amont sont valides. Lorsque la transition est réalisée, l'effecteur lié à la transition est exécuté. Un effecteur modifie

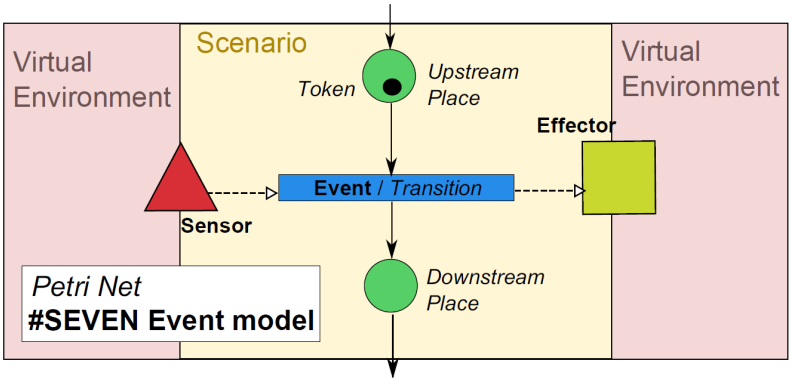


FIGURE 2.9 – Principe de définition d'un scénario pédagogique à l'aide de #SEVEN (CLAUDE, GOURANTON, BOUVILLE BERTHELOT et al. 2014).

des éléments de l'environnement virtuel (changement de valeurs d'une propriété, exécution d'un processus). La perception de l'environnement virtuel et les actions sur celui-ci s'appuient sur le moteur #FIVE. Les données perçues par les senseurs proviennent du modèle d'objet et de relation de l'environnement. Les effecteurs contrôlent la réalisation des relations (Figure 2.10).

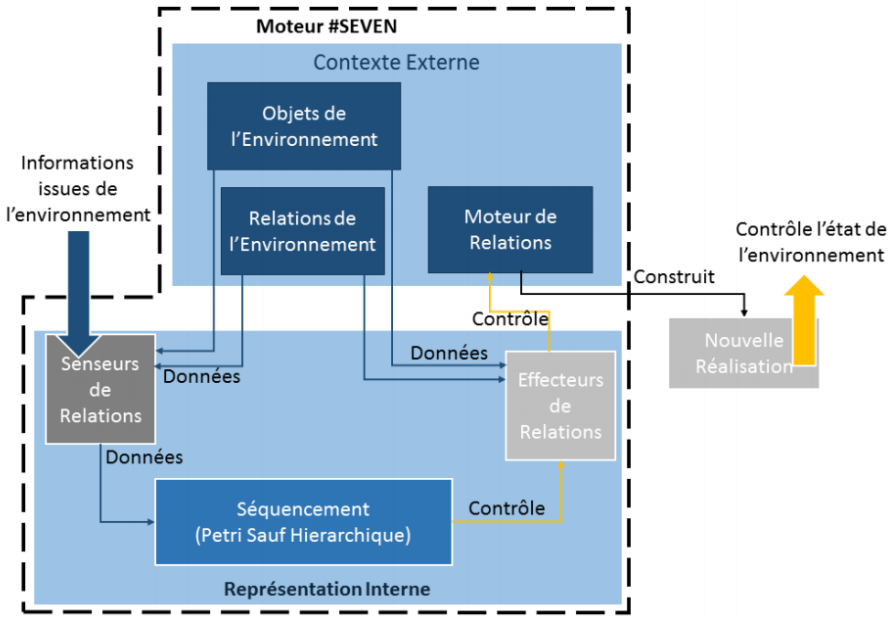


FIGURE 2.10 – Lien entre le moteur de scénario #SEVEN et le moteur d'environnement #FIVE.

#SEVEN donne également la possibilité aux formateurs de définir des rôles. Dans #SEVEN, les rôles sont définis comme un ensemble d'actions que doivent pouvoir réaliser les acteurs jouant le rôle. Les jetons du réseau de Petri sont colorés par un identifiant qui représente l'acteur. Lors d'une transition entre deux places, un rôle est assigné à l'acteur dont la référence est portée par le jeton impliqué dans cette transition. Ceci permet l'affectation dynamique de rôle aux acteurs.

#Five et #Seven ont été utilisés pour la conception d'environnement virtuel destiné à la formation. La Figure 2.11 montre un exemple d'utilisation de #Five et #Seven pour la formation du personnel hospitalier dans une salle d'opération dans le cadre du projet S3PM.

Figure 2.11 – Cas d'utilisation de #Five et #Seven dans le projet S3PM (Claude 2016).

2.1.2.2 HUMANS

Humans (HUman Models based Artificial eNvironments Software platform) (Lourdeaux et al. 2017) est une suite logicielle destinée à la formation en environnement virtuel. Même si les modèles de Humans permettent la description des compétences techniques, l'objectif des environnements virtuels conçus avec cette suite logicielle est la formation aux compétences non techniques. L'hypothèse principale de Humans est qu'il est impossible pour un formateur de rédiger l'ensemble des scénarios pédagogiques qui permettent de couvrir toutes les compétences à apprendre dans l'environnement virtuel, l'idée est donc de générer de manière adaptative le scénario en fonction des compétences non acquises par l'apprenant. Humans est composé de quatre modules, appelés moteurs décisionnels (Figure 2.12) :

- Replicants : pour les personnages virtuels autonomes.
- Seldon : pour la scénarisation, lui-même décomposé en deux sous modules (Tailor et Director).
- World Manager : pour la gestion du monde.
- Monitor : pour les traces de l'apprenant.

Le module « Replicants » gère les personnages virtuels autonomes. Ce module se focalise sur les comportements autonomes des personnages et ne traite pas le rendu graphique de ceux-ci. Ce module est basé sur l'architecture cognitive Soar et permet aux agents de réaliser les activités collaboratives définies dans le modèle. L'intérêt de ce mo-

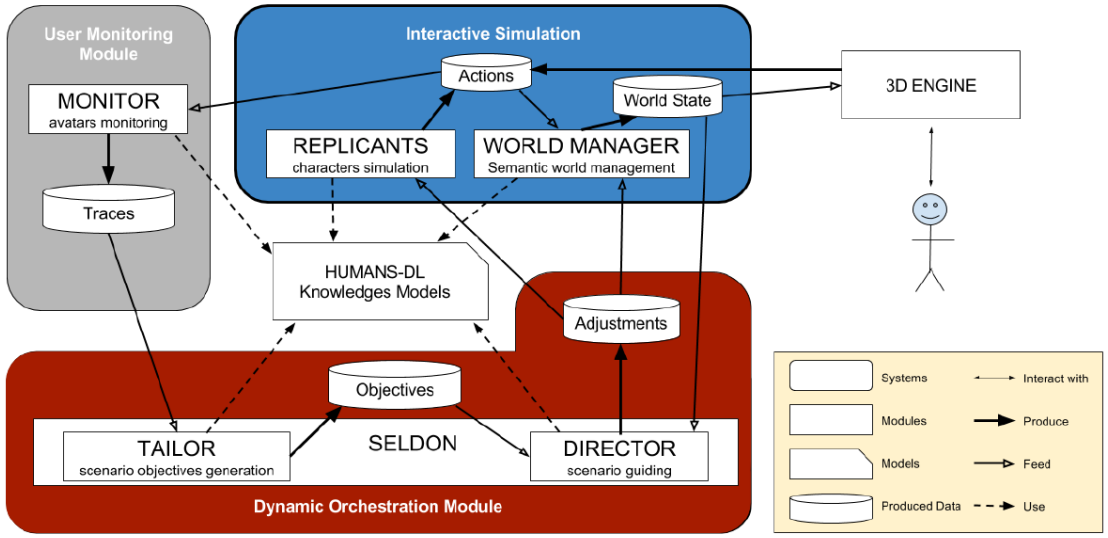


FIGURE 2.12 – Architecture globale de HUMANS (LOURDEAUX et al. 2017).

dule est centré sur la simulation des émotions et de la personnalité des personnages. Ce module peut également gérer les notions de stress, fatigue et niveaux de compétences.

Le module « MONITOR » a pour objectif d'enregistrer les actions de l'apprenant, d'en faire une analyse, et d'en faire un retour en temps réel au formateur. Ces informations peuvent également être utilisées pour le rejeu de la simulation. Ce module fournit ces informations au module « SELDON » afin que ce dernier génère de nouveaux objectifs pédagogiques et donc de nouveaux scénarios.

Le module « WORLD MANAGER » gère l'état du monde en fonction du modèle métier exprimé, des actions des utilisateurs humains et des personnages autonomes. Il est également capable de retourner l'état du monde actuel aux autres modules ou de mettre le monde dans un état spécifique en fonction des requêtes des autres modules.

Dans le cadre spécifique de notre travail, sur l'adaptation en temps réel du scénario pédagogique en fonction de l'apprenant, le module de HUMANS qui nous intéresse le plus est le module « SELDON ». En effet, ce module a pour objectif de générer des situations dans l'environnement virtuel en fonction des compétences à atteindre par l'apprenant. SELDON est composé de deux processus : « TAILOR » et « DIRECTOR ». TAILOR reçoit les traces d'activités de l'apprenant et définit un profil d'apprenant. Ce profil est constitué de l'ensemble des situations auxquelles l'apprenant a été confronté. HUMANS propose une implémentation de la théorie de la Zone Proximale de Développement (VYGOTSKY 1978). Cette théorie représente les connaissances de l'apprenant dans un espace vectoriel de classes de situations à travers des valeurs de croyance. Le module TAILOR utilise ces croyances pour définir un objectif pédagogique qui permettrait une évolution progressive des compétences de l'apprenant. Cet objectif se définit par un état du monde à atteindre afin de mettre l'apprenant dans la situation visée. Dans HUMANS, cet objectif est appelé « objectif scénaristique » et est transmis au module DIRECTOR.

Le fonctionnement du module DIRECTOR est présenté sur la Figure 2.13. En fonction de l'objectif scénaristique et de l'état du monde, DIRECTOR génère un scénario qui permet d'atteindre l'état du monde correspondant à l'objectif. Pour cela DIRECTOR raisonne sur les modèles métiers définis dans HUMANS *via* un moteur de planification. Ce moteur génère des « ajustements » du monde, qui sont transmis au « WORLD MANAGER » afin d'être exécutés dans l'environnement virtuel.

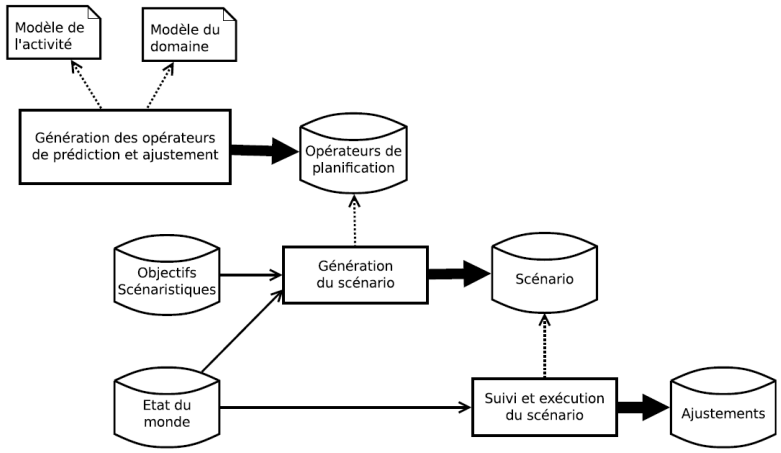


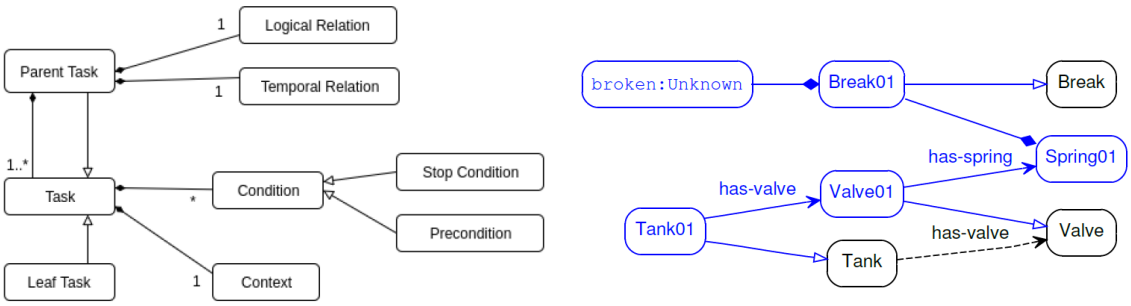
FIGURE 2.13 – Fonctionnement du module DIRECTOR (LOURDEAUX et al. 2017).

Ces quatre modules s'appuient sur des modèles de représentation de connaissance. Dans HUMANS deux types de connaissances sont exprimées : des connaissances sur le métier (activité et environnement) et des connaissances sur les liens de causalité. Pour décrire les activités, HUMANS propose un nouveau langage (ACTIVITY-DL) dont le méta-modèle est présenté sur la Figure 2.14a. Ce langage s'inspire de ceux utilisés dans le domaine de l'ergonomie (par exemple, MAD* (RODRIGUEZ et SCAPIN 1997)). Dans ce langage, une activité peut être hiérarchique. Une activité organise des tâches grâce à des opérateurs logiques ou temporels. Une tâche peut avoir des pré-conditions et ou des conditions d'arrêt. L'environnement est défini à partir d'un langage destiné aux ontologies (OWL [6]). HUMANS propose un méta-modèle appelé WORLD-DL et exprimé sous forme d'ontologie. Un modèle métier est une instanciation de ce modèle et un exemple est montré sur la Figure 2.14b.

Une idée originale proposée par HUMANS est d'ajouter aux modèles métiers une description des causalités. Ce modèle permet de spécifier les relations de causalité entre les événements qui peuvent survenir dans l'environnement et les états du monde résultants. C'est en partie grâce à ce modèle que le module SELDON peut générer des scénarios. Le méta-modèle du langage permettant l'expression de ces causalités (CAUSALITY-DL) est présenté dans la Figure 2.15a. Un exemple de modèle utilisant ce langage est présenté en Figure 2.15b.

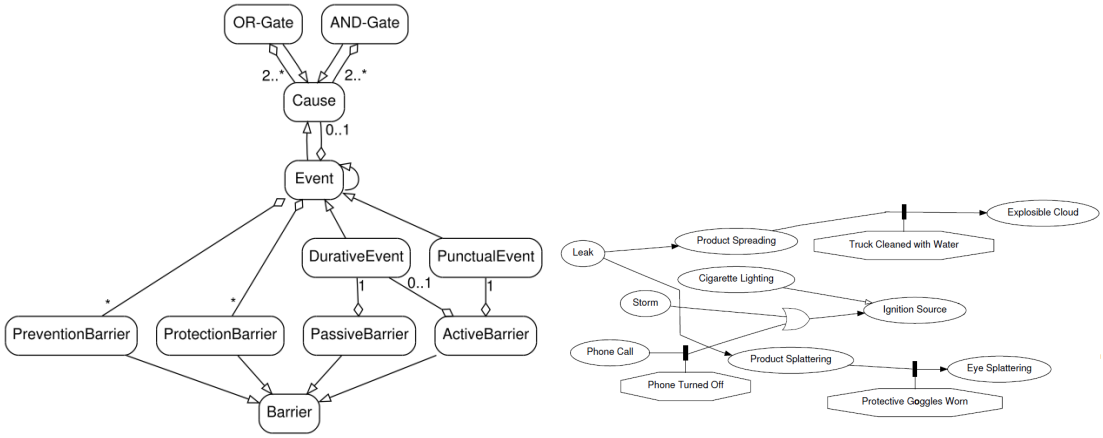
HUMANS a été utilisé dans plusieurs applications à grande échelle, liées à l'appren-

6. https://www.w3.org/TR/owl-features/



(a) (b)

FIGURE 2.14 – Méta-modèle pour la représentation des activités (a) Exemple d'ontologie pour la représentation du monde virtuel (b) (LOURDEAUX et al. 2017).



(a) (b)

FIGURE 2.15 – Modèle de représentation des relations de causalité (a) Exemple d'utilisation (b).

tissage de la gestion des risques, telle que V3S (Figure 2.16), pour l'apprentissage de procédures de sécurité sur les sites SEVESO. Plus récemment HUMANS a été utilisé pour la génération de situations dilémiques et ambigües, telles que la conduite automobile (BENABBOU 2018).

2.1.2.3 MASCARET

MASCARET (Multi Agent System for Collaborative, Adaptive & Realistic Environments for Training) (CHEVAILLIER et al. 2012), est une plate-forme pour la conception d'environnements virtuels pour l'apprentissage humain. L'approche de MASCARET est basée sur le principe de méta-modélisation permettant la définition de la sémantique des environnements virtuels. Cette plate-forme est principalement destinée à la conception d'environnements virtuels orientés systèmes industriels et techniques. Le principe de la démarche est d'externaliser les connaissances spécifiques sur le domaine d'apprentissage ainsi que les scénarios pédagogiques. Ces connaissances sont alors considérées comme des données qui sont produites par les différents intervenants dans la conception de la situation

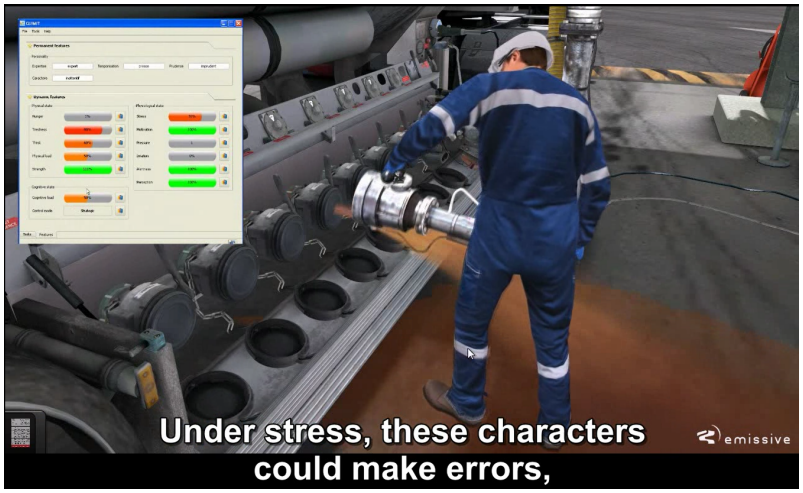


FIGURE 2.16 – Exemple d'utilisation de HUMANS dans le projet V3S (LOURDEAUX et al. 2017).

pédagogique.

Pour cela, MASCARET utilise un langage unifié qui permet de représenter deux aspects de la situation d'apprentissage : son contenu (le geste et la procédure métier) et sa forme (le scénario pédagogique). Cela signifie que dans MASCARET, un scénario pédagogique est considéré comme une procédure métier particulière, résultante de l'expertise d'un métier, celui du pédagogue. De ce fait, les deux modèles peuvent alors partager le même langage de description et être donc considérés comme deux instances d'un même méta-modèle. Le langage choisi pour produire ces données est UML (Unified Modeling Language) car il est suffisamment formalisé pour pouvoir être interprété de manière automatique par l'ordinateur (MASCARET est un interpréteur UML dans un contexte de réalité virtuelle) et il est également graphique ce qui le rend accessible aux non informaticiens. La Figure 2.17 montre le flot de conception d'une situation pédagogique en environnement virtuel à l'aide de MASCARET.

Ce flot de conception fait intervenir quatre rôles (SAUNIER et al. 2016 ; TAOUM et al. 2015) :

- expert métier,
- *designer* (informaticien),
- pédagogue,
- formateur métier.

L'« expert métier », qui connaît le domaine à apprendre, définit la structure du système, le comportement (pro-actif ou réactif) des objets et les procédures d'utilisation et de maintenance du système industriel. Cette description est indépendante de la plate-forme d'exécution. Pour définir le modèle métier, l'expert métier utilise un modeleur UML indépendant de MASCARET. La structure du système est exprimée à l'aide des diagrammes

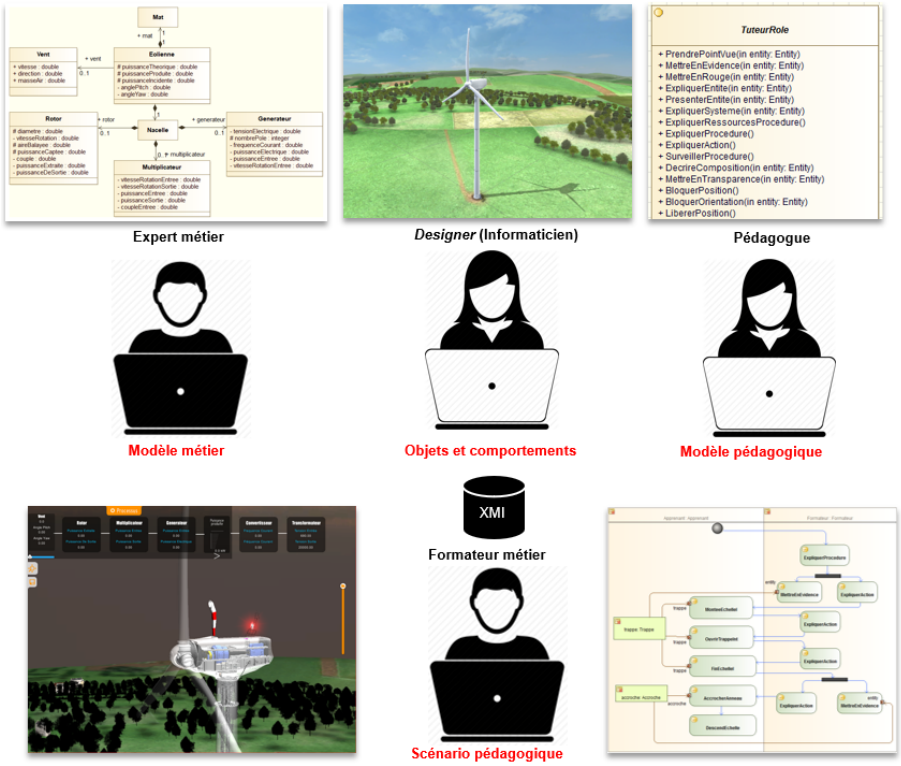
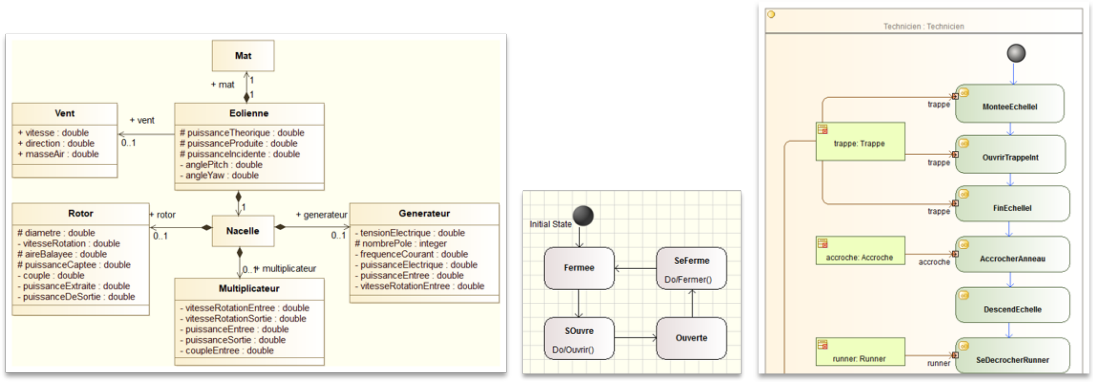


FIGURE 2.17 – Flot de conception d'une situation pédagogique en environnement virtuel à l'aide de MASCARET.

de classes (Figure 2.18a). Ainsi, l'expert définit les concepts et les composants impliqués dans le système par des classes. Ces concepts peuvent être plus ou moins abstraits grâce à la notion d'héritage. Les concepts et les composants sont structurés entre eux par les associations et les compositions UML et peuvent avoir des propriétés sous forme d'attributs UML. Dans MASCARET, les comportements de ces composants sont définis par les machines à états UML (Figure 2.18b). Ceci permet de définir des comportements réactifs asynchrones. Le comportement d'un composant est défini par des états dans lesquels le composant réalise des opérations. Ce comportement est sensible à des événements provenants d'autres composants du système ou résultants des actions de l'utilisateur. Les procédures sur le système sont définies par les diagrammes d'activité UML (Figure 2.18c). Les activités permettent d'agencer les actions des utilisateurs à l'aide d'opérateurs de contrôle (parallèle, séquentiel, conditionnel et itération). Les actions peuvent porter sur les objets et les activités peuvent être collaboratives, c'est-à-dire faire intervenir plusieurs rôles. Le modèle métier ainsi exprimé est exporté dans un fichier au format XMI (XML Metadata Interchange) standardisé par l'OMG (Object Management Group). Ce fichier est utilisé par les autres rôles et lors de l'exécution de la situation pédagogique.

Le « *designer* », ou l'informaticien a comme rôle de concevoir les géométries 3D des objets, les scènes dans lesquelles ils seront impliqués et leurs comportements sous forme de transformation géométrique, graphique ou sonore. Le *designer* a également pour rôle de faire le lien entre les objets 3D et les instances des classes définis dans le modèle



(a) (b) (c)

FIGURE 2.18 – Exemple de modélisation métier en UML dans MASCARET.

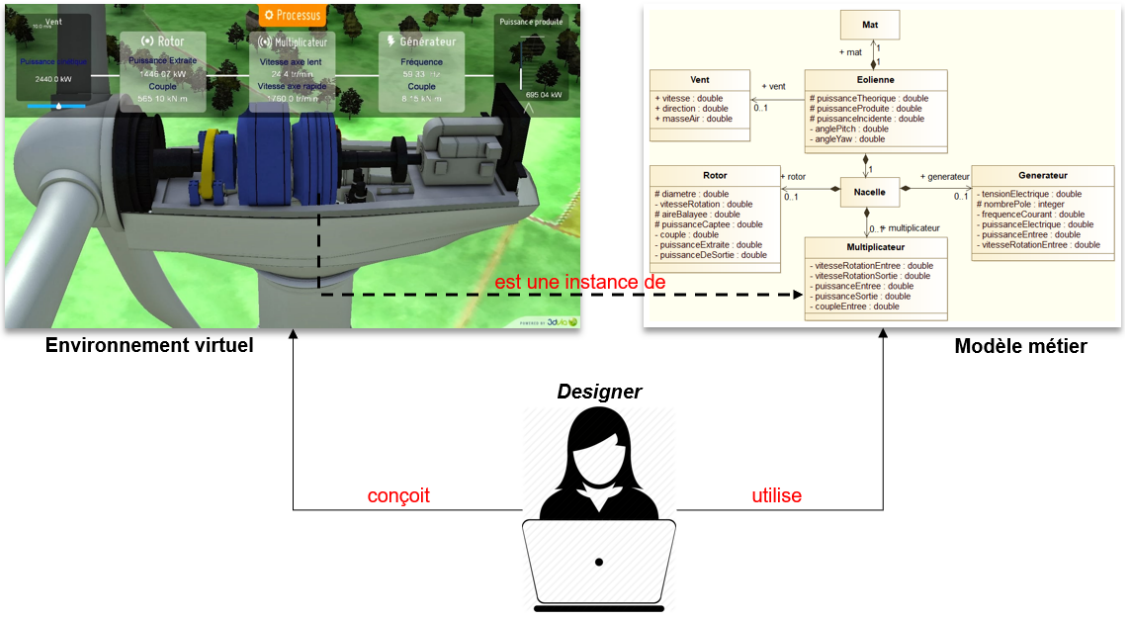


FIGURE 2.19 – Lien entre les objets 3D de la scène et le modèle métier.

UML (*via* le fichier XMI) (Figure 2.19).

Le « pédagogue » définit les actions pédagogiques qui peuvent être utilisées pour guider ou corriger l'utilisateur dans l'environnement virtuel, ainsi que les comportements pédagogiques génériques. Ces actions sont indépendantes du domaine d'application, de l'environnement technologique (dispositif et plate-forme de réalité virtuelle) et des stratégies pédagogiques. Cependant, ces actions dépendent du type d'environnement d'apprentissage. Dans notre cas, il s'agit des simulations interactives en environnement virtuel. « Mettre en évidence un objet » et « changer le point de vue de l'utilisateur » sont des exemples d'actions spécifiques à ces types de simulations. Ces actions sont définies en UML et sauvegardées dans un fichier XMI.

Le « formateur métier » définit des scénarios pédagogiques contenants les assistances pédagogiques fournies par le système en temps réel. Pour définir les scénarios, le

formateur métier utilise (1) l'environnement et les objets qu'il contient, (2) les actions potentielles de l'utilisateur sur les objets (définies par l'expert du domaine), et (3) les actions pédagogiques générales (définies par le pédagogue). Classiquement, un scénario pédagogique est composé d'objectifs pédagogiques, de conditions préalables (pré-requis), d'une organisation pédagogique (rôles), d'activités pédagogiques et d'un environnement virtuel (KOPER et VAN ES 2004). Ainsi, MASCARET propose un modèle formel de scénario pédagogique comme une extension UML. Une organisation pédagogique est considérée comme une collaboration composée de rôles. Un rôle est représenté par le concept d'interface UML. Cela signifie qu'un rôle représente un ensemble de services sans fournir effectivement une implémentation. L'agent (virtuel ou humain) qui joue ce rôle propose sa propre implémentation. Un scénario pédagogique est représenté par une activité qui agence les actions pédagogiques modifiant l'environnement virtuel (composé du système et des ressources pédagogiques).

MASCARET permet de décrire la sémantique de l'environnement virtuel d'apprentissage, que ce soit au niveau métier ou pédagogique. MASCARET propose une sémantique opérationnelle pour tous les concepts exprimés dans le modèle, ce qui permet d'exécuter l'environnement virtuel d'une manière automatique. De plus, tous ces concepts sont explicites lors de la simulation. Ils peuvent donc servir comme base de connaissances pour le raisonnement des agents virtuels.

Dans MASCARET, les environnements virtuels peuvent être peuplés d'agents autonomes. Ces agents réalisent des comportements qui peuvent modifier l'environnement. MASCARET propose des comportements génériques, comme par exemple le comportement procédural. Ce comportement permet aux agents de réaliser les procédures et de suivre leur exécution. Nous rappelons que ces procédures peuvent être des procédures métiers ou des scénarios pédagogiques. Le *designer* peut créer de nouveaux comportements.

Les agents ont également la capacité de communiquer entre eux (NAKHAL 2017). Cette capacité s'appuie sur les actes de langages proposés par l'organisation FIPA[7] (Foundation for Intelligent Physical Agents). MASCARET propose un comportement de communication générique pour gérer automatiquement la communication entre agents. Ce comportement permet, entre autres, d'une part la synchronisation entre les agents lors de l'exécution d'une activité et d'autre part de répondre aux questions de l'utilisateur humain sur le modèle.

Les agents peuvent avoir une représentation géométrique dans l'environnement virtuel. Pour cela, MASCARET intègre différentes plates-formes d'agents conversationnels animés compatibles avec l'architecture SAIBA (Situation Agent Intention Behavior Animation) (VILHJÁLMSSON et al. 2007). Nous détaillons ces plates-formes d'agents conversationnels animés dans la Section 2.2.3.

7. http://www.fipa.org/specs/fipa00008/SC00008I.html

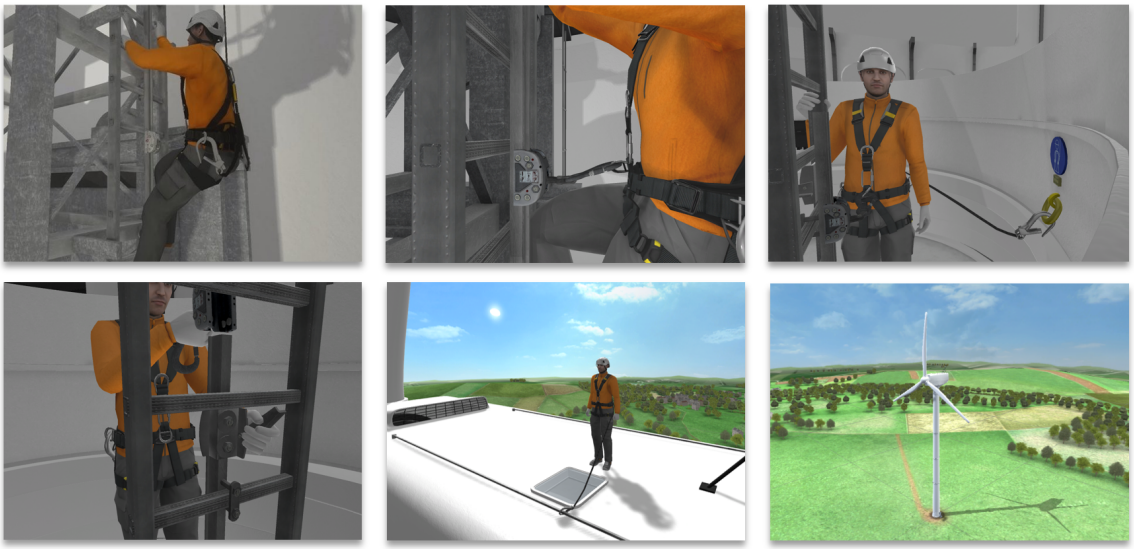


FIGURE 2.20 – Réalisation d'une procédure de sécurité dans le projet EAST à l'aide de MASCARET.

MASCARET a été utilisé dans de nombreux projets pour la formation sur des systèmes industriels tel que le projet EAST[8] (Environnements d'Apprentissage Scientifiques et Techniques) pour la formation aux procédures sur des éoliennes (Figure 2.20).

2.2 Systèmes Tutoriels Intelligents

Les systèmes tutoriels intelligents (ITS : Intelligent Tutoring System) sont des environnements d'apprentissage informatisés qui visent à imiter et simuler le comportement d'un tuteur humain dans ses capacités d'expert pédagogue et d'expert du domaine. Les ITS disposent de capacités de raisonnement. Ils évaluent les connaissances acquises par l'apprenant, en comparant ses activités aux informations du domaine et proposent des assistances adaptées.

Selon Woolf (WOOLF 1992), un ITS est composé de quatre composantes principales (Figure 2.21), jouant chacune un rôle spécifique :

- modèle du domaine : référence la connaissance de l'expert du domaine et le sujet enseigné.
- modèle de l'apprenant : représente les compétences de l'apprenant et ce qu'il a réalisé.
- modèle du tuteur (ou pédagogique) : effectue des choix d'assistance pédagogique en fonction du modèle du domaine et du modèle de l'apprenant.
- interface : représente l'interface de l'environnement interactif, permettant la communication entre l'apprenant et le système.

8. http://projet-east.blogspot.com/

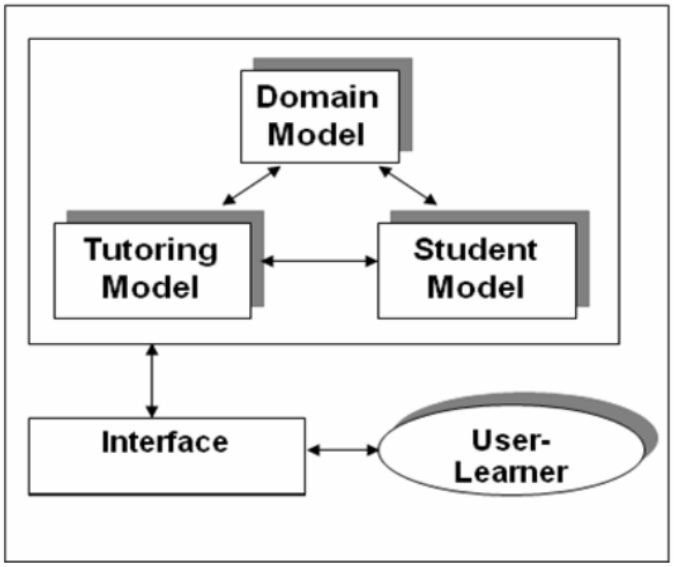


FIGURE 2.21 – Les quatre composantes d'un système tutoriel intelligent (NKAMBOU, MIZOGUCHI et al. 2010).

Cette décomposition en quatre modèles est l'architecture historique, qui constitue les fondements des premiers travaux sur les ITS. Des travaux dans ce domaine, remettent en cause cette structuration (BOURDEAU et GRANDBASTIEN 2010). En effet, il est difficile d'identifier un modèle spécifique du tuteur indépendamment des autres modèles. Comme nous l'avons vu dans la section précédente (Section 2.1), il est important que l'expert métier puisse construire par lui-même les connaissances à transmettre à l'apprenant. Nous proposons donc de présenter dans une première partie les différentes façons de représenter le modèle du domaine dans les ITS. Dans une deuxième partie, nous présentons le modèle de l'apprenant et du tuteur dans la même section, car nous estimons que ces deux modèles sont fortement liés. Enfin, des chercheurs estiment que le succès d'un tuteur intelligent résiderait dans ces capacités à interagir de manière naturelle avec l'apprenant en langage naturel par exemple (ce qui correspondrait au modèle d'interface) (BOURDEAU et GRANDBASTIEN 2010). Nous présentons donc cette capacité dans la Section 2.2.3 où nous présentons les agents pédagogiques incarnés et les architectures logicielles qui permettent de mettre en œuvre cette interface naturelle.

2.2.1 Modèle du domaine

Le « modèle du domaine » référence les connaissances de l'expert du domaine et le sujet à enseigner. Traditionnellement, il existe plusieurs façons pour définir le modèle du domaine dans les ITS. La difficulté lors de la définition de ce modèle est non seulement de fournir une représentation explicite des concepts liés au sujet de l'apprentissage, mais également de fournir les mécanismes qui permettent de raisonner sur ces concepts afin de fournir une solution à un problème posé. Selon Nkambou, trois grands types de modèles ont été utilisés pour représenter le modèle du domaine (NKAMBOU 2010) :

- Les modèles de type « Black Box » proposent une solution à un problème particulier, mais la manière d'atteindre ce résultat n'est pas explicitée (à travers un simulateur par exemple).
- Les modèles de type « Glass Box » dans lesquels chaque étape du raisonnement est explicite (par exemple, par un ensemble de règles de types « `Si Alors` »).
- Les modèles « Cognitifs » proposent de représenter la connaissance de la même manière qu'elle pourrait être représentée dans l'esprit de l'apprenant. Les architectures cognitives telles que ACT-R peuvent être utilisées.

Quelque soit le modèle utilisé, des langages ont été proposés afin de permettre à l'expert d'exprimer ses connaissances. Nkambou classe ces langages en deux catégories : les langages génériques et les langages orientés pédagogie (NKAMBOU 2010). Parmi les langages génériques, les plus connus sont les langages permettant l'écriture de règles de production. Des langages de type réseaux sémantiques, ontologies ou graphes conceptuels ont également été définis. Les méta-modèles WORLD-DL et ACTIVITY-DL de l'architecture HUMANS, présentée en Section 2.1.2.2, appartiennent à ce type de langage. Un exemple de graphe conceptuel est présenté en Figure 2.22. En complément, des langages basés sur la définition des structures, des propriétés, des relations entre les concepts et des activités ont également été proposés. #FIVE et MASCARET, présentées dans les Sections 2.1.2.1 et 2.1.2.3, peuvent être considérés comme appartenant à ce type de langage.

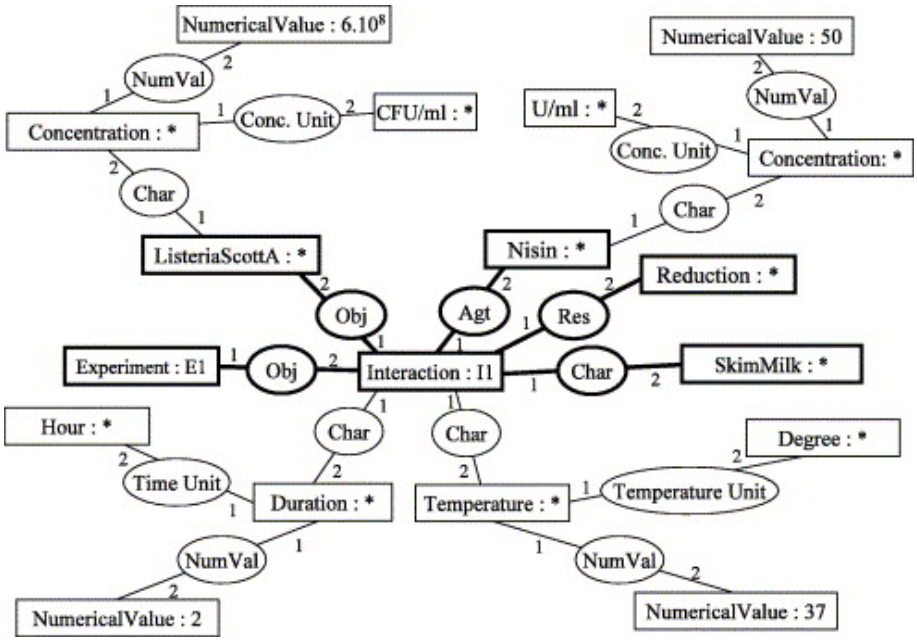


FIGURE 2.22 – Exemple de graphe conceptuel (DIBIE-BARTHÉLEMY et al. 2006).

Parmi les langages orientés pédagogie, deux langages peuvent être identifiés : MOT

(Modeling using Object Types) (PAQUETTE 2007) et CREAM (Curriculum REspresentation and Acquisition Model) (NKAMBOU, FRASSON et al. 2001).

MOT est un langage graphique destiné à la conception des situations pédagogiques et permet aux programmeurs de définir le modèle du domaine à l'aide de cinq types de connaissances : les faits, les concepts, les procédures, les processus et les principes. La Figure 2.23 présente le méta-modèle de ce langage.

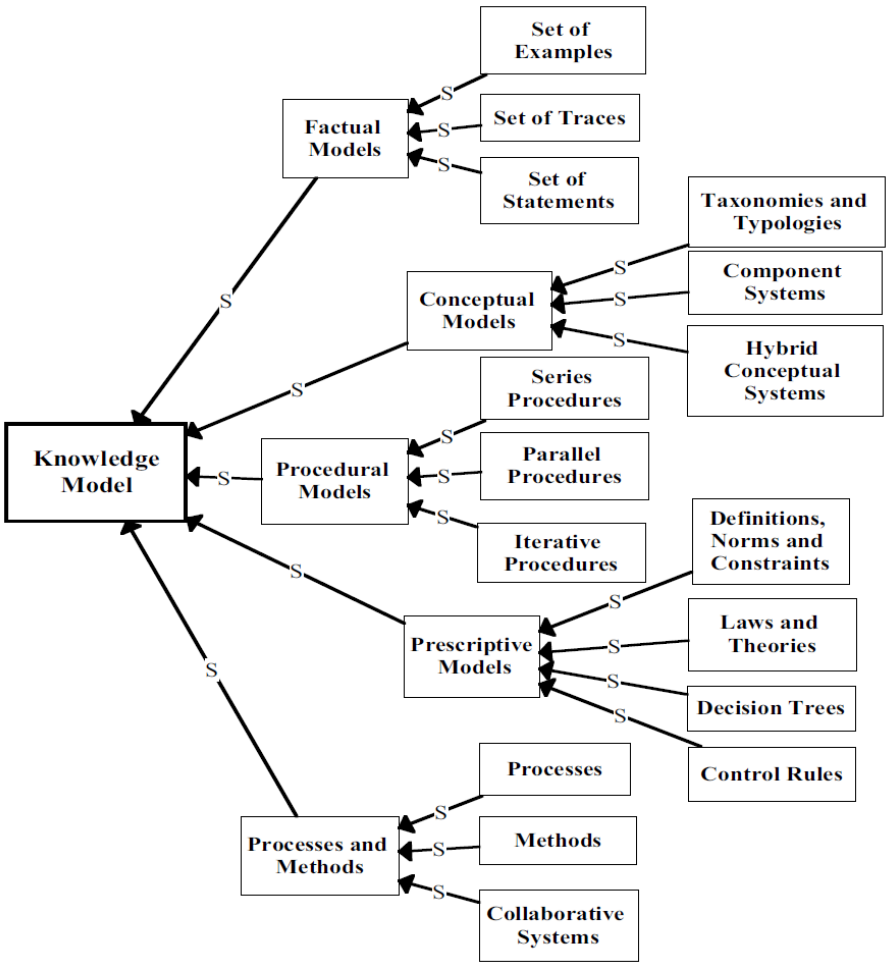


FIGURE 2.23 – Le méta-modèle du langage MOT (PAQUETTE 2007).

2.2.2 Modèle de l'apprenant et du tuteur

L'adaptation en temps réel de la situation pédagogique à l'apprenant est l'un des objectifs majeurs des systèmes de tutoriels intelligents (ITS). Pour cela les ITS s'appuient conjointement sur une représentation de l'apprenant et sur des stratégies pédagogiques (modèle du tuteur).

Classiquement, le « modèle de l'apprenant » référence ce que l'apprenant sait, ce qu'il a fait, ses stratégies d'apprentissage, etc. Les informations représentées par ce modèle peuvent être les compétences de l'apprenant, les connaissances qu'il a sur son propre apprentissage (métacognition) et ses caractéristiques affectives. Ces informations sont

généralement des inférences que se fait le système sur l'apprenant. Ces inférences sont construites par l'observation des interactions que l'apprenant a avec le système et par des mesures de performances de l'apprenant telles que le temps de réalisation d'une tâche et les erreurs observées. Par lui-même, le modèle de l'apprenant ne prend aucune décision. Il ne fait que fournir des informations au modèle du tuteur pour que celui-ci puisse adapter ses interventions à l'apprenant.

En se basant sur les connaissances du modèle du domaine et de l'apprenant, le « modèle du tuteur » contrôle les interactions entre le système et l'apprenant de manière continue pour assurer l'adaptation des stratégies du tuteur à l'apprenant. Le comportement du tuteur doit s'exécuter en temps réel et le principal défi du modèle de tuteur est d'identifier quand et comment intervenir pour aider l'apprenant (Razzaq et Heffernan 2009).

La fonction du tutorat dans les ITSs est divisée en deux parties : le diagnostic des connaissances de l'apprenant (par exemple, la détection des causes des erreurs) et le choix des stratégies de remédiation (Bourdeau et Grandbastien 2010). Nous proposons de présenter les modèles classiques permettant de réaliser ces fonctions de tutorat selon trois catégories :

- les méthodes basées sur la métacognition,
- les méthodes basées sur l'intelligence artificielle et l'analyse de trace,
- les méthodes basées sur des architectures cognitives.

Méthodes basées sur la métacognition

Selon Flavell, la « métacognition » se rapporte à la connaissance qu'un apprenant a de ses propres processus cognitifs qui touchent par exemple aux propriétés pertinentes pour l'apprentissage (Flavell 1976). La métacognition peut aboutir à une décision explicite de la part de l'apprenant de modifier son activité cognitive (Noël 1997). En d'autres termes, la métacognition est la représentation qu'un apprenant possède de ses propres connaissances et sur la façon dont il peut les construire et les utiliser.

Dans le domaine de la métacognition pour l'apprentissage, « MetaTutor » est un des outils de référence (Azevedo et al. 2011). MetaTutor est un outil d'apprentissage hypermédia (page web, image et vidéo) initialement destiné à la formation sur le système sanguin humain. Il s'appuie sur le principe de métacognition et sur l'apprentissage autorégulé. L'idée est d'aider l'apprenant à se fixer lui-même ses propres buts dans l'organisation de son apprentissage. Les fonctionnalités de MetaTutor sont présentées sur la Figure 2.24.

Les interactions avec l'apprenant sont gérées par un agent pédagogique (Figure 2.24, en haut à droite) dont le rôle est d'aider l'apprenant à planifier son apprentissage, d'observer le comportement de l'apprenant et de réaliser des actions pédagogiques. Par exemple, l'agent peut aider l'apprenant à choisir un sous-but dans les étapes de son apprentissage (Figure 2.24, *subgoals*). Cet outil propose également une palette, sur une échelle de Likert à six niveaux, qui permet à l'apprenant d'évaluer son niveau de compréhension sur

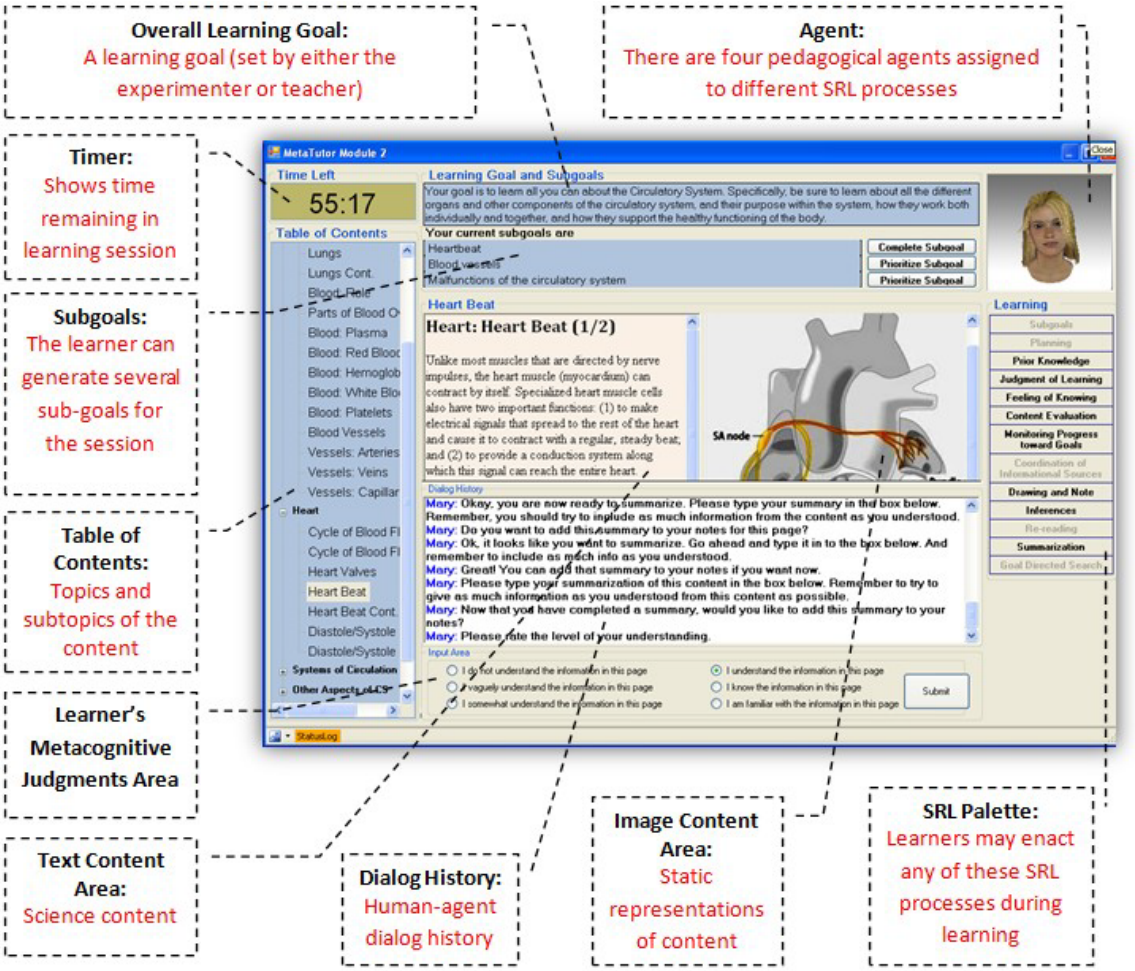


FIGURE 2.24 – Fonctionnalités de MetaTutor liées à l'apprentissage autorégulé (KAY et al. 2013).

le sujet abordé. En complément, l'agent pédagogique peut lui proposer un questionnaire. La comparaison entre les résultats de ce questionnaire et l'auto-évaluation de l'apprenant permet ainsi à l'agent pédagogique d'adapter son comportement à celui de l'apprenant.

Dans (KAUTZMANN, CARLOTTO et al. 2016), l'ITS proposé ne modélise pas explicitement les connaissances de l'apprenant, mais propose un tuteur qui demande à l'apprenant d'avoir une réflexion sur son propre apprentissage par le biais de questions métacognitives. L'ITS est représenté par un personnage animé qui peut se placer n'importe où sur l'interface. Il interagit avec l'apprenant à travers des questions sous forme de bulles. Le but de ces questions est d'encourager l'apprenant à réfléchir sur ses propres connaissances. Par exemple, en demandant à l'apprenant de caractériser le problème à résoudre, d'expliciter ses propres connaissances sur le problème avant même d'essayer de les résoudre et d'identifier des problèmes similaires qu'il a déjà pu résoudre (Figure 2.25).

Afin de contrôler les interventions du tuteur, le système calcule un indice appelé KMA (Knowledge Monitoring Assesment) qui mesure la capacité de l'apprenant à évaluer ses propres connaissances (TOBIAS et EVERSON 1996). Le principe de cet indice est de com-

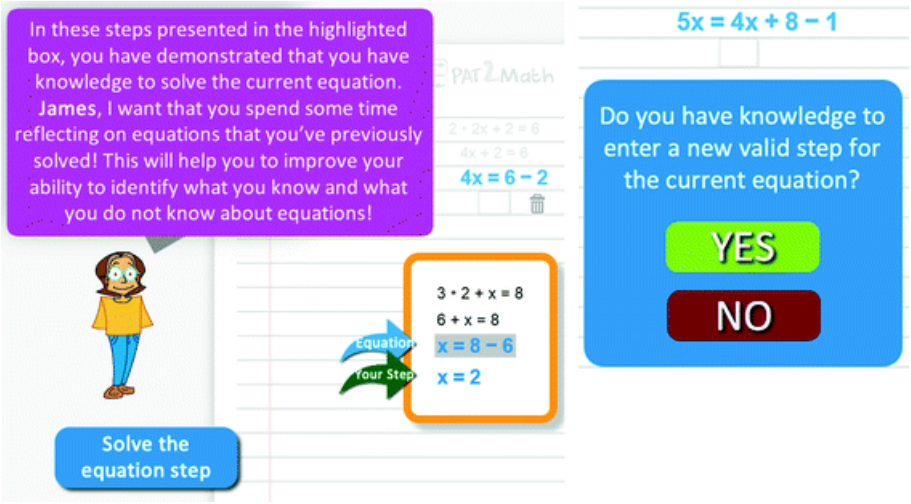


FIGURE 2.25 – Exemple de questions posées à l'apprenant par le tuteur sur ses propres connaissances (KAUTZMANN et JAQUES 2017).

parer les performances observées de l'apprenant avec l'évaluation que l'apprenant fait sur ses propres connaissances. Il existe quatre valeurs pour cet indice. Par exemple, l'apprenant estime qu'il sait résoudre le problème et il arrive effectivement le résoudre, ou l'apprenant estime qu'il sait résoudre le problème mais il ne réussit pas, etc. Le tuteur s'appuie sur ces valeurs pour décider d'agir ou non.

Méthodes basées sur l'intelligence artificielle

Les approches basées sur l'« intelligence artificielle » ne visent pas à modéliser les mécanismes de l'apprentissage humain, mais visent plutôt à identifier, en fonction des actions de l'apprenant, les nouvelles connaissances à lui apporter. Une des techniques classiques pour réaliser ce type de comportement est l'utilisation des réseaux bayésiens.

Un réseau bayésien est un modèle graphique, représenté par un graphe acyclique dont les nœuds représentent des variables aléatoires et les liens représentent les dépendances entre ces variables. Dans le cadre de la modélisation d'un apprenant, les variables aléatoires représentent les probabilités qu'un apprenant maîtrise un concept ou qu'il y a eu accès et les liens représentent les dépendances logiques entre ces concepts.

Le tuteur Andes (VANLEHN et al. 2005) est un des tuteurs intelligents les plus connus utilisant cette technique. ANDES est destiné à l'apprentissage de la physique. Il utilise des réseaux de croyances bayésiens pour inférer quels concepts l'apprenant pourrait connaître mais qu'il n'a pas encore mis en œuvre. Pour chaque problème résolu par un apprenant, ANDES construit un réseau bayésien statique dont les nœuds et les liens représentent comment les différentes étapes d'une résolution de problème découlent des étapes précédentes et des règles de physique (CONATI et al. 2002). La Figure 2.26 montre une représentation d'un réseau bayésien pour décrire les concepts impliqués dans la seconde loi de Newton qui est un des sujets de formation du tuteur ANDES.

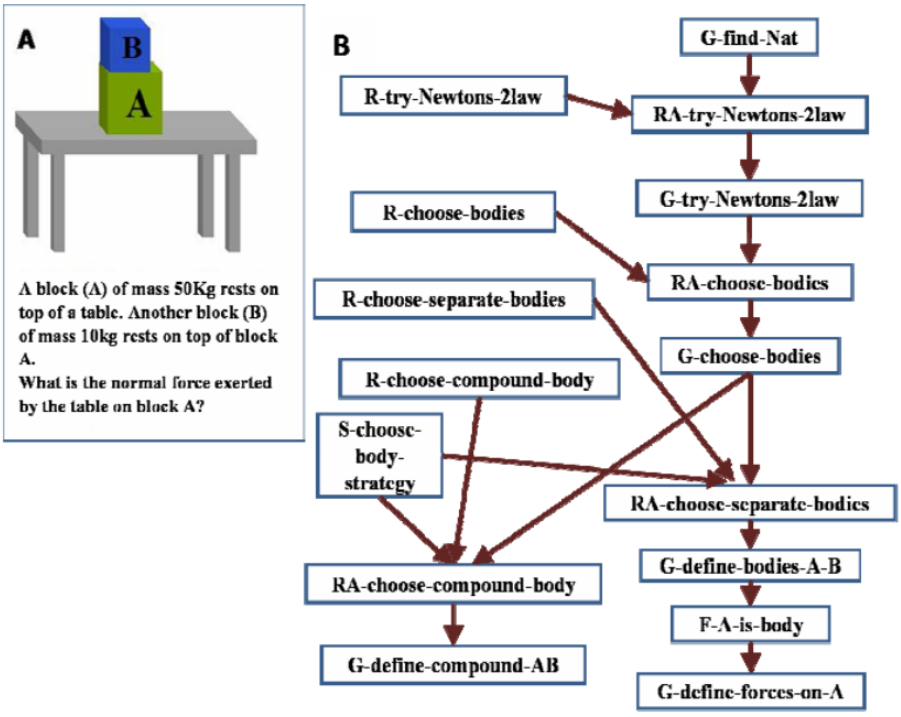


FIGURE 2.26 – Représentation d'un problème de physique à l'aide d'un réseau bayésien en utilisant le tuteur ANDES.

Le modèle de l'apprenant est mis à jour à la fin de chaque problème. Ce modèle représente la probabilité qu'un apprenant ait compris un concept spécifique de physique. Si le tuteur ANDES considère que l'apprenant n'a pas compris ce concept, alors une courte leçon sur ce concept sera proposée à l'apprenant. De plus, le chemin utilisé par l'apprenant, dans le réseau bayésien, pour résoudre le problème est également utilisé pour déterminer quel exercice sera ensuite proposé à l'apprenant.

Actuellement, les techniques à base d'apprentissage artificiel sont de plus en plus utilisées. Ces techniques servent souvent à classifier les apprenants dans des classes (au sens de classification en intelligence artificielle) partageant les mêmes propriétés. Ces travaux font l'hypothèse que les apprenants appartenant à la même classe nécessitent les mêmes interventions de la part du tuteur. Par exemple, dans les travaux de Mojarad et ses collègues (MOJARAD et al. 2018), les données utilisées proviennent d'un système appelé ALEKS utilisé par 18 classes de lycées pour l'apprentissage de l'algèbre. Les élèves utilisent le système en dehors des cours pour faire leurs devoirs. L'algorithme d'apprentissage artificiel proposé s'appuie sur un total de 5725 évaluations réalisées par un total de 700 élèves dans le cadre du système ALEKS. Le but de cet algorithme est de classer les apprenants dans différentes catégories. Le premier problème à résoudre dans par ce type d'algorithme est d'identifier le nombre de catégories et le second est de classer chaque apprenant dans une de ces catégories. Dans ce travail, l'algorithme a déterminé 5 catégories d'apprenants. Ces catégories peuvent alors être utilisées par les enseignants

pour adapter leurs stratégies face à un apprenant particulier.

Pour qu'ils puissent fonctionner, ces systèmes à base d'apprentissage artificiel ont besoin d'une quantité massive de données. Ces données représentent essentiellement les interactions de l'apprenant avec le système. Ceci est communément appelé, les traces d'apprentissage. Dans le domaine de l'analyse des traces de l'apprentissage, le recueil de l'information détaillée sur l'expérience de l'apprenant se fait généralement dans le cadre des plates-formes d'apprentissage en ligne. Les plates-formes d'apprentissage telles que les systèmes de gestion de l'apprentissage (LMS, Learning Management System) et les cours en ligne ouverts et massifs (MOOC, Massive Open Online Course) ont la capacité de suivre et de recueillir d'importantes quantités de données sur l'expérience de l'apprenant. Un des problèmes actuels est de permettre l'échange de données entre ces différents types de plates-formes. xAPI (Experience API)[9], connue initialement sous le nom de Tin Can API, est un standard récent d'interopérabilité dans ce domaine, largement adopté par différents environnements d'apprentissage. L'interface xAPI présente un modèle de données flexible pour les traces d'apprentissage et le rendement de l'apprenant (Lim 2018). L'ensemble de ces traces est stocké dans des LRS (Learning Record Score). Un LRS n'effectue par lui-même aucun traitement, mais peut partager ses informations avec d'autres LRS ou d'autres logiciels. Les algorithmes d'apprentissage artificiel peuvent donc se connecter à ces LRS pour accéder à l'important volume de données dont ils ont besoin pour fonctionner.

Méthodes basées sur les sciences cognitives

Les approches basées sur les « sciences cognitives » sont en générale divisées en deux catégories : les modèles à base de règles et les modèles cognitifs (Woolf 2010). Ces deux catégories considèrent l'apprentissage comme un processus computationnel. C'est-a-dire, qu'elles sont basées sur la modélisation des processus utilisés par l'apprenant pour résoudre un problème.

Les modèles à base de règles représentent l'ensemble des connaissances à acquérir par des règles de la forme « `Si` *condition* `Alors` *action* ». Ces règles doivent être écrites de manière exhaustive par un expert. Le tuteur fonctionnant sur ce principe, ne cherche pas à évaluer si l'apprenant possède l'ensemble de ces règles, mais détermine quand une condition d'une règle a été violée. Dans ce cas, le tuteur identifie les connaissances manquantes de l'apprenant. À l'inverse, les méthodes cognitives tentent d'identifier les règles qui sont employées par l'apprenant pour résoudre un problème. Ces règles sont également construites par un expert et le raisonnement employé par celui-ci pour résoudre un problème à l'aide de ces règles est également modélisé. Le modèle cognitif le plus connu dans ce contexte est l'architecture cognitive Act-r.

9. `https://xapi.com/statements-101/`

ACT-R (Adaptive Control of Thought-Rational) (ANDERSON et al. 2004) est une architecture cognitive qui peut être abordée selon deux points de vue. Du point de vue des chercheurs en psychologie cognitive, ACT-R est considérée comme un cadre pour représenter les différents aspects de l'intelligence humaine. Du point de vue des informaticiens, elle représente une architecture logicielle qui permet de générer des comportements intelligents pour les agents autonomes, comme par exemple des tuteurs intelligents tels que CTAT (Cognitive Tutor Authoring Tools) (ALEVEN et al. 2006).

Dans le cadre de notre travail, qui porte sur l'utilisation de la réalité virtuelle pour la formation, cette architecture est intéressante, car elle prend en compte l'incarnation de l'agent dans un environnement virtuel et prend également en compte ses perceptions (vision) et ses actions (moteur) dans l'environnement. L'architecture globale d'ACT-R est présentée dans la Figure 2.27.

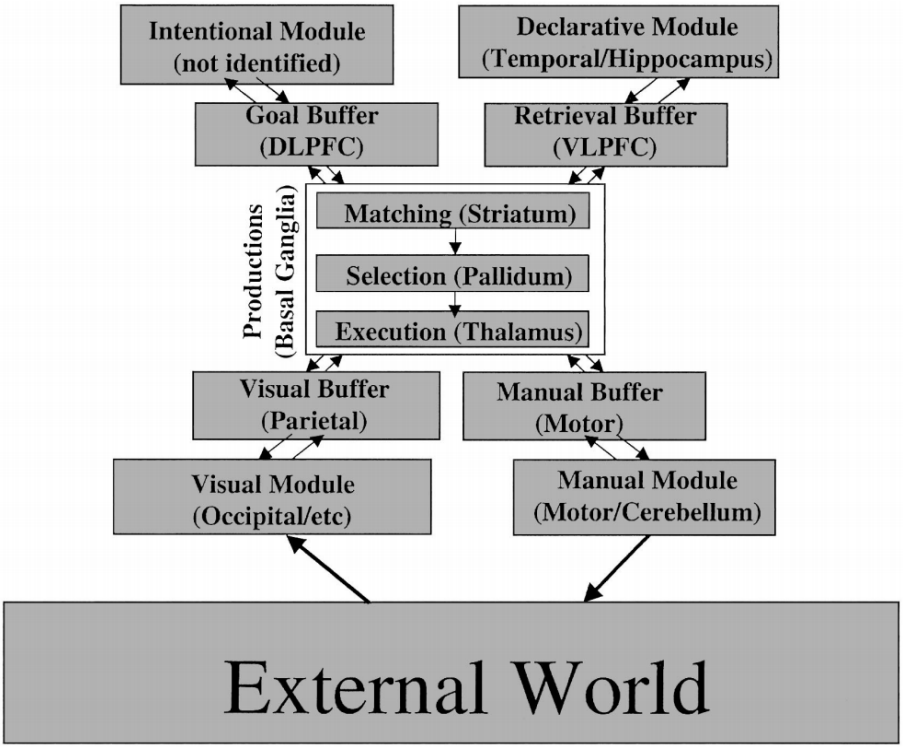


FIGURE 2.27 – Architecture globale d'ACT-R (ANDERSON et al. 2004).

Le module déclaratif (Figure 2.27, Declarative Module) contient l'ensemble des connaissances déclaratives. Ces connaissances sont représentées par des *chunks*. Les *chunks* sont des unités élémentaires de stockage, mais ils peuvent avoir plusieurs formes et plusieurs niveaux de complexité. Le programmeur doit donc tout d'abord définir tous les types de *chunks* que le module déclaratif peut contenir. Ceci est fortement dépendant du domaine d'apprentissage. Puis le programmeur doit lister l'ensemble des connaissances déclaratives en instanciant ces types de *chunks*. Ceci constitue les connaissances initiales de l'agent. Cependant, de nouvelles connaissances déclaratives peuvent être gé-

nérées lors de l'exécution du comportement de l'agent.

Le module procédural (Figure 2.27, Productions) est constitué d'un ensemble de règles de production (système à base de règles), composé d'une partie de tests et d'une autre partie d'actions. La partie de tests porte sur le contenu des modules à travers des *buffers*, par exemple le contenu du module déclaratif ou du module de perception (Figure 2.27, Visual Module). La partie d'actions porte sur la modification du contenu du module déclaratif et du module moteur (Figure 2.27, Manual Module). La partie d'actions peut particulièrement porter sur l'ajout d'un but à atteindre. Le programmeur doit prédéfinir l'ensemble de ces règles de production.

Enfin, le module de but (Figure 2.27, Intentional Module) contient un ensemble de *chunks* issu du module déclaratif.

ACT-R dispose d'un module d'inférence s'appuyant sur plusieurs phases : *Matching*, *Selection* et *Execution* pour atteindre les buts fixés. Ce module fait justement la force d'ACT-R, parce qu'il réalise de manière automatique le raisonnement sur les faits et les règles de production tel qu'un humain le ferait. Cependant, ce raisonnement n'est pas, par lui même, adaptatif. C'est-à-dire qu'il n'est pas possible de rajouter des règles de production en cours d'exécution.

Généralement, dans le cadre des tuteurs intelligents, ACT-R est utilisée pour représenter le raisonnement de l'expert. Le tuteur compare les actions de l'apprenant et essaye d'identifier quelles règles de production sont en cours d'exécution. En cas d'erreur de l'apprenant, le tuteur peut s'appuyer sur ces règles pour l'aider à résoudre le problème.

Dans cette section, nous avons montré que les modèles basés sur les systèmes tutoriels intelligents permettent d'évaluer les progrès de l'apprenant en temps réel ou à posteriori. D'autres travaux ont montré que les données multimodales peuvent être également intéressantes pour inférer l'état de l'apprenant au cours de l'apprentissage. Par exemple, les données multimodales telles que les expressions faciales, la posture et les gestes peuvent prédire l'état affectif de l'apprenant, tel que la frustration et l'engagement (GRAFSGAARD et al. 2013). Les travaux de Vail et ses collègues (VAIL et al. 2016) utilisent l'analyse des expressions faciales de l'apprenant (Figure 2.28) à la suite d'une question du tuteur, pour évaluer si le tuteur doit changer sa stratégie pédagogique.

Les expressions faciales de l'apprenant sont ainsi intéressantes pour influencer le comportement du tuteur. Nous montrons dans la section suivante que l'inverse est également intéressant. Les données multimodales du tuteur sont également des informations pertinentes pour l'apprenant et qu'il est donc intéressant de l'incarner.

2.2.3 Agents pédagogiques incarnés

La prise en compte des données multimodales du tuteur et de l'apprenant semble être pertinente lors des interactions en situations pédagogiques. Les agents pédagogiques ayant ces capacités d'interactions naturelles sont appelés les « agents pédagogiques ani-

(a) AU4 Brow Lowerer (b) AU5 Upper Lid Raiser (c) AU15 Lip Corner Depressor (d) AU23 Lip Tightener

Figure 2.28 – Exemples d'expressions faciales reconnues dans (Vail et al. 2016).

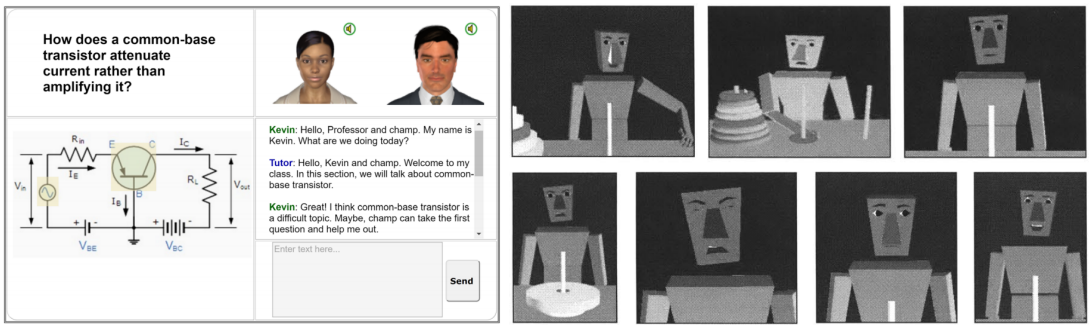
més » (Johnson, Rickel et al. 2000). Les agents pédagogiques animés semblent avoir des effets positifs sur l'engagement des apprenants (Rowe et al. 2007) et sur l'efficacité de l'enseignement (Kokane et al. 2014). Des expériences, comme celle réalisée par Lester et ses collègues (Lester et al. 1997), montrent que l'utilisation des agents pédagogiques animés peut motiver l'apprenant à accomplir les tâches requises. De plus, Baylor et ses collègues (Baylor et al. 2003) et Mayer et DaPra (Mayer et DaPra 2012) ont constaté que les apprenants avaient de meilleures performances d'apprentissage lorsque le tuteur était animé que lorsqu'il était statique. Moreno et ses collègues (Moreno et al. 2001) et Baylor et ses collègues (Baylor et al. 2003) ont constaté que la voix de l'agent contribue de façon significative à l'apprentissage, parfois même plus que l'animation de l'agent. Nous considérons donc que l'intérêt des agents pédagogiques animés repose surtout sur le fait qu'il soit incarné dans la situation pédagogique.

Nous montrons tout d'abord des exemples d'agents incarnés qui permettraient d'améliorer l'apprentissage de l'apprenant. Dans un deuxième temps, nous présentons le principe d'agent conversationnel animé qui permet d'incarner ces agents dans un environnement virtuel.

Plusieurs agents pédagogiques incarnés ont déjà été développés. « AutoTutor » est capable de répondre avec l'expression faciale appropriée à l'état émotionnel de l'apprenant et de faire preuve d'empathie (Graesser et D'Mello 2012). Il utilise le langage naturel pour communiquer avec les apprenants afin de leur enseigner des sujets scientifiques (comme la physique, la biologie, l'informatique, etc.). Récemment, le système a été amélioré pour permettre les conversations de groupe (Graesser, Li et al. 2014) : un tuteur virtuel, un apprenant virtuel et un apprenant humain. Cette version du modèle est utilisée pour l'apprentissage de l'électronique (Graesser, Hu et al. 2018) (Figure 2.29a).

Burleson a développé un système de tutorat capable de reconnaître l'état affectif de

l'utilisateur et l'utilise pour déterminer le comportement de l'agent virtuel (BURLESON 2006). L'agent répond à la frustration de l'apprenant en faisant preuve d'empathie et en favorisant le dialogue. De plus, il peut refléter certains comportements non verbaux qui semblent avoir une influence sur la persuasion, le goût et le rapport social (Figure 2.29b).



(a) (b)

FIGURE 2.29 – Exemple de situation d'apprentissage avec AutoTutor (a) Exemple de comportements non verbaux dans (BURLESON 2006) (b).

Le projet « TARDIS » (JONES et SABOURET 2013) fournit une application basée sur des scénarios pour aider les utilisateurs à améliorer leurs compétences sociales en entretien d'embauche. Un agent virtuel, jouant le rôle du recruteur, conduit un entretien d'embauche avec un utilisateur humain en suivant un scénario prédéterminé. Pendant l'exécution du dialogue, le comportement émotionnel et social de l'agent est déterminé, en temps réel, en fonction des signaux non verbaux de l'utilisateur, comme l'expression faciale, la posture, la direction de la tête, l'intonation vocale (Figure 2.30).

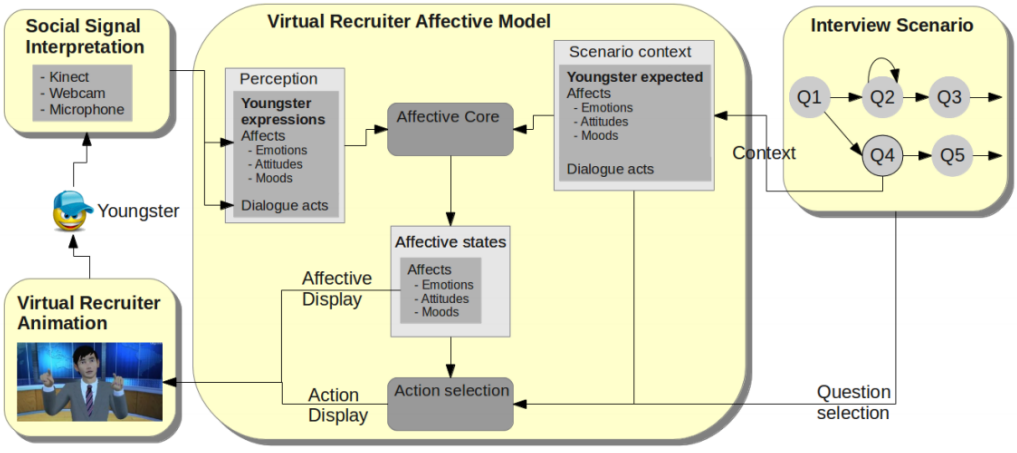


FIGURE 2.30 – Architecture du projet TARDIS.

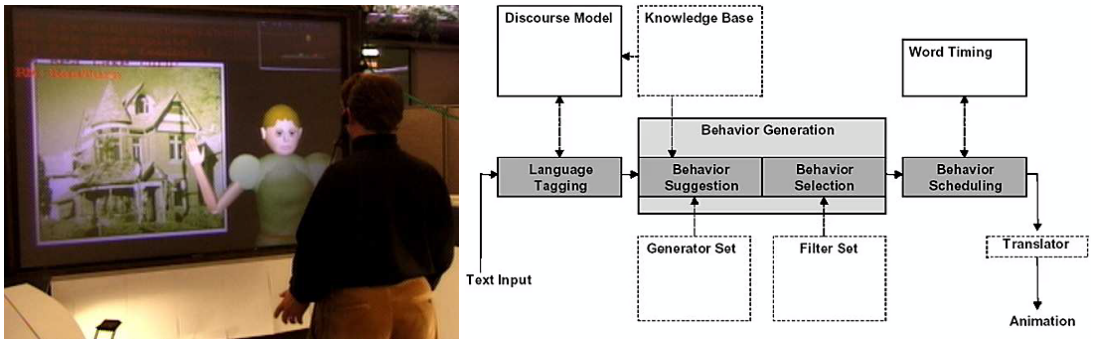
Afin d'avoir une communication naturelle, entre le tuteur et l'apprenant, il est possible d'intégrer un Agent Conversationnel Animé (ACA) comme interface entre le système et l'utilisateur. Nous présentons en détails ce type d'interface dans la section suivante.

2.2.3.1 Plates-formes d'agents conversationnels animés

Un ACA est une interface homme-machine, souvent représenté par une figure anthropomorphe, capable de communiquer de façon autonome avec les utilisateurs par des signaux verbaux et non verbaux, tels que des expressions faciales, des gestes, la parole, etc. Pour interagir naturellement avec les utilisateurs, les ACAs doivent être capable de montrer un comportement humain crédible non seulement en parlant mais également en écoutant.

Afin de faciliter la conception d'agents conversationnels animés capables d'interagir avec l'utilisateur, plusieurs plates-formes ont été développées.

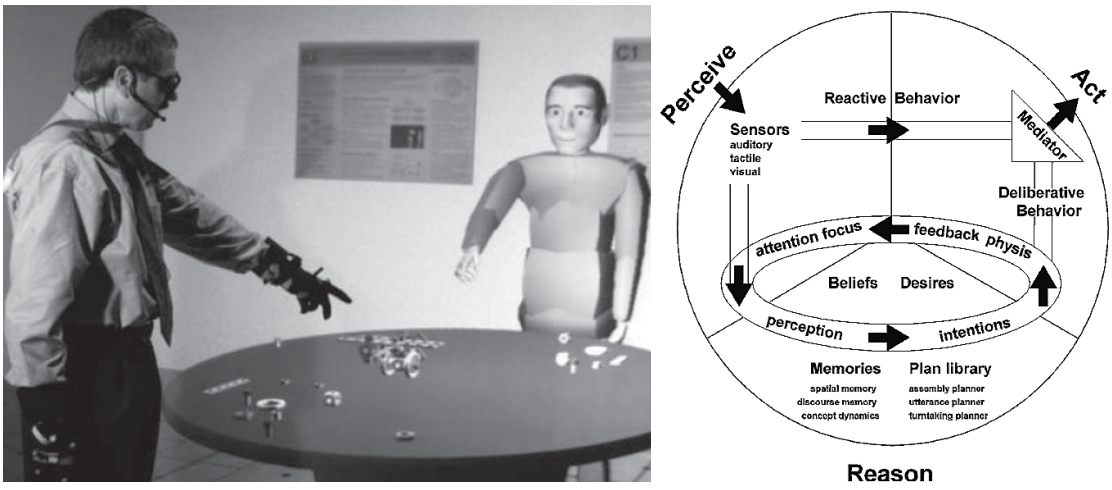
L'agent « REA » (Real Estate Agent) est un des premiers ACAs à avoir été implémenté (Cassell, Bickmore et al. 1999). Il était destiné à une application spécifique sur la vente de bien immobilier. REA était capable de reconnaître la parole et les gestes de l'utilisateur à travers des caméras et des microphones, de planifier un comportement en s'appuyant sur une base de connaissances de bien immobilier et de générer des gestes et des paroles avec des intonations particulières. Plus tard, afin de rendre générique l'utilisation de REA, les auteurs ont proposé une boîte à outils appelée BEAT (Behavioral Expression Animation Toolkit) (Cassell, Vilhjálmsson et al. 2004). BEAT est une architecture modulaire qui prend en entrée le texte produit par l'utilisateur et génère le comportement non verbal de l'ACA sous forme d'expression faciale, de gestes, de voix et d'intonation. BEAT s'appuie sur des représentations de connaissances exprimées dans un langage propriétaire. Cette architecture est présentée sur la Figure 2.31b.



(a) (b)

Figure 2.31 – L'agent REA (a) Architecture BEAT (b).

Par la suite l'agent « MAX » (Multimodal Assembly eXpert) a été développé (Kopp, Jung et al. 2003). MAX enrichit les capacités d'interaction de l'utilisateur et permet un usage dans des situations immersives comme dans un CAVE. Un de ses premiers usages a été l'assistance aux procédures de montage (Figure 2.32a). Similairement à REA, MAX est capable de générer des comportements non verbaux comme des expressions faciales et des gestes. MAX s'appuie sur une architecture cognitive évoluée s'inspirant des BDI (Belief Desire Intention) (Rao et Georgeff 1991) et implémentée à l'aide l'architecture cognitive Act-r(Figure 2.32b).



(a) (b)

FIGURE 2.32 – L'agent MAX (KOPP, JUNG et al. 2003) (a) Architecture de MAX (b).

Ces architectures d'ACA ont adopté un principe de modularité. La communauté des chercheurs dans ce domaine a débuté un travail de normalisation permettant l'échange d'information entre les modules afin de faciliter l'interopérabilité entre les différents modules des différentes architectures. Il s'agit du langage BML (Behavior Markup Langage) (KOPP, KRENN et al. 2006) et plus généralement de l'architecture SAIBA (VILHJÁLMSSON et al. 2007).

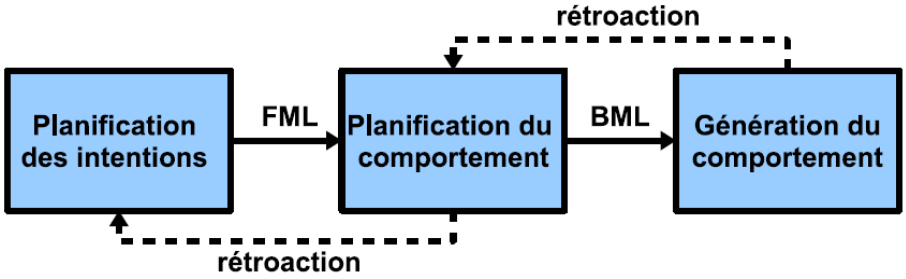


FIGURE 2.33 – Architecture SAIBA selon (DE SEVIN et al. 2010).

L'architecture SAIBA est divisée en trois modules et propose deux langages normalisés pour communiquer entre ces modules (Figure 2.33).

Le module de « Planification des intentions » définit les buts et l'état émotionnel de l'agent. Dans notre cas, il s'agira des buts générés par le comportement du tuteur. Le module de « Planification des comportements » correspond à la génération des signaux communicatifs (parole, expression faciale, etc.) issus de ces buts. Enfin, le module de « Génération du comportement » correspond à l'exécution effective de ces signaux sur la représentation graphique de l'ACA (par exemple, *Text-To-Speech*, animation faciale, etc.).

Nous présentons par la suite deux plates-formes d'agents conversationnels animés, « GRETA » et « Virtual Human Toolkit » qui font références dans le domaine et qui s'ap-

puient fortement sur l'architecture SAIBA.

GRETA

L'architecture « GRETA » (DE SEVIN et al. 2010) propose une implémentation de l'ensemble des modules de l'architecture SAIBA et respecte ses formats de données BML et FML (Function Markup Language). L'architecture de GRETA est présentée sur la Figure 2.34 qui montre que l'ensemble des modules communiquent entre eux à travers un tableau blanc appelé « Psyclone » (THÓRISSON et al. 2005) et se synchronisent par une « Horloge centrale ». Le module le plus avancé dans GRETA est le module de « Planification du comportement ». En effet, à partir d'une seule intention communicative fournie par le module de « Planification des intentions » GRETA est capable de produire un ensemble de comportements non verbaux différents.

Le module de « Planification des intentions de l'interlocuteur » est essentiellement basé sur trois modules de rétroactions (réactives, cognitives et imitation). Il s'agit des signaux multimodaux fournis par l'auditeur sans tenter de prendre la parole (YNGVE 1970). Le module de « rétroactions réactives » s'appuie sur des règles probabilistiques pour générer des signaux de rétroactions lorsqu'un comportement de l'utilisateur est reconnu (mouvement de tête, variation du ton de la voix, etc.). Le module de « rétroactions cognitives » se base sur l'état mental de GRETA et sur ses croyances concernant le discours entre l'agent et l'utilisateur pour générer les rétroactions. Enfin, le module de « rétroactions d'imitation » génère des comportements d'imitation qui peuvent être observés quand l'utilisateur est complètement engagé dans une conversation.

Le module de « Planification du comportement » a pour objectif de transformer l'intention communicative en un ensemble de comportements à exécuter par le module de « Génération de comportement ». Le choix de ces comportements à partir de l'intention communicative s'appuie également sur les caractéristiques de base de l'agent. Ces caractéristiques portent sur les préférences de l'agent à utiliser certaines modalités (tête, regard et visage). Le comportement à exécuter est ainsi calculé en fonction des préférences de l'agent paramétrées par son état émotionnel calculé dynamiquement en fonction de l'évolution du discours. Ainsi, une même intention communicative pourra être exprimée de manière différente au cours de l'interaction entre l'utilisateur et l'agent.

L'objectif des modules « Génération du comportement » et « Moteur d'animation FAP-BAP » est de générer concrètement l'animation de l'ACA. Dans GRETA, ces modules s'appuient sur le format « MPEG-4 » et sur les paramètres d'animation faciale (FAP) et celles du corps (BAP). La parole est générée par des moteurs de *Text-To-Speech* tel que OpenMary [10] et Cereproc [11].

GRETA a été utilisée dans plusieurs applications, par exemple, en tant que patient virtuel pour former des médecins à annoncer les mauvaises nouvelles (Figure 2.35a). Récemment l'équipe s'intéressent à l'expression du rire pour les agents virtuels (Figure

10. `http://mary.dfki.de`
11. `https://www.cereproc.com/`

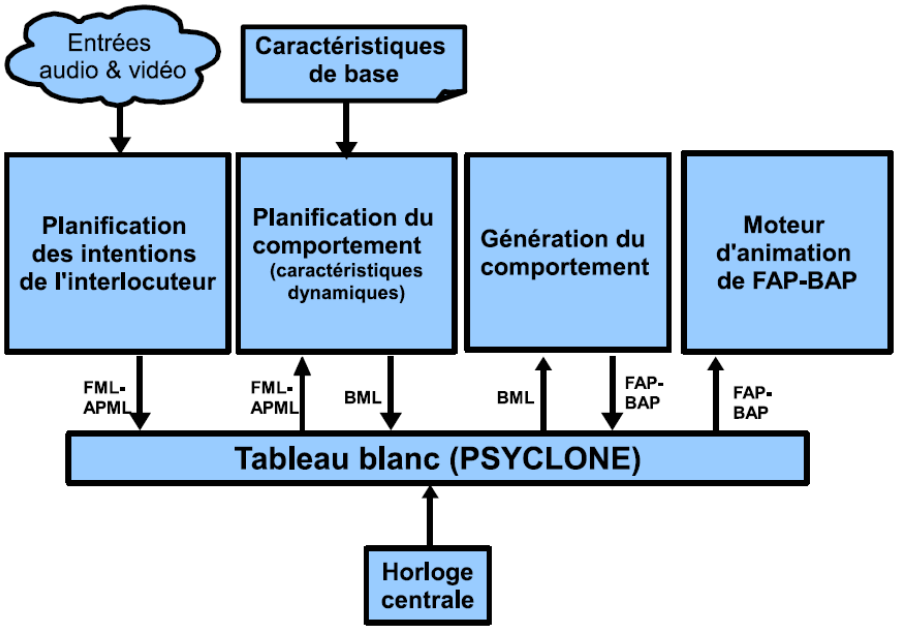


FIGURE 2.34 – Architecture de GRETA (DE SEVIN et al. 2010).

2.35b).

(a) (OCHS et al. 2017) (b) (DING et al. 2014)

FIGURE 2.35 – Utilisation de GRETA en tant que patient virtuel (a) Génération d'animation de rires avec GRETA (b).

Virtual Human Toolkit

« Virtual Human Toolkit » (HARTHOLT et al. 2013) (VHT) est une boîte à outils pour permettre la conception rapide d'agents conversationnels animés. VHT fournit plusieurs bibliothèques qui peuvent être liées entre elles et qui couvrent les fonctionnalités proposées par l'architecture SAIBA. La Figure 2.36 présente l'ensemble des bibliothèques proposées, ainsi que les liens entre elles. VHT peut être divisée en quatre grands sous-modules.

Les modules « MultiSense » et « AcquireSpeech » ont pour objectif de détecter les

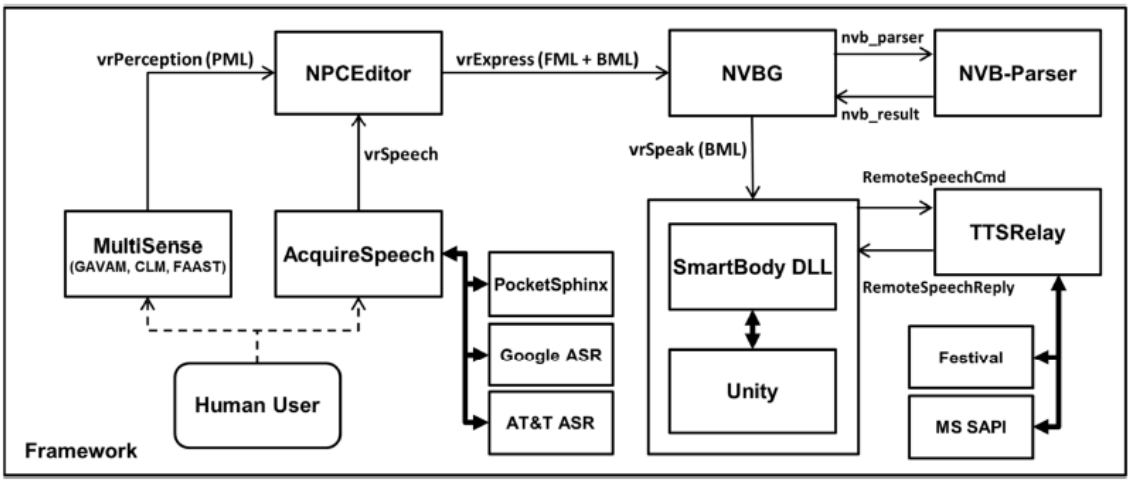


FIGURE 2.36 – Architecture de Virtual Human Toolkit (HARTHOLT et al. 2013).

comportements audio-visuels et non verbaux de l'utilisateur. Pour cela, ces modules peuvent utiliser différents types de reconnaissance vocale (PocketSphinx, Google ASR, etc.). Ces modules utilisent également des outils tels que Microsoft Kinect ou des caméras RGB pour détecter les gestes et les expressions faciales de l'utilisateur. Le module « MultiSense » est capable de faire la fusion entre les données d'entrée provenant de différents capteurs. Les informations provenant de ces modules sont transmises au module « NPCEditor ».

Le module « NPCEditor » est le composant qui permet de traiter les textes reçus en langage naturel. Dans VHT, ce module s'appuie sur une classification statistique qui permet de faire la correspondance entre un texte énoncé par l'utilisateur et une réponse de l'agent. *À priori*, la réponse de l'agent ne dépend pas d'un raisonnement sur une base de connaissance sur l'environnement virtuel, mais c'est le programmeur qui doit créer la table de correspondance. Ce module correspond au module de « Planification des intentions » dans l'architecture SAIBA. Le texte à énoncer ainsi que les propriétés telles que l'émotion et l'attitude sont transmises au module « NVBG ».

Le module « NVBG » (Non Verbal Behavior Generator) génère le comportement non verbal de l'agent. Pour générer ce comportement, ce module s'appuie d'une part sur un système à base de règles que le programmeur doit rédiger et d'autre part, sur l'interprétation du texte à énoncer. Ce module représente le module de « Planification du comportement » dans l'architecture SAIBA et génère les fichiers BML qui seront exécutés par le module « SmartBody ».

« SmartBody » (THIEBAUX et al. 2008) a pour objectif la génération du comportement verbal et non verbal de l'agent (LEE et MARSELLA 2006). Ce module génère le texte, grâce à une API de *Text-To-Speech* et gère le regard, les gestes et les mouvements de la tête de l'agent. Le module « SmartBody » enrichit le langage BML afin de permettre la réalisation d'animation complexe et le déplacement de l'agent dans l'environnement virtuel.

Virtual Human Toolkit a été utilisée dans différentes applications portant sur les relations sociales et émotionnelles entre un ACA et un utilisateur humain. La Figure 2.37a montre un exemple pour l'apprentissage de la négociation et la Figure 2.37b d'un agent intervieweur dans le domaine la santé.

(a) (DeVault, Mell et al. 2015) (b) (DeVault, Artstein et al. 2014)

Figure 2.37 – SAM, un agent de négociation (a) SimSenSei Kiosk, un agent intervieweur dans le domaine de la santé (b).

2.3 Bilan

Dans ce chapitre nous avons montré que la réalité virtuelle permet d'apprendre différents types de compétences, comme les compétences techniques sur les systèmes industriels. Dans le cadre de notre travail de recherche, nous souhaitons faire en sorte que la situation pédagogique en environnement virtuel s'adapte en temps réel en fonction de l'évolution des compétences de l'apprenant. Dans notre travail, les compétences qui nous intéressent sont celles liées aux connaissances sur des systèmes techniques industriels et sur la capacité à réaliser des procédures sur ces systèmes.

Pour réaliser cette adaptation, l'agent pédagogique doit être capable de raisonner sur les connaissances représentées dans l'environnement virtuel. Ces connaissances portent non seulement sur les connaissances liées aux métiers mais également sur les connaissances liées aux stratégies pédagogiques souvent représentées par des scénarios pédagogiques. Dans la Section 2.1, nous avons présenté différents modèles permettant de rendre explicites ces connaissances. La suite logicielle #Seven propose un langage formalisé à l'aide des réseaux de Petri, qui permet un raisonnement de la part du système sur les actions pédagogiques à réaliser. Cependant, #Seven s'appuie sur #Five pour représenter les concepts et les objets du domaine. #Five est une API destinée aux programmeurs qui permet de faciliter la conception de l'environnement virtuel. Cependant, les concepts métiers ne sont pas explicités de manière générique. Ils ne sont donc pas accessibles au raisonnement du système afin de s'adapter à l'apprenant. Au contraire,

HUMANS est un modèle qui propose des langages basés sur les ontologies qui permet de décrire les activités humaines sur le système et également sur les concepts et les objets du modèle métier. Toutefois, HUMANS a été majoritairement utilisé pour l'acquisition des compétences non-techniques et est conçu pour générer dynamiquement et de manière autonome des nouvelles situations pédagogiques auxquelles l'apprenant n'a jamais été confronté. HUMANS ne s'appuie donc pas sur l'explicitation des scénarios pédagogiques par le formateur, ce qui ne rejoint pas l'objectif que nous souhaitons atteindre dans ce travail. En effet, nous souhaitons que le formateur puisse contrôler la manière dont les connaissances vont être présentées à l'apprenant. MASCARET est un modèle qui permet de représenter tous les aspects de la situation pédagogique : les concepts et les objets métiers, les activités humaines et les scénarios pédagogiques. Toutes ces connaissances sont explicites en cours d'exécution et permettent donc au système de raisonner sur ces connaissances afin d'adapter la situation pédagogique à l'apprenant. Dans la suite de ce travail, nous choisissons d'utiliser MASCARET pour représenter le métier à apprendre, les stratégies pédagogiques et l'évolution des connaissances de l'apprenant.

L'adaptation des situations pédagogiques à l'apprenant est un des objectifs fondamentaux des systèmes tutoriels intelligents. Nous avons présenté dans la Section 2.2 différents types de tuteurs intelligents et différentes méthodes de conception des comportements de tuteurs adaptatifs. Nous avons identifié trois catégories de méthodes. La première méthode porte sur le principe de métacognition qui ne tente pas de représenter les connaissances de l'apprenant, mais le pousse à raisonner lui-même sur ses propres connaissances. La seconde méthode basée sur les techniques d'apprentissage artificiel, ne permet pas non plus de représenter les connaissances de l'apprenant, mais analyse ses traces dans l'environnement pédagogique pour identifier dans quel classe d'apprenant il se situe et donc d'identifier quel type d'exercice il aura besoin de réaliser lors de la prochaine session d'apprentissage. Ces méthodes ne raisonnent pas sur les connaissances effectives de l'apprenant en temps réel. Elles ne peuvent donc pas convenir pour atteindre notre objectif. La troisième catégorie de méthodes identifiées s'appuie sur les sciences cognitives. Ces méthodes proposent de représenter explicitement les processus cognitifs utilisés par les utilisateurs humains lors de la résolution de problème. D'un point de vue informatique, ces méthodes ont été utilisées pour représenter le comportement intelligent des agents autonomes. Alors que, d'un point de vue psychologique, ces méthodes servent à étudier les processus cognitifs. Parmi les architectures informatiques proposées dans le cadre de ces méthodes, ACT-R est une des architectures les plus connues. Cependant, dans ACT-R, le traitement des informations en mémoire est une « boîte noire ». Elle peut être utilisée pour générer le comportement du tuteur mais pas pour représenter le flux de connaissances au sein du modèle de l'apprenant. Pour cette raison, nous proposons notre propre architecture permettant d'expliciter les processus cognitifs de l'apprenant mis en jeu lors de l'apprentissage de procédures sur des systèmes techniques. Ces processus portent essentiellement sur la mémorisation, nous

nous inspirons donc de la théorie de la mémoire d'Atkinson et Shiffrin (ATKINSON et SHIFFRIN 1968), que nous instancions dans le contexte de l'apprentissage des procédures, tout en s'inspirant des concepts d'ACT-R.

Dans la Section 2.2.3 nous avons montré que l'incarnation du tuteur dans l'environnement virtuel favorise l'apprentissage. Étant donné que notre travail ne porte pas sur la proposition d'une nouvelle méthode d'incarnation du tuteur dans l'environnement virtuel, nous proposons de nous appuyer sur les plates-formes génériques d'agents conversationnels animés existantes. Parmi les plates-formes présentées, GRETA est la plate-forme qui propose aux programmeurs une interaction de plus haut niveau. C'est-à-dire, qu'il suffit de transmettre à GRETA l'intention communicative de l'agent pour que la plate-forme génère automatiquement les comportements verbaux et non verbaux adéquats. Nous choisissons donc d'utiliser cette plate-forme pour incarner notre agent pédagogique dans l'environnement virtuel.

II Proposition et Validation

3 Modèle MEMORIA

Rappelons la définition classique des systèmes tutoriels intelligents (ITS : Intelligent Tutoring System). Il s'agit des environnements d'apprentissage informatisés qui visent à imiter et simuler le comportement d'un tuteur humain dans ses capacités d'expert pédagogue et d'expert du domaine. Les ITS disposent de capacités de raisonnement. Ils évaluent les connaissances acquises par l'apprenant, en comparant ses activités aux informations du domaine et proposent des assistances adaptées. Selon Woolf (Woolf 1992), un ITS est composé de quatre composantes principales, jouant chacune un rôle spécifique : le modèle du domaine, le modèle de l'apprenant, le modèle du tuteur et l'interface de communication (Figure 2.21).

Dans la Figure 3.1, nous présentons les idées originales que nous proposons pour formaliser les quatre composantes d'un ITS.

Nous choisissons de définir notre modèle de domaine par un système expert à base de règles. Il est considéré comme difficile d'exprimer toutes les connaissances du domaine à travers un ensemble de règles exhaustif (Section 2.2.1). Cependant, en faisant le choix d'utiliser Mascaret, nous proposons aux experts métier, un langage basé sur l'ontologie qu'ils peuvent utiliser eux-mêmes pour fournir au système les connaissances du domaine. Mascaret couvre tous les aspects de la représentation sémantique de l'environnement virtuel : l'ontologie du domaine, la structure de l'environnement, le comportement des entités, et à la fois les interactions et les activités de l'utilisateur et des agents. C'est l'ensemble de ces aspects qui représente le modèle du domaine.

Dans Mascaret, la pédagogie est considérée comme un modèle du domaine spécifique. Dans Mascaret, le langage (UML) est utilisé pour décrire à la fois le modèle du domaine et le modèle pédagogique. Le modèle du tuteur (également appelé le modèle pédagogique) est représenté dans Mascaret par un scénario pédagogique. Koper considère qu'un scénario pédagogique est composé de cinq éléments principaux : objectifs

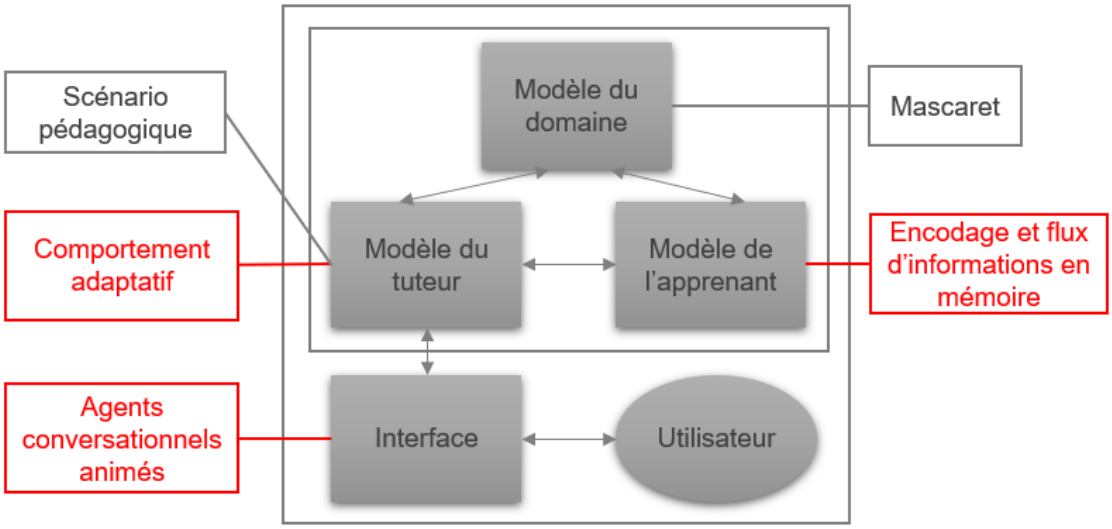


FIGURE 3.1 – Schéma des caractéristiques de notre tuteur dans le contexte des quatre composantes d'un système tutoriel intelligent.

pédagogiques, pré-requis pédagogiques, activités pédagogiques, organisations pédagogiques et environnements pédagogiques (KOPER et VAN ES 2004). Dans MASCARET les scénarios pédagogiques sont implémentés par une chaîne d'actions et d'activités. Ces actions et activités peuvent être soit des actions pédagogiques, comme expliquer une ressource, ou des actions de domaine, comme manipuler un objet. Ces scénarios permettent alors de définir les activités qui doivent être réalisées par le tuteur et l'apprenant, le séquençage de ces activités ainsi que les objectifs pédagogiques à atteindre. Cependant, ces scénarios restent généraux. Ils peuvent être efficaces au début de l'apprentissage quelque soit l'apprenant, mais pas forcément dans les répétitions suivantes. En partant du principe que chaque apprenant peut évoluer différemment, il est important d'adapter l'exécution de ces scénarios pédagogiques en fonction de l'évolution de l'apprenant.

Pour cela, nous proposons dans la Section 3.3 un comportement de tuteur adaptatif, capable de s'adapter en fonction de l'évolution des apprenants, c'est-à-dire d'adapter l'exécution du scénario pédagogique et de personnaliser ces interventions en choisissant des actions pédagogiques tel que modifier l'environnement, poser des questions, répondre à des questions, etc.

Le choix des actions pédagogiques réalisées par le tuteur est basé, essentiellement, sur une représentation du modèle de l'apprenant. Dans notre travail, la représentation du modèle de l'apprenant (Section 3.2) est formalisée autour des travaux en psychologie cognitive définissant la manière dont les informations données par un tuteur et l'environnement virtuel sont encodées et stockées dans la mémoire de l'apprenant. Le modèle informatique de l'encodage de ces informations constitue l'apport majeur de notre travail.

Afin de permettre à l'apprenant d'interagir avec le système d'une manière naturelle, en se rapprochant le plus possible du type de communication et d'interaction multimodale auquel les utilisateurs humains sont habitués, nous choisissons de représenter notre interface par un agent conversationnel animé (ACA) (Section 3.1). Par conséquent, en prenant en compte les *feedbacks* de l'apprenant et des inférences concernant le contenu de ses mémoires, notre tuteur est capable non seulement de parler avec l'apprenant mais aussi de modifier et d'adapter son comportement à celui de l'apprenant.

Sommaire

3.1 Modèle de l'interface

Le modèle de l'interface réalise et analyse les interactions entre l'apprenant et le système. Plus précisément dans notre travail, son rôle est de gérer la communication et les interactions entre l'apprenant et le tuteur dans l'environnement virtuel. Afin de pouvoir maintenir l'engagement de l'apprenant et d'obtenir un comportement réaliste de l'apprenant, nous considérons qu'il est important de maintenir une communication naturelle entre le tuteur et l'apprenant, afin de s'approcher le plus possible du type de communication auquel les utilisateurs humains sont habitués. Ceci permet d'une part de maintenir l'engagement de l'apprenant et d'autre part de favoriser le comportement naturel de l'apprenant. Nous proposons donc, comme dans les projets présentés dans la Section 2.2.3, de représenter le modèle de l'interface par un agent virtuel incarné. En effet, la présence d'agents virtuels interactifs également appelés Agents Conversationnels Animés (ACA), en prenant le rôle de tuteurs (Johnson, Rizzo et al. 2004), semble avoir des effets positifs sur l'engagement de l'apprenant (Rowe et al. 2007) et sur l'efficacité de l'enseignement (Kokane et al. 2014). De plus, des expériences, comme celle présentée dans (Lester et al. 1997), montrent que l'utilisation des ACA peut motiver l'utilisateur à accomplir les tâches requises.

L'architecture globale de notre modèle de l'interface est représentée sur la Figure 3.2. Dans notre modèle, l'ACA possède une base de connaissances sur le modèle métier (le modèle du domaine et les scénarios pédagogiques). L'ACA possède également des comportements dont le comportement adaptatif du tuteur (Section 3.3.1), le comportement de suivi de procédure (pour la réalisation des scénarios pédagogiques) et le comportement de communication. Lors de la réalisation de ces comportements, l'ACA réalise des actions dans l'environnement virtuel, qui sont à destination de l'utilisateur humain (apprenant). Dans le cadre des systèmes tutoriels intelligents, l'apport de notre idée est que ces actions sont réalisées d'une manière naturelle par un ACA.

Le rôle principal de notre modèle de l'interface est de reconnaître les actions réalisées par l'apprenant et de réaliser l'action choisie par le comportement du tuteur et/ou par le scénario pédagogique. Pour faciliter la reconnaissance de ces actions, nous formalisons, dans le modèle de l'interface, les actions de base qu'un agent incarné est capable d'effectuer et de reconnaître en se basant sur les Primitives Comportementales Virtuelles (PCV) (Richir et al. 2015). Nous définissons donc trois types d'actions primitives :

- communiquer : communication verbale (donner une information, répondre à une question, écouter) et non-verbale (expression faciale, geste, regard),
- agir sur l'environnement : actions qui peuvent modifier l'environnement (manipuler ou mettre en évidence un objet),
- naviguer : observer, se déplacer.

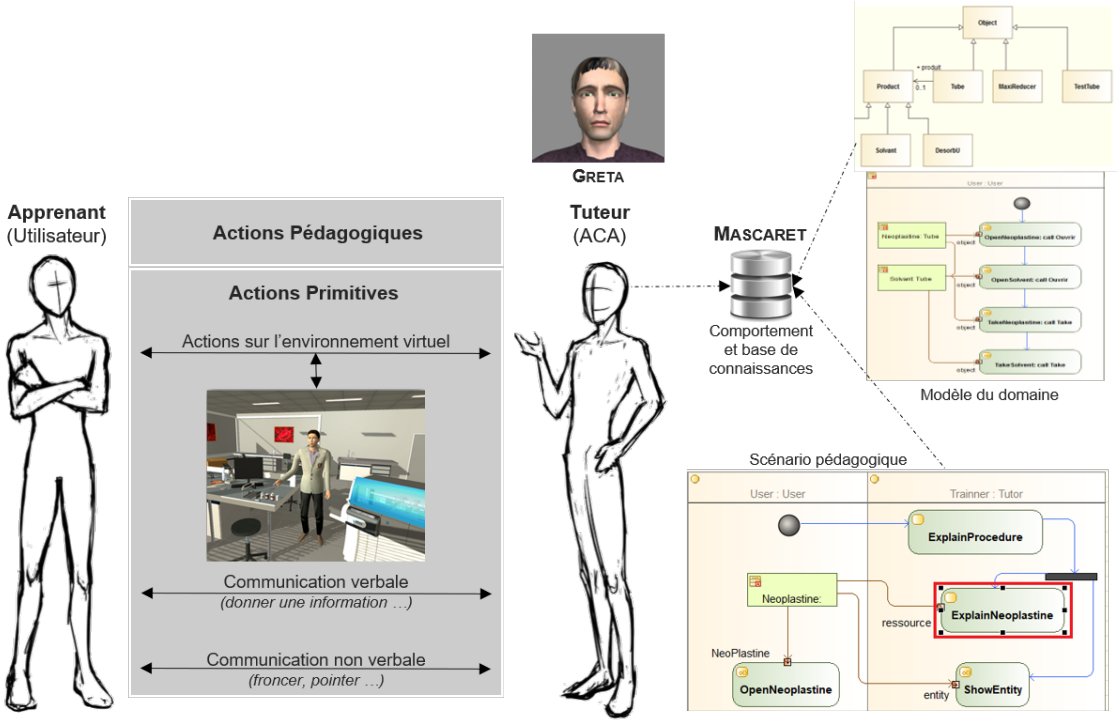


FIGURE 3.2 – Représentation du modèle de l'interface.

Ces actions primitives sont utilisées pour implémenter les actions pédagogiques et les actions métiers. Dans le cadre ce travail, nous ne traitons pas la problématique de reconnaissance des actions de l'utilisateur dans l'environnement virtuel (par l'analyse de geste, ou de l'état de l'environnement après une action). Ainsi, lors de la simulation, l'utilisateur sélectionne l'objet qu'il veut manipuler, un menu est alors affiché avec l'ensemble des actions réalisables sur cet objet. Ceci est réalisable grâce à la connaissance du modèle du domaine (Figure 3.3).

Dans ce travail, nous nous intéressons principalement aux actions de type « communiquer ». Nous distinguons deux types d'actions de communication : verbale et non-verbale. Pour la reconnaissance et la réalisation des actions de communication verbales, nous nous appuyons sur les travaux de Querrec et ses collègues (QUERREC et al. 2018). Dans ce travail, ces actions véhiculent un contenu dont la sémantique est représentée en s'appuyant sur le méta-modèle MASCARET. Nous nous appuyons sur ce contenu pour identifier les informations transférées à l'apprenant. Nous considérons ces actions de communication comme des instructions données à l'apprenant et nous montrons dans la Section 3.2 comment nous en tenons compte.

3.1.1 Reconnaissance des actions de communication non-verbale

La communication verbale et/ou non verbale joue un rôle important dans l'expression du comportement humain, qui constitue une partie importante de l'interaction sociale et du comportement humain. Plus particulièrement, les expressions faciales sont une partie

FIGURE 3.3 – Menu des actions sur un objet, généré à partir du modèle du domaine.

essentielle de la communication non verbale. Ekman et Friesen distinguent six expressions universelles : la tristesse, la joie, le dégoût, la colère, la peur et la surprise (EKMAN et FRIESEN 1976). De nombreux chercheurs se sont inspirés de ce travail pour analyser et reconnaître les expressions faciales, en identifiant généralement les émotions à travers le codage de marqueurs faciaux, le discours (BUSSO et al. 2004) et les caractéristiques physiologiques (par exemple, la fréquence cardiaque, la résistance de la peau, etc.) (LEON et al. 2007). Différents domaines d'applications ont évoqué l'intérêt de la reconnaissance des émotions à partir des expressions du visage tels que le domaine de l'interaction homme-machine, la réalité virtuelle, la détection de la fatigue ou l'analyse de la satisfaction des clients. Cependant, la reconnaissance automatique des expressions faciales demeure une tâche difficile. Pour la reconnaissance des actions de communication non-verbale, nous nous appuyons sur la technologie Intel® RealSense™. Plusieurs travaux ont examiné l'utilité de cette technologie en tant que complément ou remplacement des techniques et architectures intensives des expressions faciales. Des résultats ont indiqué que la technologie Intel® RealSense™est un outil potentiellement utile, fiable et efficace pour les expressions faciales (PATIL et BAILKE 2016).

Dans notre travail, la caméra Intel® RealSense™a été utilisée pour détecter le visage de l'apprenant (dont la position et l'orientation). La fonctionnalité de suivi et de reconnaissance du visage de cette caméra permet d'identifier la présence de l'utilisateur et ces caractéristiques faciales et prend en charge jusqu'à 78 points de repère (appelés en anglais les *landmarks*), présentés dans la Figure 3.4. Ces *landmarks* sont utilisés pour re-

connaître un ensemble d'expressions faciales de haut niveau (Figure 3.5) et les émotions associées (colère, dégoût, peur, joie, tristesse et surprise, selon Intel®, représentées en Figure 3.6). Ces émotions correspondent à celles identifiées par Ekman et Friesen (Ekman et Friesen 1976).

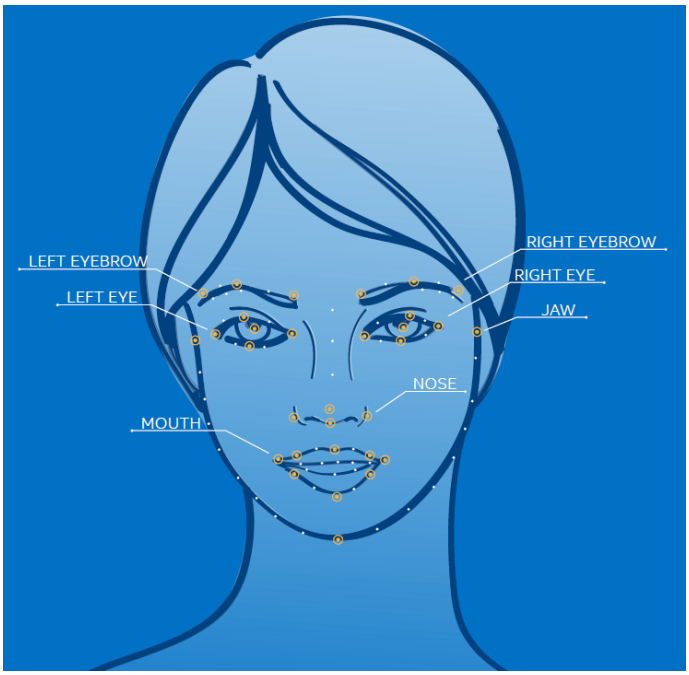


Figure 3.4 – Positions des 78 *landmarks* dans Intel® RealSense™.

Toutefois, même si la technologie Intel® RealSense™offre cette possibilité, nous n'avons pas utilisé les informations sur les émotions dans notre modèle pour les raisons suivantes :

- la notion d'émotion reste encore discutable dans le domaine de la reconnaissance automatique par ordinateur. L'émotion reconnue, est-elle réellement une émotion interne, ressentie ou juste exprimée par l'apprenant ?

- nous n'avons pas jugé pertinent d'utiliser des émotions de types joie, peur ou dégoût dans une situation pédagogique.

Nous avons cependant intégré la reconnaissance d'expression faciale dans notre modèle de l'interface. L'expression reconnue est stockée dans notre modèle de l'interface et peut être utilisée par le comportement adaptatif du tuteur (Section 3.26).

Par exemple, lorsque le tuteur énonce une instruction, et que le système détecte que l'apprenant fronce les sourcils (*Eye-brow lowered* sur la Figure 3.5) le tuteur peut choisir de réaliser une action pédagogique supplémentaire. De plus, le tuteur pourrait également froncer les sourcils pour montrer que l'action que l'apprenant mène actuellement est fausse. Ainsi, le tuteur montre à l'apprenant qu'il commet une erreur et, en même temps,

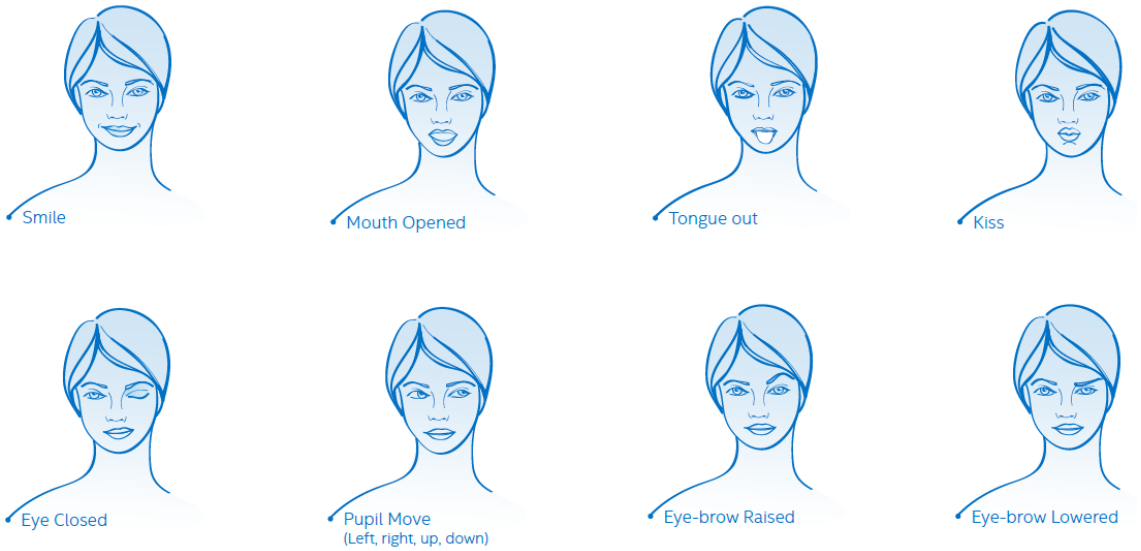


FIGURE 3.5 – Les expressions faciales de haut-niveau selon Intel® RealSense™.

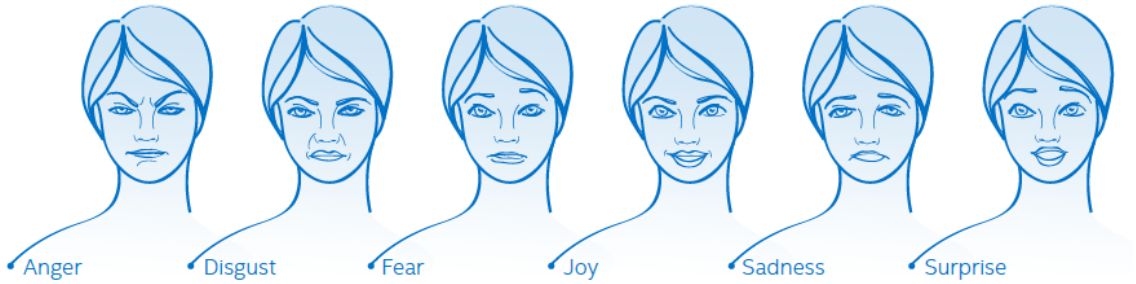


FIGURE 3.6 – les six émotions principales selon Intel® RealSense™.

qu'il est attentif à sa présence et à ses actions. Un autre exemple, si le tuteur remarque que l'apprenant ne regarde pas dans la bonne direction (orientation du visage et *Pupil move* sur la Figure 3.5), par exemple où l'objet d'intérêt est, il pourrait pointer vers l'objet pour guider l'apprenant. Afin de démontrer la faisabilité de ces comportements dans notre modèle, nous montrons en Section 3.3 comment certains d'entre eux ont été implémentés.

3.1.2 Génération des actions de communication verbale et non-verbale

Pour réaliser de manière naturelle les actions de communication verbale et non-verbale, nous avons intégré la plate-forme d'agent conversationnelle animé GRETA (NIEWIADOMSKI et al. 2009). Cette plate-forme fournit plusieurs représentations de personnages virtuels qui peuvent être affichés dans divers périphériques de réalité virtuelle. Ces caractères sont capables de sélectionner et d'effectuer des comportements communicatifs et expressifs multimodaux. La plate-forme GRETA est compatible avec l'architecture standardisée SAIBA [1] (VILHJÁLMSSON et al. 2007). L'architecture SAIBA est une initiative de

1. http://www.mindmakers.org/projects/saiba

recherche dont le principal but est de définir une architecture standard pour la génération de comportements interactifs des agents virtuels animés. Cette architecture divise la génération du comportement de l'agent virtuel en trois modules :

- le module de « planification des intentions » (Intent Planner) : détermine les intentions de communication de l'agent.
- le module de « planification du comportement » (Behavior Planner) : transforme les intentions de l'agent en signaux multimodaux.
- le module de « génération du comportement » (Behavior Realizer) : réalise les signaux multimodaux sur la représentation de l'agent virtuel.

L'intégration générique des plates-formes d'ACA a déjà été réalisée dans MASCARET (NAKHAL 2017). La plate-forme GRETA possède à la fois un module de planification du comportement et un module de génération du comportement, ce qui permet de la connecter à MASCARET aux deux niveaux (Figure 3.7). Pour l'intégration de la plate-forme GRETA dans notre modèle de l'interface, nous avons connecté uniquement son module de planification du comportement. Ce dernier est assez puissant pour sélectionner automatiquement un ensemble de signaux multimodaux. Dans MASCARET, seuls les agents incarnés (Figure 3.7, classe `EmbodiedAgent`) disposent d'un module de planification du comportement (Figure 3.7, classe `BehaviorPlanner`). Afin d'être compatible avec toutes plates-formes SAIBA, MASCARET définit des classes abstraites pour le module de planification du comportement et du module de génération du comportement (Figure 3.7, classe `BehaviorPlanner` et `BehaviorRealizer`). Pour intégrer la plate-forme GRETA nous héritons ces classes abstraites dans deux classes concrètes `GretaBehaviorPlanner` et `GretaBehaviorRealizer`. Le travail principal d'intégration de la plate-forme GRETA se situe dans la méthode `ParseIntention` de la classe `GretaBehaviorPlanner`. Nous détaillons cette méthode dans la suite de cette section.

D'un point de vue technique, nous avons utilisé la version de la plate-forme GRETA qui fonctionne dans le moteur Unity 3D. Cette version est composée de deux parties : une première sous Unity 3D qui s'occupe de la géométrie et des animations des personnages, et une deuxième en java qui s'occupe de la sélection des signaux multimodaux. La deuxième partie s'articule autour de modules qui sont connectés les uns aux autres afin d'échanger des messages. Dans notre modèle de l'interface, nous avons créé notre propre configuration de modules présentée dans la Figure 3.8.

Dans cette configuration, le premier module est le module « `MascaretCommandReceiver` » que nous avons développé spécifiquement pour notre modèle de l'interface. L'objectif de ce module est de faire le lien entre les intentions communicatives générées par la classe `GretaBehaviorPlanner` de MASCARET et le module de planification du comportement de GRETA. Nous expliquons plus loin dans cette section comment ces intentions communicatives sont générées par MASCARET. La communication entre MASCARET et le

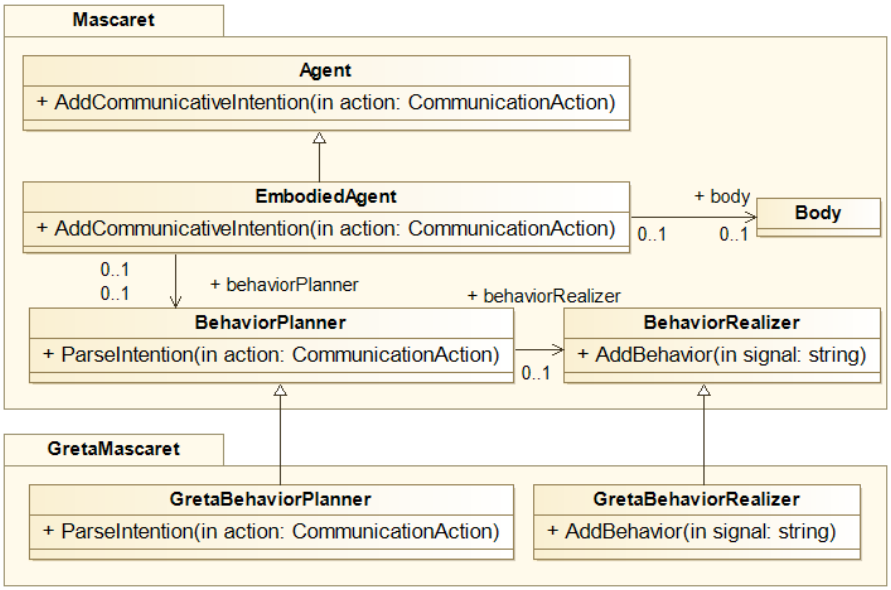


FIGURE 3.7 – Diagramme de classe de l'intégration de la plate-forme GRETA dans MASCARET.

module de planification du comportement se fait par l'échange de fichier FML [2]. Ce format de fichier n'a pas encore été formalisé dans l'architecture SAIBA. Nous utilisons donc le format spécifique de GRETA. Les intentions communicatives sont traitées par le module de planification du comportement et exécutées par le module de génération du comportement. Ce dernier module se connecte aux modules qui gèrent l'animation du personnage, la génération de sa voix, ses animations faciales et labiales. Toutes ces informations sont ensuite transmises au moteur Unity 3D.

La plate-forme GRETA a sa propre représentation du monde qui n'est pas liée au moteur 3D. Toutefois, il est nécessaire que l'environnement simulé en réalité virtuelle dans notre modèle par MASCARET soit cohérent avec l'environnement de GRETA. En effet, si l'ACA doit pointer un objet dans l'environnement virtuel, les animations à jouer sont calculées par GRETA et sont dépendantes de la position de l'ACA et de la position de l'objet à pointer. Or la position de l'ACA et de l'objet sont calculées par les comportements exécutés par MASCARET. Pour des raisons d'optimisation de la simulation, nous avons fait le choix de ne pas mettre à jour continuellement l'environnement de GRETA. Nous réalisons cette mise à jour uniquement lorsque notre modèle de l'interface a besoin de réaliser une action de communication. L'action de communication est interprétée par la méthode `ParseIntention` de la classe `GretaBehaviorPlanner` que nous avons implémenté dans de la plate-forme de réalité virtuelle. À chaque nouvelle action de communication, la position de l'ACA est envoyée par un message au format FML au module `MascaretCommandReceiver`. Le code de génération de ce message est présenté dans l'Algorithme 1.

2. *Function Markup Language*

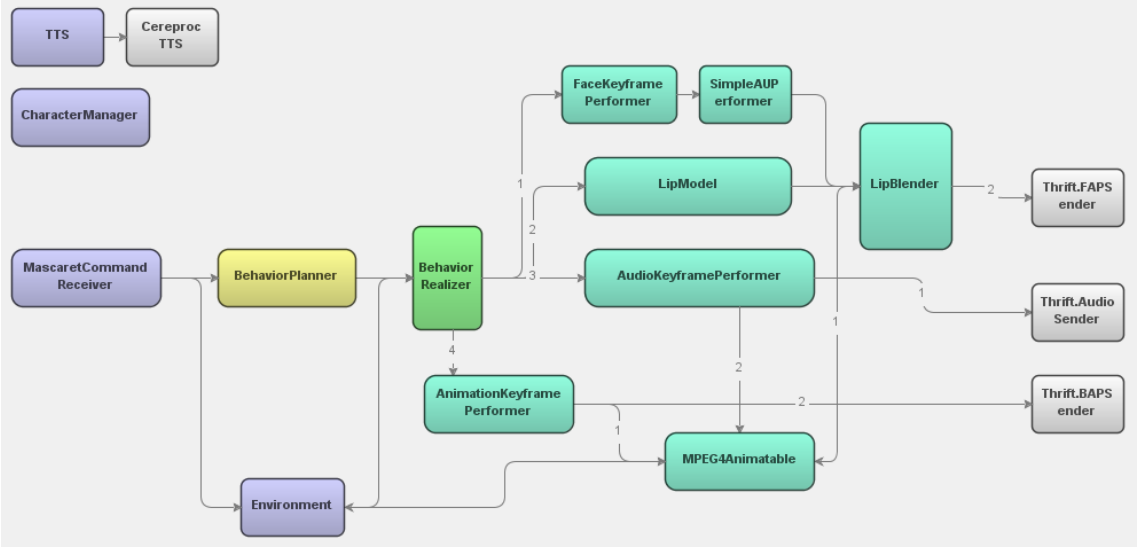


FIGURE 3.8 – Configuration de modules de GRETA utilisée pour notre modèle de l'interface.

Une fois que la position de l'agent est envoyée, la position des objets référencés par l'action de communication est également envoyée par un message au format FML au module `MascaretCommandReceiver`. Le code de génération de ce message est présenté dans l'Algorithme 2.

Après avoir mis à jour l'environnement du côté GRETA, l'intention communicative est générée dans le message au format FML à partir de l'action de communication. La première étape consiste à faire une correspondance entre les performatives exprimées dans l'action de communication et les performatives de GRETA. Nous rappelons que les performatives utilisées dans les actions de communications sont celles de FIPA-ACL (Section 2.1.2.3). La plate-forme GRETA propose des performatives telles que : *inform, greet, disagree, propose, warn,* etc. Si certaines de ces performatives sont communes à FIPA-ACL et GRETA, d'autres n'ont pas de correspondance. Nous avons donc uniquement pris en compte la performative « `INFORM` » qui est utilisée pour transmettre les instructions à l'apprenant par le tuteur. Le contenu de l'action de communication est ensuite placé dans le tag « `speech` » du message FML. Si l'action de communication porte sur des ressources (des objets dans l'environnement), le tag « `world` » est ajouté dans le message pour faire référence à ces ressources et permet à l'ACA de réaliser le pointage.

3.2 Modèle de l'apprenant

Le but du modèle de l'apprenant est d'inférer un certain nombre d'éléments au plan cognitif (tels que le niveau de connaissances, ses objectifs) ainsi que les caractéristiques affectives de l'utilisateur (par exemple, ses préférences, son état émotionnel, etc.) (NKAMBOU, MIZOGUCHI et al. 2010). Ces éléments permettent au système d'identifier certaines caractéristiques de l'apprenant, comme par exemple, sa stratégie d'apprentissage préfé-

Algorithme 1 Envoi de la position de l'agent à GRETA.

```
1 : Real x = agt.position.x;
2 : Real y = agt.position.y;
3 : Real z = agt.position.z;
4 : Real x1 = agt.rotation.x;
5 : Real y1 = agt.rotation.y;
6 : Real z1 = agt.rotation.z;
7 : Real w1 = agt.rotation.w;
8 :
9 : msg += "<animatable>";
10 : msg += "<node name = " + agt.name + ">";
11 : msg += "<position x = " + x + " y = " + y + " z = " + z + "/>";
12 : msg += "<orientation x = " + x1 + " y = " + y1 + " z = " + z1 + " w = "
   + w1 "/>";
13 : msg += "</node>";
14 : msg += "</animatable>";
15 : thrift.Message gretaAgt = new thrift.Message();
16 : gretaAgt.Id = "0";
17 : gretaAgt.String_content = msg;
18 : commandSender.send(gretaAgt);
```

Algorithme 2 Envoi des ressources de l'action à GRETA.

```
1 : if ressource != " " then
2 :    Real x = obj.position.x;
3 :    Real y = obj.position.y;
4 :    Real z = obj.position.z;
5 :    Real x1 = obj.rotation.x;
6 :    Real y1 = obj.rotation.y;
7 :    Real z1 = obj.rotation.z;
8 :    Real w1 = obj.rotation.w;
9 :
10 :    msg += "<environment>";
11 :    msg += "<node name = " + ressource + ">";
12 :    msg += "<position x = " + x + " y = " + y + " z = " + z + "/>";
13 :    msg += "<orientation x = " + x1 + " y = " + y1 + " z = " + z1 + " w =
   " + w1 "/>";
14 :    msg += "</node>";
15 :    msg += "</environment>";
16 :    thrift.Message gretaEnv = new thrift.Message();
17 :    gretaEnv.Id = "2";
18 :    gretaEnv.String_content = msg;
19 :    commandSender.send(gretaEnv);
```

rée. Le modèle de l'apprenant observe le comportement de l'utilisateur, évalue ses performances (par exemple, le temps d'exécution, les erreurs commises), et fournit les informations inférées au modèle du tuteur (représenté dans notre travail par le comportement du tuteur, Section 3.26) afin d'ajuster les *feedbacks* du système à l'apprenant. Puisque nous nous intéressons à l'apprentissage des activités humaines dans les systèmes industriels, les connaissances que notre modèle de l'apprenant doit inférer, sont liées au processus de mémorisation des procédures et des actions à réaliser. Plus précisément, ce qui va être inféré correspond au contenu de la mémoire de l'apprenant (Section 3.2.1). Ceci implique la transformation des stimuli venant du tuteur incarné, représentés dans notre cas par des instructions, et de l'environnement virtuel en connaissances qui peuvent être stockées dans la mémoire.

L'idée originale que nous proposons dans ce travail de recherche, est d'opérationnaliser la théorie de la mémoire humaine d'Atkinson et Shiffrin (Atkinson et Shiffrin 1968) pour concevoir le modèle de l'apprenant. Ceci implique de proposer une formalisation des informations manipulées en mémoire et d'implémenter les processus de traitement de l'information (encodage, stockage, récupération et répétition) qui sont décrites dans la Section 3.2.1. Nous mettons en œuvre cette théorie dans le cadre de l'apprentissage de procédures sur des systèmes industriels, comme expliqué dans la Section 3.2.2.

Notre contribution à cette théorie consiste à expliciter le traitement des informations en s'inspirant de l'architecture cognitive Act-r (Anderson et al. 2004) : (1) en formalisant, de manière générique dans le contexte de l'apprentissage de procédure, les informations de la mémoire de l'utilisateur, et (2) en mettant en œuvre la transformation des stimuli en connaissances et le flux d'informations entre les composantes de la mémoire humaine simulée dans notre modèle.

Dans les sections suivantes, nous décrivons en détail nos contributions. Tout d'abord nous présentons, dans la Section 3.2.1, la théorie de la mémoire humaine d'Atkinson et Shiffrin. Ensuite, dans la Section 3.2.2, nous décrivons notre formalisation de l'encodage des informations dans la mémoire sensorielle, la mémoire de travail et la mémoire à long-terme. Le modèle général du flux de données entre les mémoires lors d'une interaction entre un tuteur et un apprenant, dans une situation pédagogique, est décrit dans la Section 3.2.3. Un tel flux de données concerne les quatre opérations impliquées dans le traitement de l'information de la mémoire (l'encodage, le stockage, la récupération et la répétition).

3.2.1 Théorie de la mémoire d'Atkinson et Shiffrin

Selon le modèle « Modal » (également appelé en anglais « Multi Store Model ») proposé par Atkinson et Shiffrin (Atkinson et Shiffrin 1968), la mémoire humaine est divisée en trois composantes structurelles : la mémoire sensorielle, la mémoire de travail et la mémoire à long terme (Figure 3.9).

La « mémoire sensorielle » est le support de l'étape initiale du traitement de l'information. Le rôle de la mémoire sensorielle est de retenir l'information en provenance de l'environnement juste le temps qu'il faut pour que l'individu puisse sélectionner les éléments

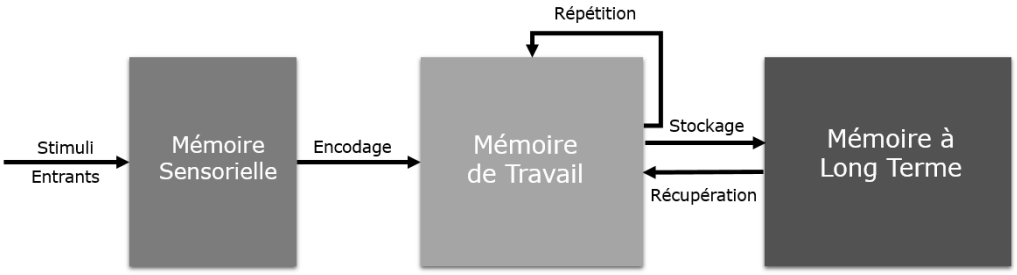


FIGURE 3.9 – Théorie de la mémoire humaine d'Atkinson et Shiffrin.

importants parmi le flot continu d'informations que lui livrent ses différents sens (l'ouïe, la vue, l'odorat, le goût et le toucher). Chaque sens est associé à un sous-système, également appelé « registre d'information sensorielle ». Ainsi, cette dernière est composée d'un registre auditif, d'un registre visuel, d'un registre olfactif, d'un registre gustatif et d'un registre tactile. La durée de vie d'une information dans la mémoire sensorielle est très courte. Elle est de l'ordre d'une demi seconde à deux secondes. Selon le registre, l'information est conservée plus ou moins longtemps. Par exemple, les informations auditives sont maintenues dans le registre auditif de la mémoire sensorielle durant deux secondes environ, tandis que les informations visuelles ne demeurent pas plus d'une demi seconde dans le registre visuel. Dans un premier temps, l'apprenant retient les informations qui lui paraissent importantes, c'est-à-dire uniquement celles qui ont attiré son attention, mais il est capable de traiter à nouveau ces informations si elles sont encore présentes dans le registre d'information sensorielle.

Ces informations pertinentes sont dirigées ensuite vers la « mémoire de travail », dont le rôle principal est de retenir une information, le temps d'effectuer une tâche impliquant cette information. Cette mémoire est utilisée pour le stockage et la manipulation des informations. La quantité d'informations stockées en mémoire de travail est limitée. Elle est estimée à sept (plus ou moins deux) « *chunks* » (MILLER 1956). Un *chunk* est une unité significative d'information (ANDERSON 1996) qui peut correspondre à une lettre ou un chiffre, un groupe de mots, voire une phrase entière. Il résulte d'un processus cognitif consistant à regrouper l'information dans un paquet formant une entité facilement mémorisable en mémoire de travail. Il est possible de regrouper des *chunks* en blocs d'informations. Ces blocs deviennent un seul et unique *chunk*, ce qui permet ainsi d'augmenter la capacité de la mémoire de travail. Par exemple, la mémorisation des 13 lettres S-N-C-F-B-M-W-E-D-F-I-B-M peut être facilitée en regroupant ces lettres en 4 blocs d'informations ayant une signification connue, comme SNCF-BMW-EDF-IBM. De cette façon, la capacité de la mémoire de travail peut être augmentée. Plutôt que de tenter de retenir 13 lettres (une lettre par *chunk*, 13 étant supérieur à 7), il est possible de retenir 4 blocs d'information, représentant des acronymes dans cet exemple, constitués de plusieurs unités. Ce processus de regroupement est appelé « *chunking* ». L'origine des informations transmises à la mémoire de travail provient soit de la mémoire sensorielle soit de la mémoire à long terme.

Elle peut permettre d'analyser une information provenant de la mémoire sensorielle en la confrontant au contenu de la mémoire à long terme lorsqu'il s'agit de reconnaître un visage par exemple.

La « mémoire à long terme » sert à stocker de façon permanente (ou quasi-permanente) les informations pertinentes provenant de la mémoire de travail. Elle a une capacité de stockage illimitée. Dans la mémoire à long terme, Atkinson et Shiffrin distinguent deux autres mémoires. La mémoire déclarative et la mémoire procédurale. La mémoire déclarative contient des connaissances correspondant au savoir général. Il s'agit, selon Gagné (Gagné 1985), de la connaissance de faits, de règles, de lois et de principes, comme par exemple savoir que Washington est la capitale des États-Unis ou que la glace fond à la chaleur. La mémoire procédurale contient des connaissances liées aux savoir-faire, par exemple, savoir faire du vélo, savoir nager, etc.

Le traitement de l'information dans ces mémoires implique quatre opérations de base : l'encodage, le stockage, la récupération et la répétition (Atkinson et Shiffrin 1968).

L'« encodage » est le processus par lequel les informations provenant de l'environnement sont traduites ou codées en une forme qui peut être stockée dans la mémoire de travail. Le niveau d'abstraction de cette forme (représenté par le processus de « *chunking* ») dépend du contenu de la mémoire à long terme. Une fois l'information encodée en mémoire, il faut la stocker et la faire durer dans le temps. Le « stockage » se définit alors par le maintien et la conservation dans le temps des informations apprises. Le rappel ou la « récupération » se réfère au processus qui permet à une information d'être extraite de la mémoire à long terme vers la mémoire de travail. Il s'agit de la capacité de restituer une information préalablement apprise. Par ailleurs, la répétition de la pratique joue un rôle important dans l'acquisition de connaissances (Craik et Lockhart 1972).

3.2.2 Structure des mémoires et formalisation de leurs contenus

Dans cette section nous présentons notre proposition d'implémentation de la structure des mémoires et le formalisme de leurs contenus. Nous proposons une implémentation de la théorie d'Atkinson et Shiffrin dans le contexte de l'apprentissage de procédures de maintenance, d'utilisation et de diagnostic de panne sur des systèmes industriels. Le principe général de notre proposition est présenté dans la Figure 3.10.

Dans cette figure nous présentons un agent autonome jouant le rôle de tuteur (Figure 3.10, partie gauche) en interaction avec un apprenant, dont nous représentons les connaissances selon les trois mémoires : mémoire sensorielle, mémoire de travail et mémoire à long terme (Figure 3.10, partie droite). Le tuteur et l'apprenant interagissent au sein d'un environnement virtuel, qui constitue l'ensemble des stimuli accessibles à l'apprenant. Les informations verbales (auditives) que produit le tuteur contiennent des instructions qui portent sur l'action que l'apprenant doit réaliser (ou sur des concepts se rap-

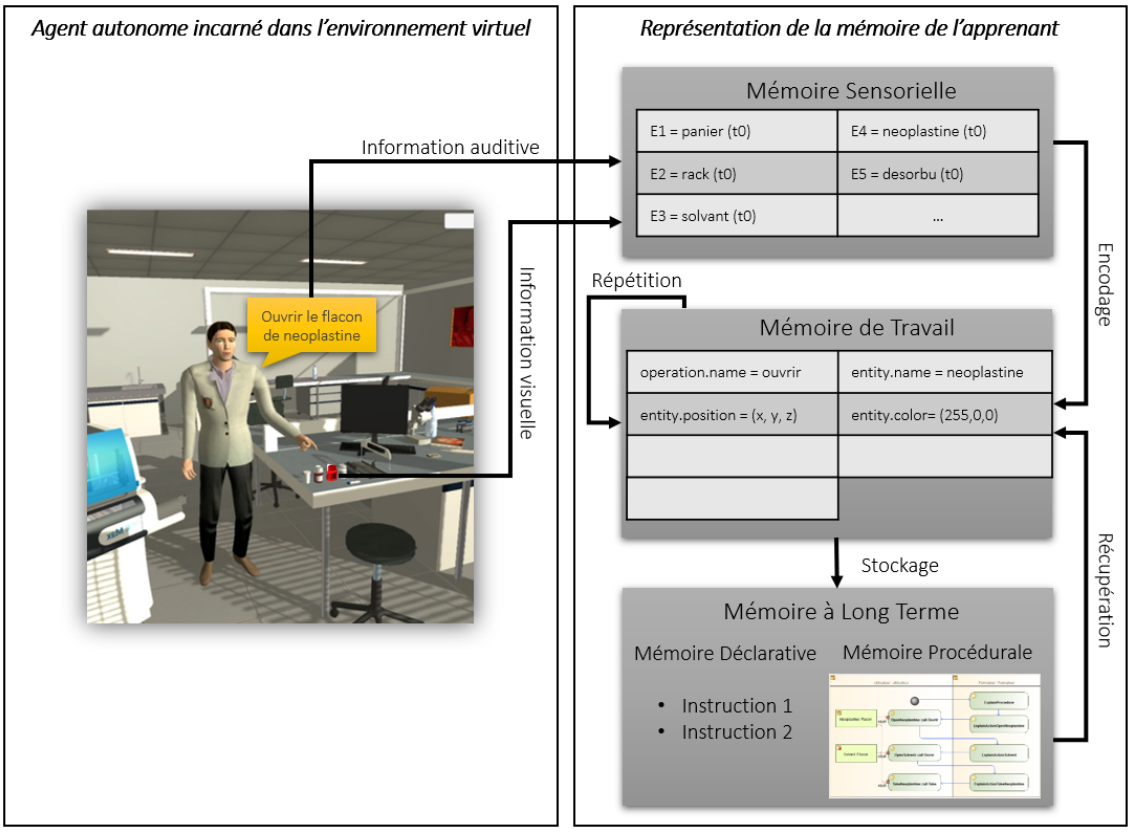


FIGURE 3.10 – Formalisation de l'encodage des instructions dans les mémoires.

portant à l'action). L'environnement virtuel fournit également des informations visuelles pour enrichir les informations verbales. Le problème que nous traitons dans cette section est la définition d'un formalisme d'encodage des informations, issues de ces interactions, dans la mémoire de l'apprenant. Pour cela, nous nous appuyons sur le formalisme de données et le concept d'agent proposés dans MASCARET (Section 2.1.2.3).

Nous présentons dans un premier temps l'architecture globale des mémoires de l'apprenant, puis nous détaillons chacune des mémoires en terme de formalisme d'encodage.

3.2.2.1 Architecture globale

Dans MASCARET, toute entité qui agit sur l'environnement est considérée comme un « agent ». De ce fait, un utilisateur humain, qui dans notre contexte joue le rôle d'apprenant, est considéré comme un agent. Dans MASCARET, tout agent a une base de connaissances qui est représentée en utilisant le principe d'environnement (Figure 3.11, classe `Environment`), représentant l'ensemble des instances et des objets 3D. L'environnement est défini selon un modèle (Figure 3.11, classe `Model`), représentant l'ensemble des classes du métier considéré. Chaque agent peut avoir sa propre instanciation de la classe `Environment` et `Model`. Cette base de connaissances porte sur l'ensemble des concepts, des actions et des procédures que doit acquérir l'apprenant. Nous considérons donc la base de connaissances des agents comme leur mémoire à long terme.

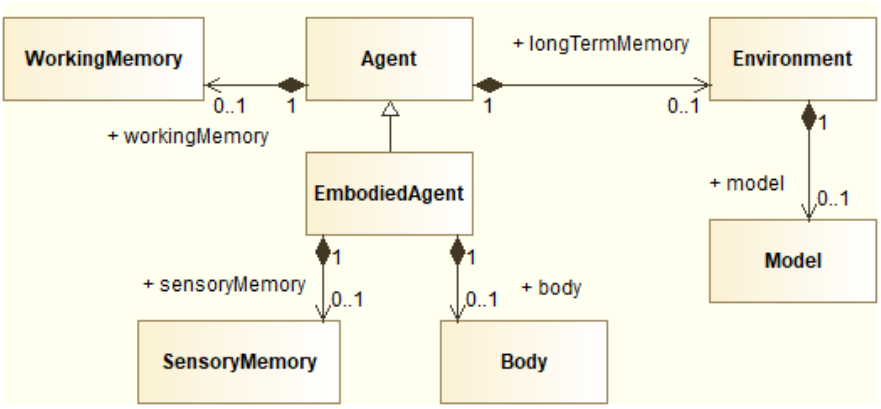


FIGURE 3.11 – Architecture globale du modèle de mémoire.

Seuls les agents incarnés peuvent percevoir l'environnement. Ces agents sont représentés par la classe `EmbodiedAgent` et référencent une mémoire sensorielle (Figure 3.11, classe `SensoryMemory`). Même si l'apprenant peut ne pas être représenté géométriquement dans l'environnement virtuel (pas d'instanciation de la classe `Body`), il est tout de même considéré comme incarné dans l'environnement virtuel, car certains de ses sens sont stimulés par l'environnement virtuel et ses actions ont un impact dans ce dernier.

Les agents, qu'ils soient incarnés ou non, manipulent des connaissances lors de la réalisation de leurs comportements. Dans notre modèle, la classe `Agent` référence donc une mémoire de travail (Figure 3.11, classe `WorkingMemory`).

Rappelons que tout agent, autonome ou utilisateur humain, est représenté par la classe `Agent` ou `EmbodiedAgent` s'il est incarné. Ceci signifie que le tuteur virtuel dispose également d'une mémoire sensorielle, d'une mémoire de travail et d'une mémoire à long terme. Ce modèle est comparable aux architectures proposées dans ACT-R et SOAR. Mais contrairement à ces architectures cognitives, nous n'utilisons pas ce modèle pour définir le comportement de l'agent autonome, mais pour expliciter les connaissances de l'apprenant, sur lesquelles l'agent autonome peut raisonner.

Nous détaillons dans la suite le formalisme d'encodage de l'information pour chacune des mémoires.

3.2.2.2 Mémoire sensorielle

Dans notre travail, les stimuli entrants provenant de l'environnement virtuel et du tuteur virtuel sont limités à ceux liés à la vision et à l'audition. L'environnement virtuel d'apprentissage et les dispositifs que nous utilisons ne stimulent pas les autres sens. Ainsi, l'apprenant peut voir et manipuler des objets 3D et entendre des instructions sonores prononcées par le tuteur sur les activités à réaliser. Par conséquent, nous encodons des données sur les objets et les activités, représentés respectivement par la notion d'« `Entity` » et d'« `Activity` » de MASCARET.

Une entité est représentée par un nom, des propriétés géométriques (position, orientation et forme) et des propriétés du modèle de domaine (attribut de classe) (Figure 3.12). Indépendamment du niveau de connaissances de l'apprenant, un objet est représenté dans toutes les mémoires comme une instance d'`Entity`.

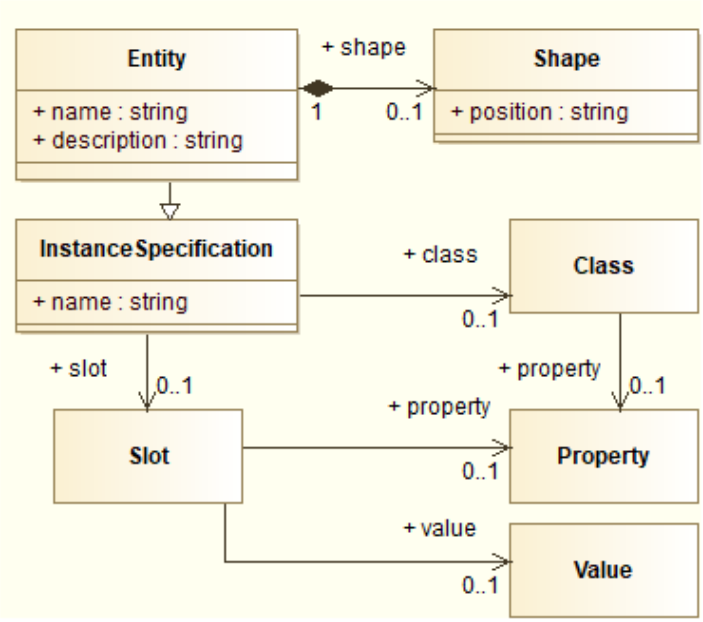


FIGURE 3.12 – Diagramme de classe de la classe `Entity`.

Une activité (`Activity`) est composée de plusieurs sous-activités, de rôles (`Role`) et d'actions (`Action`) (Figure 3.13). Une action (`Action`) est représentée par l'opération (`Operation`) à réaliser et la ressource (`Ressource` jouée par une `Entity`) à manipuler. L'agencement des actions dans l'activité est réalisé par les concepts de `ActionNode` et `Edge`. L'instruction prononcée par le tuteur est représenté par la classe `ActionNode`.

Comme nous l'avons déjà dit, la mémoire sensorielle conserve très brièvement l'information apportée par les différents sens. Pour cette raison, les informations dans le modèle de la mémoire sensorielle ont une durée de vie. Les entités sont stockées dans une liste d'entités horodatées `TimedEntity`, provenant du canal visuel, et les instructions dans une liste de `TimedInstruction` horodatées, provenant du canal auditif. Ce modèle est présenté dans la Figure 3.14.

Prenons l'exemple de la Figure 3.15, dans lequel l'apprenant a dans son champ de vision le flacon de neoplastine et entend simultanément l'instruction « ouvrir le flacon de neoplastine ». Dans ce cas, ces informations sont instanciées par un `TimedEntity` qui référence l'instance d'`Entity` représentant le neoplastine et par un `TimedInstruction` qui référence l'instance d'`ActionNode` qui représente l'action à réaliser. Ces informations portent l'heure à laquelle elles ont été perçues et sont ajoutées à la mémoire sensorielle.

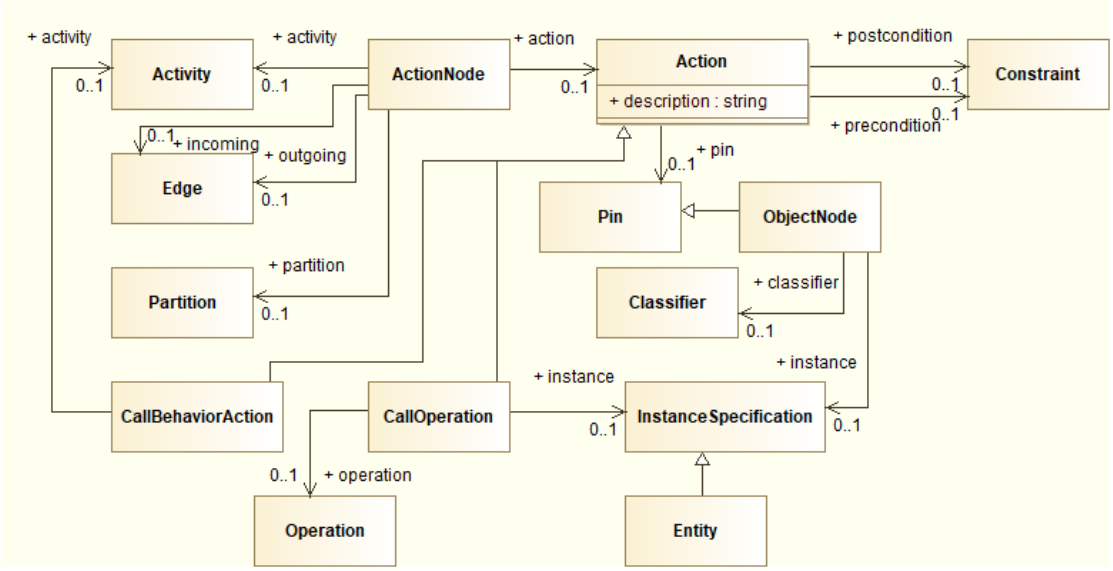


FIGURE 3.13 – Diagramme de classe de la classe `ActionNode`.

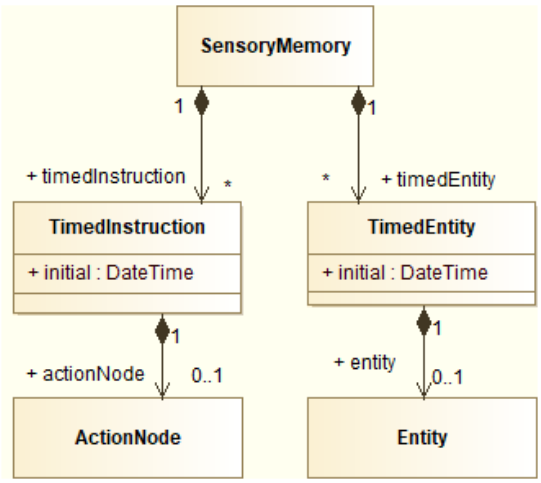


FIGURE 3.14 – Diagramme de classe du modèle de la mémoire sensorielle.

Les informations de la mémoire sensorielle servent à instancier les informations dans la mémoire de travail. Dans la section suivante, nous expliquons la structure de cette mémoire.

3.2.2.3 Mémoire de travail

Comme vu précédemment dans la Section 3.2.1, la mémoire de travail sert à stocker les informations le temps qu'il faut pour réaliser une action. Ces informations sont stockées dans des *chunks*, une information ou un groupe d'informations par *chunk*. La difficulté pour rendre le modèle d'Atkinson et Shiffrin opérable, est de formaliser cette notion d'information. De plus le contenu de la mémoire de travail et le niveau de complexité des informations stockées dépendent du contenu antérieur de cette dernière, du contenu

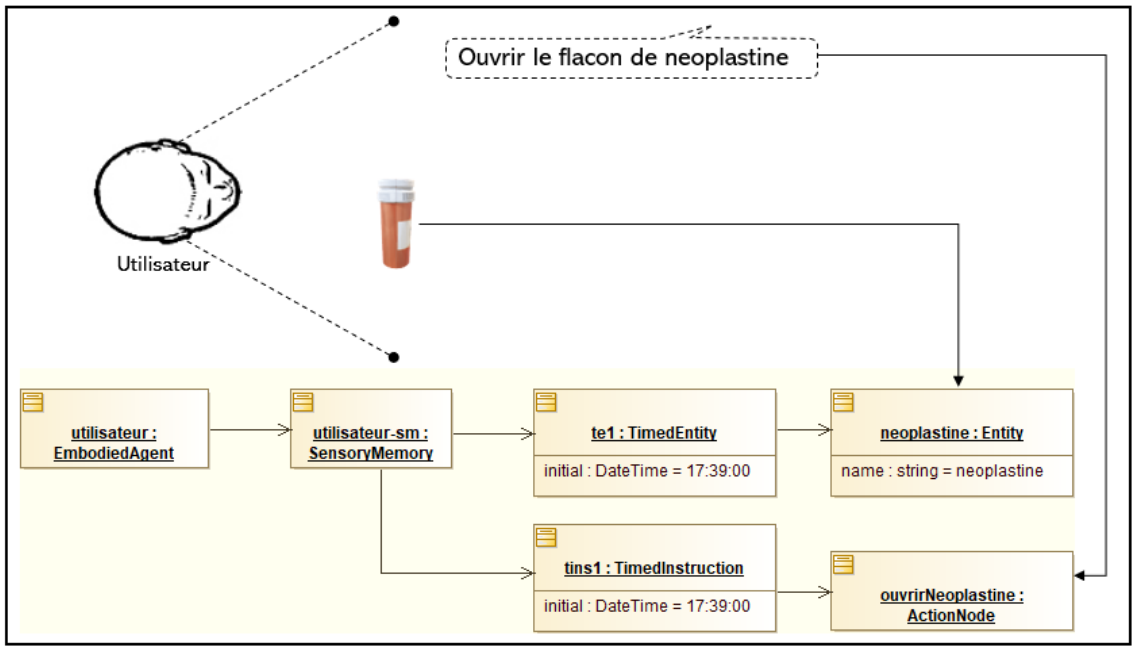


FIGURE 3.15 – Exemple d'instanciation d'une entité et d'une instruction en mémoire sensorielle.

de la mémoire sensorielle et de la mémoire à long terme. Nous proposons dans notre modèle de formaliser la mémoire de travail autour des notions de *chunk* et d'information, que nous présentons par la suite.

Dans notre modèle, la mémoire de travail est composée de *chunks*, chaque *chunk* référençant une information. Ce modèle générique est présenté dans la Figure 3.16.

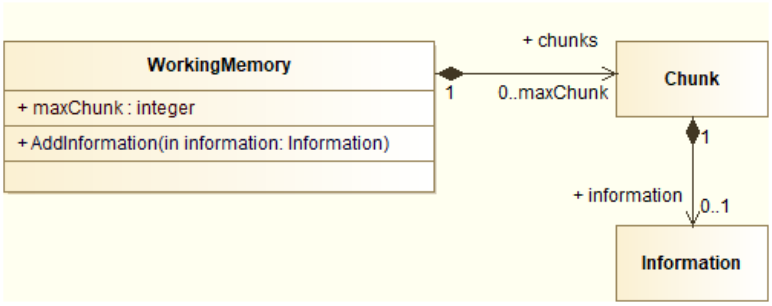


FIGURE 3.16 – Diagramme de classe du modèle de la mémoire de travail.

Selon Miller (MILLER 1956), le nombre de *chunks* dans la mémoire de travail est limité (sept plus ou moins deux). Nous représentons ce nombre par la variable `maxChunk`. Par défaut cette variable vaut 7, mais elle peut être ajustée en fonction du profil de l'apprenant et au cours de la simulation. En effet, chaque utilisateur possède son propre nombre de *chunks*, appelé « empan mnésique », qui pourrait être stocké dans le profil de l'apprenant dans un LMS (Learning Management System) et peut être mesuré par dictée progressive

de chiffres au hasard (CONWAY et al. 2005). Due à un stress ou à la fatigue, ce nombre maximum de *chunk*, peut également varier pour un même apprenant. Notre modèle peut prendre ceci en compte.

Les informations sont ajoutées dans la mémoire de travail par la méthode `AddInformation`. Quand la mémoire de travail est pleine (nombre de *chunk* ajouté est supérieur à `maxChunk`) une information doit être retirée. Nous avons tenu compte des effets de primauté et de récence (GLANZER et CUNITZ 1966), selon lesquels les premières et les dernières informations stockées en mémoire sont plus susceptibles d'être retenues. Ainsi, l'information retirée de la mémoire de travail, pour stocker une nouvelle information, est celle qui se trouve le plus au milieu dans `maxChunk`.

De la même manière que pour la mémoire sensorielle, nous stockons dans la mémoire de travail des informations liées aux objets et aux instructions. Cependant une instruction peut être stockée dans un seul *chunk* ou plusieurs, selon le niveau de connaissance de l'apprenant. Ce choix d'instanciation est expliqué dans la Section 3.2.3 (flux de données). Les types d'informations que nous stockons dans la mémoire de travail sont représentés dans la Figure 3.17. Nous pouvons stocker des informations sur l'entité (`EntityInformation`) et sur une instruction (`ActionNodeInformation`) dans un seul *chunk*. Par ailleurs, dans notre modèle, une instruction peut contenir trois informations :

- l'opération à réaliser `OperationInformation`,
- le but de l'action à réaliser `GoalInformation`,
- les ressources de l'action, représentées par les entités `EntityInformation`.

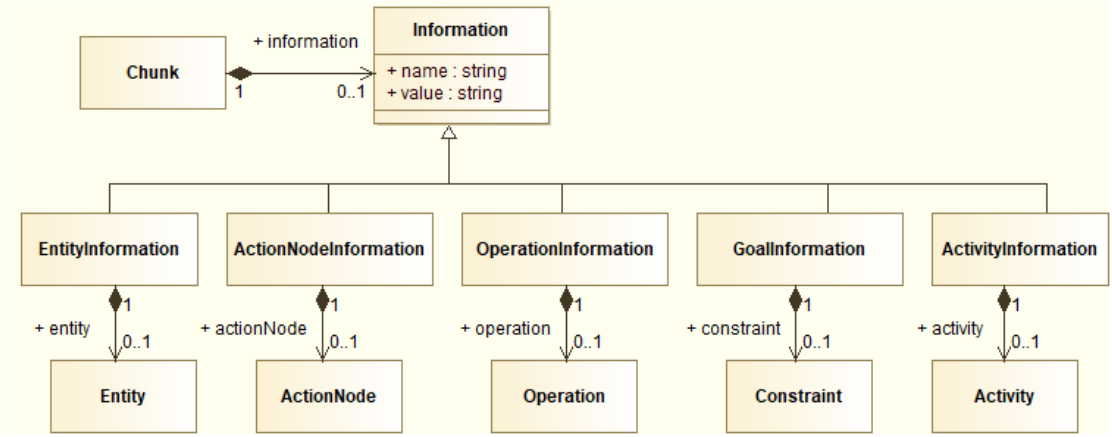


FIGURE 3.17 – Diagramme de classe des types d'informations pouvant être stockés en mémoire de travail.

Selon le niveau de connaissance de l'apprenant, le type et le nombre d'informations qui seront instanciées dans la mémoire de travail peuvent être différents. De plus, pour un type particulier d'information, certaines de ses propriétés peuvent ne pas avoir de valeurs. Ceci nous permet de définir un niveau de complexité de stockage de l'information.

L'évolution du niveau de complexité utilisé pour stocker l'information dans la mémoire de travail au cours de l'apprentissage, correspond au principe de *chuncking*.

Pour chacun des deux concepts, entité et activité, nous avons défini différents niveaux de complexité qui servent à l'encodage des informations dans la mémoire de travail. Ces niveaux de complexité sont également utilisés pour le stockage et la récupération des informations en mémoire à long terme. Pour l'encodage des informations liées aux entités, nous avons défini trois niveaux (Tableau 3.1) et les activités sont définies selon cinq niveaux de complexité (Tableau 3.2).

Tableau 3.1 – Niveaux de complexité des informations liées aux entités.

Niveau de complexité	**Description**
Premier niveau	Une `EntityInformation` est instanciée et référence une `Entity` dont l'attribut `name` peut avoir une valeur, s'il a été donné dans une instruction ou si l'apprenant connaît déjà l'entité.
Deuxième niveau	Une `EntityInformation` est instanciée et référence une `Entity`, comme dans le premier niveau, et possède une référence vers sa `Shape` (représentant la géométrie et la position) est ajoutée.
Troisième niveau	Ce niveau représente l'ajout de connaissances déclaratives : `Class`, `Slot`, `Property`. Ces informations sont instanciées en fonction des connaissances préalables de l'apprenant et des interactions verbales avec le tuteur (Section 3.1).

Tableau 3.2 – Niveaux de complexité des informations liées aux activités.

Niveau de complexité	**Description**
Premier niveau	Une `OperationInformation` est instanciée et référence l'opération à réaliser, ainsi que sa description.
Deuxième niveau	Une `ActionNodeInformation` est instanciée, qui référence l'opération à réaliser, sa description mais également les ressources à manipuler via la notion de `ObjectNode` et `Pin` (Figure 3.13).
Troisième niveau	Une `ActionNodeInformation` est instanciée avec l'ensemble des informations du second niveau, mais y sont également ajoutés ses flux entrants et sortants (`ControlFlow`), c'est à dire l'agencement entre les actions.
Quatrième niveau	Ce niveau est constitué du troisième niveau dans lequel est ajouté le but de l'action (`Constraint`).
Cinquième niveau	Une `ActivityInformation` est instanciée et représente l'activité `Activity` dont dépends l'action à réaliser. Ce niveau est lui même hiérarchique, car l'activité peut être une sous partie d'une activité de plus haut niveau.

Prenons deux exemples d'instanciation d'informations dans la mémoire de travail en fonction du niveau de connaissances de l'apprenant.

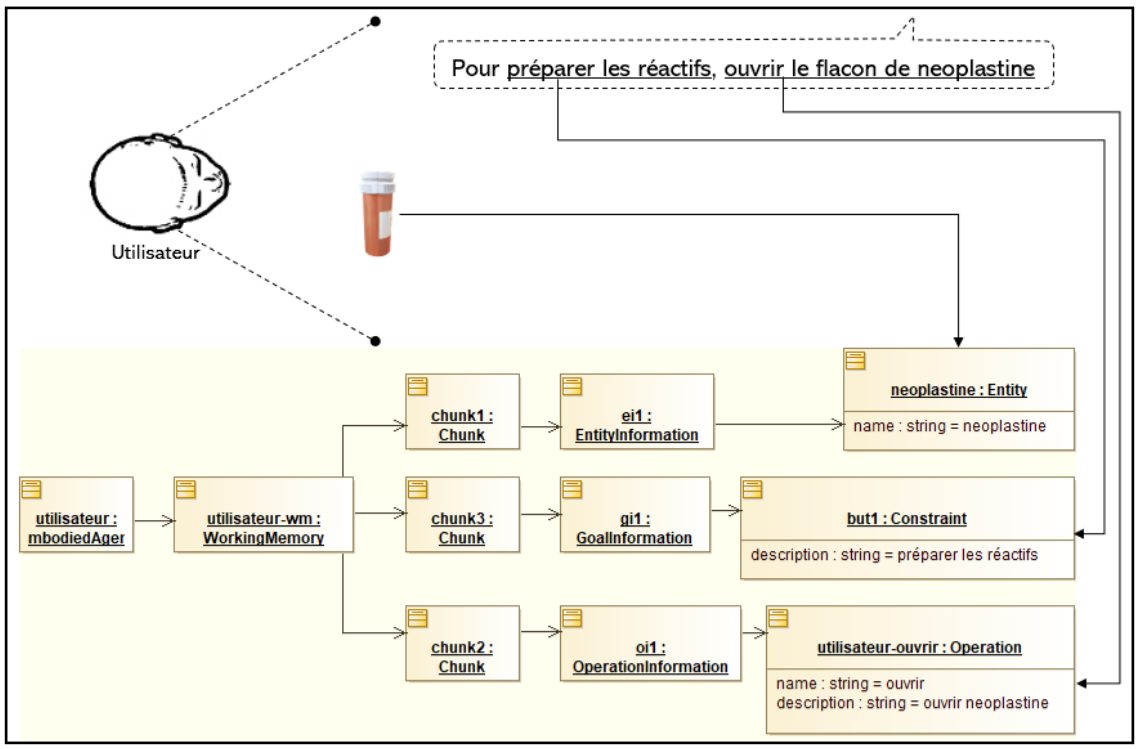


FIGURE 3.18 – Exemple d'instanciation d'une instruction en mémoire de travail sans connaissances préalables.

Dans le premier exemple (Figure 3.18), l'apprenant voit le flacon de neoplastine (et uniquement ce flacon) et entend l'instruction « Pour préparer les réactifs, ouvrir le flacon de neoplastine ». Ces informations sont traitées et encodées pendant un cours laps de temps en mémoire sensorielle puis transférées dans la mémoire de travail. L'apprenant n'a aucune connaissance préalable, donc lorsqu'il entend l'instruction, celle-ci est encodée en mémoire de travail dans trois *chunks* séparés. Cette instruction contient l'opération à réaliser « ouvrir », l'objet sur lequel porte cette opération « neoplastine » et le but de cette opération « préparer les réactifs ». Comme nous considérons que l'apprenant n'a aucune connaissance préalable, le premier niveau de complexité est utilisé pour instancier ces informations. Ainsi, une `EntityInformation` est stockée dans un premier *chunk* (`chunk1`) pour référencer l'entité que voit l'apprenant. Comme nous considérons, dans l'exemple de la Figure 3.18, que l'apprenant ne voit qu'un seul objet et que celui-ci a été nommé dans l'instruction, le nom « neoplastine » est également stocké. L'opération « Ouvrir » est instanciée dans une `OperationInformation`, et est stockée dans un second *chunk* (`chunk2`). L'objet sur lequel porte cette opération est « neoplastine », mais puisque celui-ci a été nommé au moment où l'apprenant voyait cet objet, nous considérons que l'apprenant fait le lien entre ces deux informations qui sont stockées dans le même *chunk*, le `chunk1`. Le fait de présenter simultanément des informations visuelles

et auditives, correspond au principe de contiguïté temporelle (MAYER et MORENO 2002). Enfin, le but de l'opération ayant été énoncé, celui-ci est stocké dans un troisième *chunk* (`chunk3`) référençant une `GoalInformation`, représentée par une contrainte.

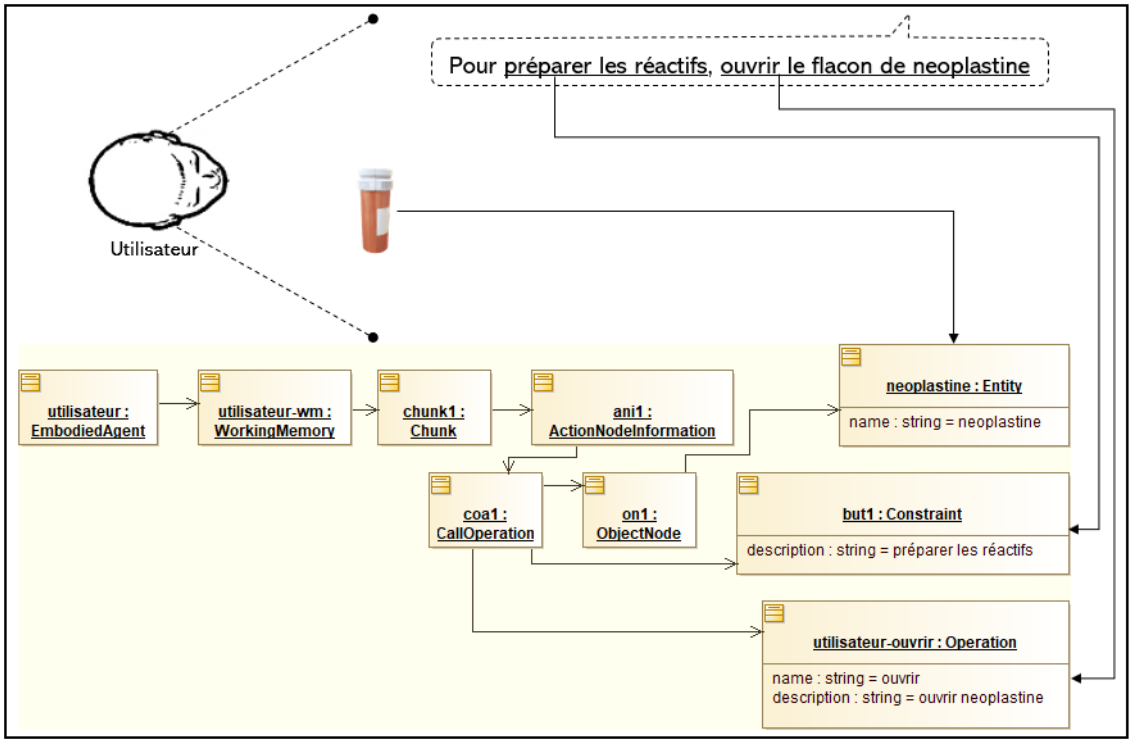


FIGURE 3.19 – Exemple d'instanciation d'une instruction en mémoire de travail avec connaissances préalables.

Dans le second exemple, nous considérons que l'apprenant possède des connaissances préalables sur l'action qu'il doit réaliser, donc lorsqu'il entend l'instruction, elle est encodée en mémoire de travail dans un seul *chunk*. Ainsi lorsque l'instruction « Pour préparer les réactifs, ouvrir le flacon de neoplastine » est prononcée, une `ActionNodeInformation` est instanciée et stockée dans le `chunk1`. L'`ActionNodeInformation` référence un `ActionNode` référençant lui-même l'opération à réaliser (`coa1:CallOperation` et `utilisateur-ouvrir:Operation`), le but de l'opération (`but1:Constraint`) et la ressource de l'opération (`on1:ObjectNode` et `neoplastine:Entity`).

3.2.2.4 Mémoire à long terme

La mémoire à long terme représente les connaissances préalables de l'apprenant à propos de la procédure en début d'exercice et les connaissances qu'il a retenu à la fin d'un exercice. Rappelons que l'expert exprime l'ensemble des connaissances à acquérir grâce à MASCARET, d'une part par la notion de modèle (classe `Model`), contenant l'ensemble de concepts, propriétés et d'activités et d'autre part, par un environnement (classe `Environment`) qui instancie les éléments du modèle. En cours d'apprentissage, l'apprenant acquière des connaissances issues de ce modèle et de cet environnement. Nous considérons donc que la mémoire à long terme de l'apprenant est une sous-partie

du modèle du domaine. Ainsi la mémoire à long terme référence un environnement, lui même défini par un modèle (Figure 3.20).

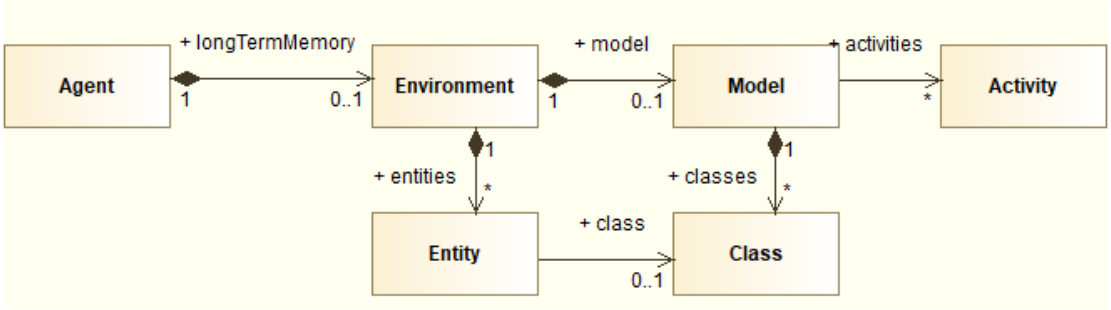


FIGURE 3.20 – Diagramme de classe du modèle de la mémoire à long terme.

Selon la théorie d'Atkinson et Shiffrin, la mémoire à long terme serait divisée en deux sous-parties : les connaissances déclaratives et les connaissances procédurales (ATKINSON et SHIFFRIN 1968).

Les connaissances déclaratives correspondent aux savoirs généraux, comme par exemple des faits. Dans notre modèle, ces connaissances sont représentées par des instances et des valeurs de propriétés dans l'environnement référencé par la mémoire à long terme de l'apprenant. Par exemple, si l'apprenant sait qu'un flacon contient une certaine quantité d'un produit et qu'en particulier, le flacon de neoplastine contient 20 ml de neoplastine l'instanciation présentée en Figure 3.21 est réalisée.

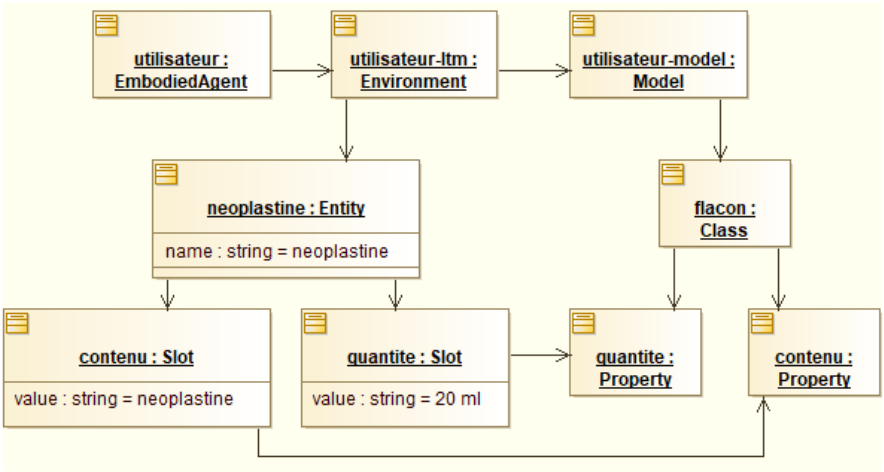


FIGURE 3.21 – Exemple d'instanciation d'une connaissance déclarative en mémoire à long terme.

Les connaissances procédurales concernent la réalisation concrète d'une action et les procédures à suivre. Dans notre modèle, ces connaissances sont représentées par

la construction du diagramme d'activité correspondant à la procédure, au fur et à mesure de la réalisation des actions de l'apprenant. Par exemple, si l'apprenant réalise l'action « ouvrir le flacon de neoplastine » dans le cadre de la procédure « préparer les réactifs » l'instanciation présentée en Figure 3.22 est réalisée. Dans cette figure, un extrait de la mémoire à long terme `utilisateur-model` de l'apprenant est présenté. Cette mémoire référence l'activité en cours de réalisation (préparer les réactifs). Lors de la réalisation de l'action, une nouvelle instance d'`ActionNode` est instanciée (`an1:ActionNode`) et référence l'ensemble des propriétés de l'opération réalisée.

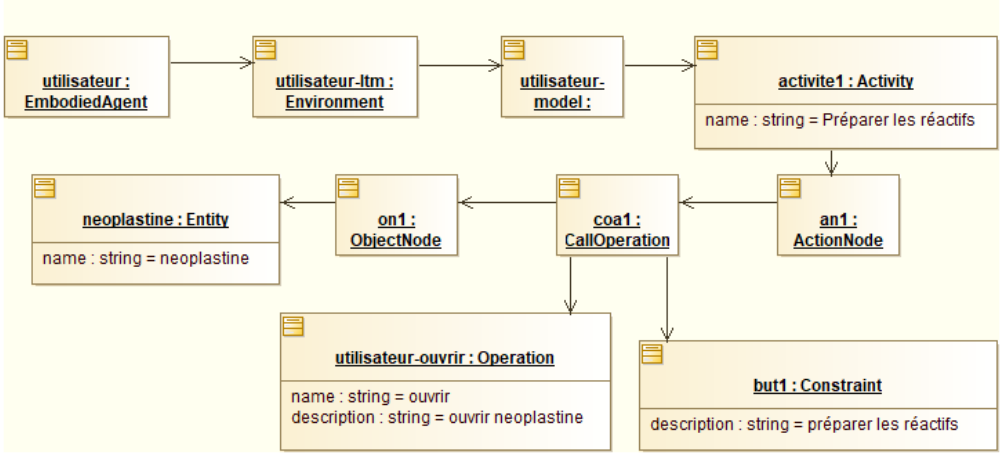


FIGURE 3.22 – Exemple d'instanciation d'une connaissance procédurale en mémoire à long terme.

Cette instanciation correspond à la réalisation d'un extrait de procédure défini par le diagramme d'activité présenté dans la Figure 3.23.

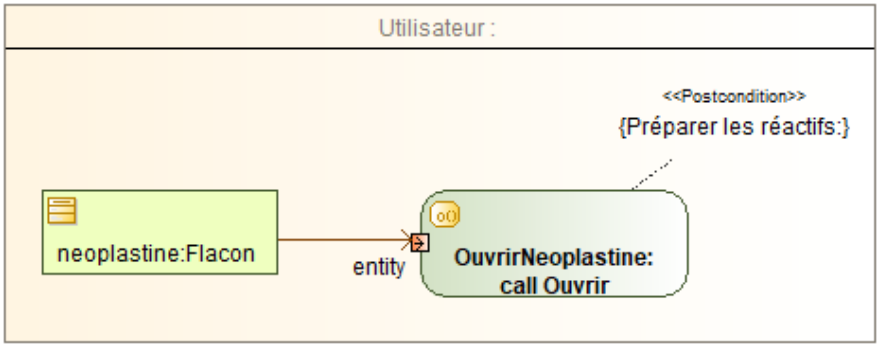


FIGURE 3.23 – Extrait de procédure réalisée par l'utilisateur.

3.2.3 Flux de données entre les mémoires

Dans la section précédente, Section 3.2.2, nous avons présenté notre formalisme de structuration et d'instanciation des informations dans les trois mémoires. Dans cette

section nous décrivons le flux de ces informations entre ces trois mémoires en trois étapes (Figure 3.24).

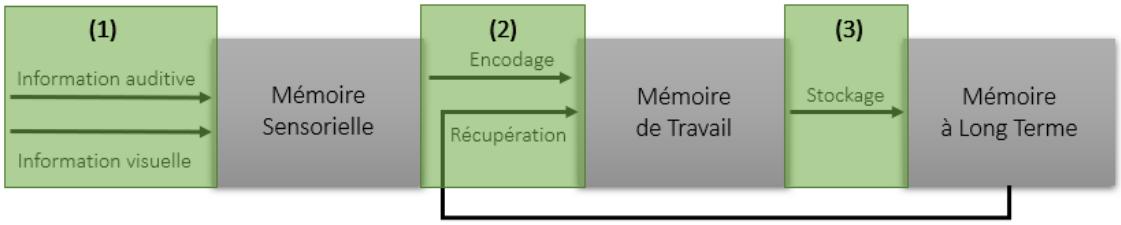


FIGURE 3.24 – Flux d'informations et d'opérations entre les trois mémoires.

La première étape consiste à sélectionner les informations pertinentes provenant du tuteur et de l'environnement parmi les informations brièvement stockées en mémoire sensorielle. Dans la second étape ces informations sont encodées dans la mémoire de travail en fonction des informations récupérées de la mémoire à long terme. Enfin, dans la troisième étape, les informations de la mémoire de travail sont stockées dans la mémoire à long terme. Nous détaillons par la suite chacune de ces trois étapes.

3.2.3.1 Étape 1 : la sélection

La première opération impliquée dans le flux de données est la sélection des informations pertinentes provenant de l'environnement virtuel et du tuteur. Comme nous l'avons expliqué en Section 3.2.1, les stimuli entrants provenant de l'environnement et du tuteur sont visuels (ensemble d'objets présents dans le champ de vision de l'apprenant) et auditifs (instructions prononcées par le tuteur). La représentation formelle de ces informations a été décrite dans la Section 3.2.2.2. Nous présentons dans cette partie la manière dont seules les informations pertinentes sont sélectionnées.

L'apprenant possède un champ de vision, qui lui permet de percevoir l'ensemble des objets présents dans l'environnement dans lequel il se déplace et interagit. Ce processus de perception est permanent et continu. En conséquence, il est traduit en un comportement (`AgentBehavior` de MASCARET) appelé comportement d'observation : `ObservationBehavior`. Ce comportement s'appuie sur la plate-forme de réalité virtuelle (`VRPlatform`) utilisée, pour récupérer les objets présents dans le champ de vision de l'apprenant à travers la fonction `getObservableEntities` (Algorithme 3). La scène pouvant être complexe, un grand nombre d'objet pourrait être pris en compte. Nous souhaitons donc sélectionner parmi ces objets ceux qui sont pertinents et vont être stockés de façon privilégiée en mémoire sensorielle. Nous considérons que l'apprenant fait automatiquement la différence entre les objets du décor et les objets interactifs appartenant au domaine métier en cours d'apprentissage. Donc parmi tous les objets présents dans son champ de vision, seuls les objets qui ont des géométries (`Shape`) d'entités du domaine sont conservés.

Parmi toutes ces entités, certaines peuvent être saillantes parce qu'elles sont mises en évidence dans l'environnement virtuel. Les entités mises en saillance sont alors ajoutées

Algorithme 3 Sélection des entités dans le canal visuel.

```
1 : List<Entity> entities = Environment.getEntities();
2 : Integer iEntity;
3 : List<Entity> result;
4 : for iEntity = 0 .. entities.Length do
5 :    Entity e = entities[iEntity];
6 :    Real angleE = Vector3.Angle(host, e);
7 :    if (angleE < host.angleVision) then
8 :       host.LookAt (e);
9 :       Entity hit =
10:                Raycast(host, Vector3.Front, host.depthVision);
11 :      if hit == e then
12 :         result.Add(e);
```

à la mémoire sensorielle (Algorithme 4).

Algorithme 4 Ajout des entités pertinentes dans le canal visuel.

```
1 : List<Entity> observableEntities = VRPlatform.getObservableEntities();
2 : Integer iEntity;
3 : Entity e;
4 : for iEntity = 0 .. observableEntities.Length do
5 :    e = observableEntities[iEntity];
6 :    agt.Sensorymemory.addVisualInformation(e);
7 :    if e.isHighLighted then
8 :       agt.Sensorymemory.putInEvidence(e);
```

Les entités qui sont ajoutées à la mémoire sensorielle sont horodatées, ceci signifie que si elles ne sont plus dans le champ de vision, elles ne sont plus actives dans la mémoire sensorielle. Elles seront donc retirées automatiquement de la mémoire sensorielle au bout d'un certain temps.

Nous considérons que toute instruction sonore reçue par l'apprenant est une information pertinente. Les instructions sonores sont entendues en langage naturel par l'apprenant. Cependant, elles sont générées grâce aux actions de communication qui portent sur un `ActionNode` (Section 3.1). Pour sélectionner les informations pertinentes portées par l'instruction, nous n'analysons pas ce que l'apprenant a entendu, mais nous utilisons les informations de l'`ActionNode`. Dans MASCARET, l'instruction est transférée entre le tuteur et l'apprenant *via* les actions de communication qui créent un message sous forme FIPA. Le contenu de ce message est exprimé en FIPA-SL et référence l'`ActionNode`. Le problème est d'identifier les propriétés de l'`ActionNode` qui ont été énoncées. Par exemple, est-ce que l'instruction porte sur le but de l'action à réaliser et/ou sur les objets à manipuler ? Nous avons modifié le contenu de ces messages en FIPA-SL pour y ajouter ces propriétés. Dans le message en FIPA-SL nous ajoutons des paramètres, pour spécifier si l'instruction porte sur l'opération à réaliser, le but, et/ou les ressources à manipuler. Ce principe est illustré dans l'exemple 3.2.1 :

Exemple 3.2.1 Traduction de langage naturel vers FIPA-SL.
« Pour préparer les réactifs, ouvrir le flacon de neoplastine »
`((action utilisateur (NEXT :nextAction ouvrirNeoplastine :operation 1 :ressource 1 :goal 1)))`

Pour tenir compte de ces propriétés dans la mémoire sensorielle, nous modifions donc le modèle présenté dans la Section 3.2.2.2 par le diagramme de classe de la Figure 3.25.

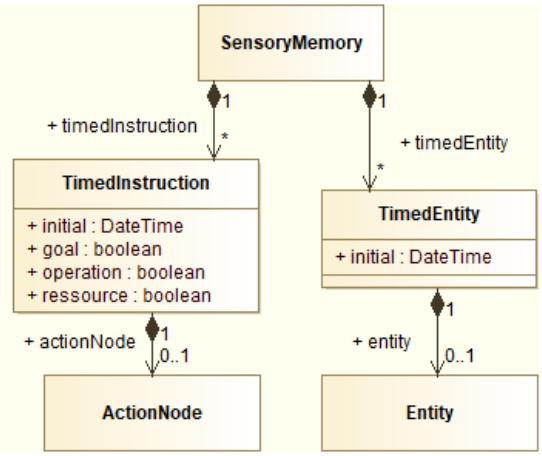


FIGURE 3.25 – Diagramme de classe modifié du modèle de la mémoire sensorielle.

3.2.3.2 Étape 2 : l'encodage et la récupération

Après avoir sélectionné les informations pertinentes, elles sont ensuite codées en une forme qui peut être stockée dans la mémoire de travail. Les informations encodées dans la mémoire de travail et leur niveau de complexité dépend non seulement du contenu de la mémoire sensorielle de l'apprenant, mais également de ses connaissances préalables (informations déjà stockées dans sa mémoire à long terme).

Avant d'encoder une information dans la mémoire de travail, nous vérifions qu'elle existe déjà dans la mémoire de travail. Si c'est le cas, alors rien de nouveau n'est instancié. Sinon, si elle existe dans la mémoire à long terme, alors nous instancions l'information dans la mémoire de travail en utilisant le même niveau de complexité utilisé dans la mémoire à long terme. Pour rappel, les types d'informations instanciables en mémoire de travail sont présentés dans la Section 3.2.2.3. Dans les algorithmes suivants, nous détaillons l'origine des informations encodées en mémoire de travail et comment le type d'informations est défini. Nous définissons deux étapes dans l'algorithme d'encodage dans la mémoire de travail. La première consiste à encoder des informations liées aux entités et la seconde liées aux instructions.

Nous décrivons dans l'Algorithme 5 la manière d'encoder les entités dans la mémoire de travail. Dans un premier temps, si aucune entité n'est mise en évidence dans la mémoire sensorielle, alors nous vérifions si des entités présentes dans la mémoire senso-

Algorithme 5 Encodage des entités en mémoire de travail.

```
1 : Integer iEntity;
2 : Integer tEntities = timedEntities.Length;
3 : if !learner.Sensorymemory.hasEntityHighlighted() then
4 :    for iEntity = 0 .. tEntities do
5 :       Entity entity = timedEntities[iEntity];
6 :       if learner.Longtermmemory.hasEntity(entity.name) then
7 :          if !learner.Workingmemory.hasEntity(entity.name) then
8 :             EntityInformation info = new EntityInformation();
9 :             info.entity = entity;
10 :            learner.Workingmemory.addInformation(info);
11 : else
12 :    for iEntity = 0 .. learner.Sensorymemory.timedEntities.Length do
13 :       Entity entity = learner.Sensorymemory.timedEntities[iEntity];
14 :       if entity.isHighlighted then
15 :          if !learner.Longtermmemory.hasEntity(entity.name) then
16 :             if !learner.Workingmemory.hasEntity(entity.name) then
17 :                Entity newEntity = new Entity(entity);
18 :                newEntity.ActiveShape = null;
19 :
20 :                if learner.Sensorymemory.hasInstWithName(entity) then
21 :                   newEntity.name = entity.name;
22 :
23 :                EntityInformation info = new EntityInformation();
24 :                info.entity = newEntity;
25 :                learner.Workingmemory.addInformation(info);
26 :
27 :                ShapeInformation sInfo = new ShapeInformation();
28 :                sInfo.Shape = entity.ActiveShape;
29 :                learner.Workingmemory.addInformation(sInfo);
30 :          else
31 :             Entity newEntity =
32 :                  learner.Longtermmemory.getEntity(entity);
33 :             EntityInformation info = new EntityInformation();
34 :             info.entity = newEntity;
35 :             learner.Workingmemory.addInformation(info);
36 :
37 :             if newEntity.ActiveShape == null then
38 :                ShapeInformation sInfo = new ShapeInformation();
39 :                sInfo.Shape = entity.ActiveShape;
40 :                learner.Workingmemory.addInformation(sInfo);
```

rielle sont déjà connues par l'apprenant (présentes dans sa mémoire à long terme). Si c'est le cas, elles sont alors ajoutées dans la mémoire de travail (Algorithme 5, lignes 1 à 9). Si au moins une entité est mise en évidence dans la mémoire sensorielle, alors pour chaque entité mise en évidence, nous vérifions que l'apprenant connaît déjà cette entité. S'il ne la connaît pas, nous l'ajoutons dans la mémoire de travail selon le premier niveau de complexité. Dans ce cas, deux *chunks* sont ajoutés à la mémoire de travail, un contenant l'entité et un autre sa géométrie (Algorithme 5, lignes 22 à 28). De plus, si à cet instant il existe dans la mémoire sensorielle, une instruction portant sur cette entité et la nommant, alors le nom de cette entité est ajouté dans le *chunk* contenant l'entité (Algorithme 5, lignes 19 et 20).

Si cette entité est déjà présente dans la mémoire à long terme de l'apprenant, on instancie un ou deux *chunks* en fonction du niveau de complexité déjà stocké en mémoire à long terme (Algorithme 5, lignes 30 à 38).

La manière dont les instructions sont encodées est formalisée dans l'Algorithme 6. Toutes les instructions sonores sont considérées comme des informations pertinentes et sont encodées dans la mémoire de travail. Si l'apprenant ne connaît pas l'instruction qu'il doit réaliser, alors elle est encodée dans la mémoire de travail au premier niveau de complexité, tout en tenant compte du contenu de cette instruction stocké en mémoire sensorielle. Ainsi, si l'instruction porte sur l'opération, alors un *chunk* contenant l'opération est ajouté dans la mémoire de travail (Algorithme 6, lignes 6 à 10). Si l'instruction porte également sur le but, un autre *chunk* est ajouté, portant cette information (Algorithme 6, lignes 11 à 17). Si l'instruction porte également sur les ressources de l'action, alors ces informations sont déjà prises en compte par l'Algorithme 5 qui traite l'encodage des entités.

Si l'apprenant connaît déjà l'instruction, alors elle est encodée dans sa mémoire de travail dans un *chunk* contenant un `ActionNode` lui même possédant les informations déjà présentes dans la mémoire à long terme (Algorithme 6, lignes 19 à 23). Si l'`ActionNode` ne dispose pas d'informations, dans la mémoire à long terme, sur le but et que celui-ci a été donné dans l'instruction, alors un autre *chunk* est instancié pour le stocker (Algorithme 6, lignes 24 à 31).

3.2.3.3 Étape 3 : le stockage

Le stockage consiste à transférer certaines informations de la mémoire de travail vers la mémoire à long terme. Nous avons présenté, dans la Section 3.2.2.4, le formalisme de stockage dans la mémoire à long terme et nous avons identifié deux types de connaissances : les connaissances déclaratives et les connaissances procédurales. En se basant sur des règles issues de la psychologie cognitive en rapport avec le transfert des informations vers la mémoire à long terme et leur stockage, nous considérons que toute connaissance déclarative peut être transférée directement en mémoire à long terme. Tandis que les connaissances procédurales sont transférées au cours des répétitions de l'exécution

Algorithme 6 Encodage des instructions en mémoire de travail.

```
1 : Integer iInst;
2 : for iInst = 0 .. learner.Sensorymemory.timedInstructions.Length do
3 :    TimedInstruction inst =
4 :         learner.Sensorymemory.timedInstructions[iInst];
5 :    if !learner.Longtermmemory.hasInstruction(inst.actionNode) then
6 :       if inst.operation then
7 :          if !learner.Workingmemory.hasOperation(inst.actionNode) then
8 :             OperationInformation oi = new OperationInformation();
9 :             oi.operation = inst.actionNode.operation;
10 :            learner.Workingmemory.addInformation(oi);
11 :      if inst.goal then
12 :         if !learner.Workingmemory.hasGoal(inst.actionNode) then
13 :            Constraint goal =
14 :                 new Constraint(inst.actionNode.postCondition);
15 :            ConstraintInformation ci = new ConstraintInformation();
16 :            ci.Goal = goal;
17 :            learner.Workingmemory.addInformation(ci);
18 :    else
19 :       ActionNodeInformation ani = new ActionNodeInformation();
20 :       ActionNode an =
21 :            learner.Longtermmemory.getInstruction(inst.actionNode);
22 :       ani.actionNode = an;
23 :       learner.Workingmemory.addInformation(ani);
24 :       if an.postCondition == null then
25 :          if inst.goal then
26 :             if !learner.Workingmemory.hasGoal(inst.actionNode) then
27 :                Constraint goal =
28 :                     new Constraint(inst.actionNode.postCondition);
29 :                ConstraintInformation ci = new ConstraintInformation();
30 :                ci.Goal = goal;
31 :                learner.Workingmemory.addInformation(ci);
```

de la procédure.

Nous présentons par la suite la manière dont les connaissances procédurales peuvent être stockées en mémoire à long terme. Selon Ganier, le transfert de certains éléments liés à une action, de la mémoire de travail vers la mémoire à long terme, a lieu lorsque l'apprenant a réalisé l'action (Ganier 2004). Lorsqu'une information est transférée de la mémoire de travail vers la mémoire à long terme, elle est stockée selon un niveau de complexité immédiatement supérieur à celui de la mémoire de travail. Nous avons défini trois niveaux de complexité pour les entités et cinq niveaux pour les actions (Tableaux 3.1 et 3.2). Ceci permet de respecter le principe de *chunking*. Ainsi, lorsque l'information sera à nouveau récupérée de la mémoire à long terme vers la mémoire de travail, elle n'occupera plus le même nombre de *chunks*, ce qui permettra de palier la capacité limitée de la mémoire de travail.

Par la suite, nous expliquons comment le transfert s'effectue quand l'action réalisée par l'apprenant est correcte. Nous transférons deux types d'informations : les informations liées à l'entité à manipuler et les informations liées à l'action à réaliser. Dans les deux cas, le principe est d'identifier si l'action et/ou l'entité existe déjà dans la mémoire à long terme ou non. Si l'entité et/ou l'objet n'existent pas dans la mémoire à long terme, alors ils seront stockés selon le premier niveau de complexité. Dans le cas contraire, nous identifions le niveau de complexité dans lequel ils ont été stockés en fonction des informations qu'ils contiennent et nous ajoutons les informations du niveau de complexité supérieur. Nous rappelons également que la procédure et ses actions sont stockées en mémoire à long terme par un diagramme d'activité. Le transfert en mémoire à long terme revient donc à la construction en temps réel et itérative d'un diagramme d'activité.

Lorsque l'action a été correctement réalisée par l'apprenant, il y a un stockage d'informations sur l'entité (Algorithme 7). Nous vérifions d'abord que l'entité existe ou pas dans la mémoire à long terme. Si elle n'existe pas, nous l'ajoutons mais sans sa forme géométrique `ActiveShape` (premier niveau de complexité). Dans le cas contraire, nous vérifions que l'entité stockée en mémoire à long terme possède une géométrie ou non. Si elle n'en possède pas, nous l'ajoutons (deuxième niveau de complexité). Dans le cas contraire, nous ne modifions pas le contenu de la mémoire à long terme. Le troisième niveau de complexité n'est pas pris en compte lors de la réalisation des actions, parce qu'il porte sur les connaissances déclaratives.

De plus, lorsque l'action a été correctement réalisée par l'apprenant, il y a également stockage d'informations sur l'action. Ce stockage est formalisé par l'Algorithme 8. Nous commençons tout d'abord par récupérer la procédure en cours de stockage à partir de la mémoire à long terme. Puis, nous vérifions que l'action réalisée existe déjà dans la procédure. Si elle n'existe pas, nous instancions un `ActionNode` auquel nous ajoutons la description et l'opération à réaliser (premier niveau de complexité). Si elle existe, nous vérifions qu'elle contient les références vers les entités à manipuler (`ObjectNode`). Si elle ne les contient pas, nous les ajoutons (deuxième niveau de complexité). Si l'action existe

Algorithme 7 Transfert des informations sur l'entité vers la mémoire à long terme.

```
1 : Entity entityWM = learner.Workingmemory.getEntity(ressource.name);
2 : if entityWM then
3 :    Entity entityLTM = learner.Longtermmemory.getEntity(ressource.name);
4 :    if !entityLTM then
5 :       entityWM.ActiveShape = null;
6 :       learner.Longtermmemory.addInstance(entityWM);
7 :    else
8 :       if !entityLTM.ActiveShape then
9 :          entityLTM.ActiveShape = entityWM.ActiveShape
```

et qu'elle contient les références vers les entités à manipuler, alors nous vérifions que l'action référence ses flux entrants (`ControlFlow`, lien vers les actions précédentes). Si ce n'est pas le cas, nous les ajoutons (troisième niveau de complexité). Cette idée est appliquée également pour le quatrième niveau de complexité dans lequel nous ajoutons le but de l'action (`Constraint`) et pour le cinquième niveau dans lequel nous ajoutons la référence à l'activité de niveau hiérarchique immédiatement supérieur.

Algorithme 8 Transfert des informations sur l'action vers la mémoire à long terme.

```
1 : Procedure proc = learner.Longtermmemory.Model.organisation.procedure;
2 : if !procedureHasAction(proc, actionToDo) then
3 :    ActionNode an = new ActionNode(actionToDo.name);
4 :    an.Action = new CallOperationAction(actionToDo.Action);
5 :    an.Description = actionToDo.Description;
6 :    procedure.Activity.addNode(an);
7 : else
8 :    ActionNode an = procedureGetAction(proc, actionToDo);
9 :    if an.InputPins.Length == 0 then
10 :      an.InputPins.copy(actionToDo.Action.InputPins);
11 :   else
12 :      if an.Incoming.Count == 0 then
13 :         ActionNode previous = getPreviousAction(actionToDo);
14 :         ActivityEdge edge = new ActivityEdge("ControlFlow");
15 :         edge.Source = previous;
16 :         edge.Target = currentAction;
17 :         previous.Outgoing.Add(edge);
18 :         edge.Target.Incoming.Add(edge);
19 :         procedure.Activity.addEdge(edge);
20 :      else
21 :         if an.goal == "" then
22 :            an.goal = actionToDo.goal;
```

Anderson (ANDERSON 1983) considère que pour apprendre une procédure, l'apprenant devrait la répéter plusieurs fois. À la fin de chaque répétition, nous enregistrons le contenu de la mémoire à long terme dans deux fichiers. La mémoire à long terme étant une sous partie de l'environnement et du modèle, nous utilisons le même format que ceux

utilisés par MASCARET : XML pour les instances et XMI pour le modèle. Ils sont lus au début de chaque répétition (c'est-à-dire, à chaque exécution de la procédure) pour instancier la mémoire à long terme de l'apprenant. Ces fichiers peuvent être stockés dans un LMS, car ils représentent l'évolution de l'apprenant.

3.3 Modèle du tuteur

Classiquement, le modèle du tuteur effectue des choix d'assistance pédagogique en fonction du modèle du domaine et du modèle de l'apprenant (WOOLF 1992). Dans notre cas le modèle du tuteur s'appuie également sur les scénarios pédagogiques qui ont été rédigés *a priori* par un formateur. Notre objectif dans ce modèle est d'adapter l'exécution de ces scénarios pédagogiques en fonction des connaissances de l'apprenant et des interactions entre le tuteur et l'apprenant. Dans notre travail, les connaissances de l'apprenant sont stockées dans ses trois mémoires (Section 3.2). Les interactions entre le tuteur et l'apprenant sont représentées *via* un agent conversationnel animé (Section 3.1). Nous décrivons dans ce chapitre le comportement adaptatif du tuteur.

3.3.1 Comportements

Dans MASCARET, les agents peuvent avoir plusieurs comportements, qui peuvent s'exécuter cycliquement et en parallèle. Le comportement global du tuteur adaptatif que nous proposons est composé de trois comportements.

Le premier comportement est le comportement de suivi de procédure « `ProceduralBehavior` ». Ce comportement existe déjà dans MASCARET. Nous ne faisons que l'utiliser. Ce comportement est important dans le cadre du comportement global du tuteur pour deux raisons. La première est que ce comportement permet au tuteur d'exécuter le scénario pédagogique, qui décrit l'ensemble des actions pédagogiques que doit réaliser le tuteur. La deuxième raison est que ce comportement permet au tuteur de connaître les actions que doit réaliser l'apprenant afin de le guider.

Le deuxième comportement est celui de communication « `CommunicationBehavior` », qui permet à l'agent de recevoir et d'envoyer des messages à d'autres agents. Nous rappelons que l'utilisateur humain est considéré comme un agent dans l'environnement. Ce comportement est également déjà présent dans MASCARET. Les messages sont véhiculés dans MASCARET par des actions de communication (QUERREC et al. 2018), dont le contenu est exprimé en FIPA-SL et peuvent être exécutées par un ACA (Section 3.1).

Nous définissons un troisième comportement qui est le comportement adaptatif « `AdaptiveTutorBehavior` ». Ce dernier permet au tuteur d'adapter l'exécution du scénario pédagogique en fonction de ce que l'apprenant fait ou sait. Ce comportement utilise les deux premiers comportements afin d'obtenir des informations sur les actions à réaliser et

sur les messages échangés entre le tuteur et l'apprenant. Nous décrivons dans la section suivante les principes généraux de ce comportement.

3.3.2 Principes généraux du comportement adaptatif

Le comportement du tuteur adapte l'exécution du scénario pédagogique en fonction du modèle de l'apprenant représenté dans notre travail par les mémoires de l'apprenant (Section 3.2). La Figure 3.26 décrit le comportement de prise de décision du tuteur. Ce comportement prend tout d'abord en compte l'action réalisée par l'apprenant.

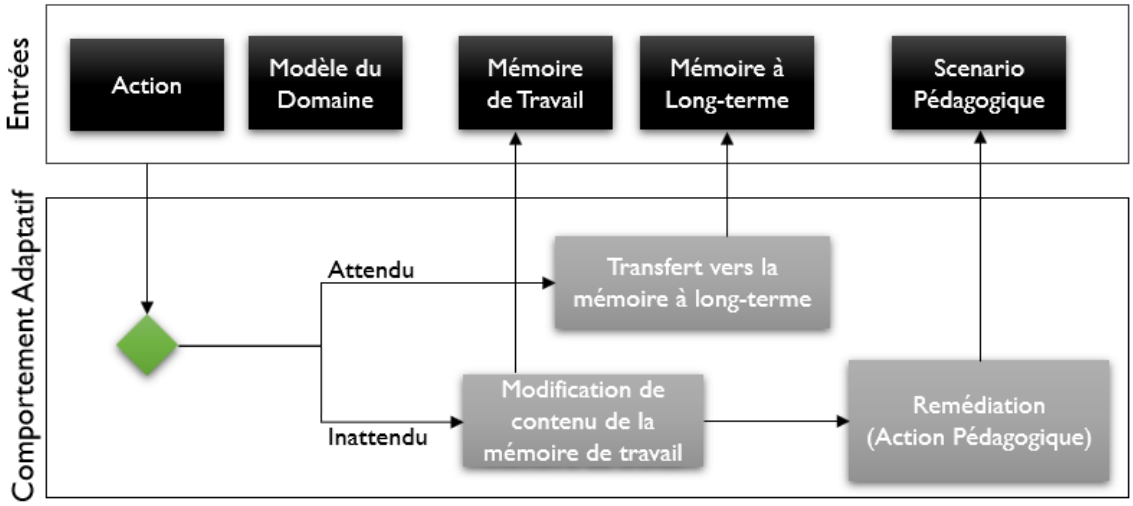


FIGURE 3.26 – Comportement de prise de décision du tuteur.

Nous définissons deux types d'actions :

- actions liées au modèle du domaine : une action peut être soit une action du domaine sur un objet spécifique ou une réponse aux questions du tuteur, voir même une inaction. Le tuteur s'appuie sur le modèle du domaine pour vérifier si cette action est considérée comme une erreur ou non. Si l'action est incorrecte, alors elle est considérée comme inattendue par le tuteur.

- actions liées aux interactions : une action réalisée par l'apprenant peut aussi être un *feedback* à l'action du tuteur (par exemple, une expression faciale, une question, observer l'environnement, etc.). Dans ce cas, au lieu d'utiliser les connaissances du domaine, le tuteur évalue si le *feedback* est négatif ou non. Si le *feedback* est négatif, cela est également considéré comme inattendu par le tuteur.

Si l'action réalisée par l'apprenant est attendue par le tuteur (par exemple, action ou réponse correcte), alors le transfert vers la mémoire à long terme se produit. L'adaptation de l'exécution du scénario a lieu lorsque l'action effectuée par l'apprenant est inattendue (par exemple, action incorrecte, expression facial négative). Dans ce cas, le tuteur modifie

l'inférence sur le contenu de la mémoire de travail et réalise une autre action pédagogique.

Par exemple, si selon le scénario pédagogique, le tuteur explique la prochaine action que l'apprenant doit faire, on instancie deux *chunks* dans la mémoire de travail, un premier pour l'`Action` et un deuxième pour l'`Entity` (comme expliqué dans la Section 3.2). Si l'apprenant réalise une action inattendue (par exemple, s'il montre une expression faciale négative), le comportement du tuteur considère alors que l'apprenant ne connaît pas la position de l'objet, contrairement à ce que le tuteur a inféré. Dans ce cas, le tuteur remédie à cette situation en réévaluant le contenu de la mémoire de travail de l'apprenant puis en réalisant une nouvelle action pédagogique pour mettre en évidence l'objet.

3.3.3 Implémentation du comportement adaptatif du tuteur

Dans le cadre de ce travail, nous avons implémenté un comportement adaptatif de tuteur, qui respecte les principes proposés dans la section précédente.

Lors de la réalisation du comportement du tuteur, ce dernier inspecte tout d'abord les actions de communication non-verbales réalisées par l'apprenant. Comme nous avons vu dans le modèle de l'interface (Section 3.1), nous avons commencé à intégrer la notion d'expression faciale. La relation entre les expressions faciales et les émotions de l'apprenant constitue encore un travail de recherche qui dépasse le cadre de notre travail. Nous montrons par la suite, uniquement comment ces expressions faciales ou les émotions exprimées peuvent influencer le comportement du tuteur. L'Algorithme 9 décrit comment nous prenons en compte une expression faciale négative pour adapter le scénario pédagogique.

Si l'expression faciale détectée est « `FROWN` » (sourcils froncés), alors le tuteur considère que la dernière instruction a été mal comprise. Ceci ce fait dans les lignes 3 à 11 de l'Algorithme 9, en vérifiant que le dernier message envoyé par le tuteur est bien une performative « `INFORM` » et dont le contenu porte sur une action. Nous n'avons pas pris en compte pour l'instant les autres performatives, par exemple « `QUERY-REF` » dans le cas où le dernier message envoyé par le tuteur était une question. Nous n'avons également, pour l'instant, pas pris en compte les autres comportements non verbaux que nous avons présenté dans le modèle de l'interface (Section 3.1).

Si le message portait bien sur une action, le tuteur vérifie l'état de la mémoire de travail de l'apprenant. Si l'apprenant connaît l'entité à manipuler dans l'action (Algorithme 9, lignes 12 à 16), alors le tuteur considère l'expression faciale négative comme un manque de connaissances déclaratives sur cette entité. Le tuteur décide donc de réaliser une nouvelle action pédagogique et de décrire cette entité *via* une action de communication qui porte sur la description de cette entité, provenant du modèle du domaine (Algorithme 9, lignes 17 à 26).

La deuxième étape du comportement adaptatif du tuteur prend en compte la situation dans laquelle l'apprenant ne réalise pas d'action. La première fois que le tuteur ne détecte pas d'action de l'apprenant, au bout d'un certain temps limite (`TIMEOUT`), il vérifie

Algorithme 9 Adaptation de scénario pédagogique en fonction des expressions faciales.

```
1 : String emotion = learner.getEmotion();
2 : if emotion == "FROWN" then
3 :    ACLMessage msg = tutor.Mailbox.getLastSentMessage();
4 :
5 :    if msg.Performative == ACLPerformative.INFORM then
6 :       string content = msg.Content;
7 :       commBehavior = tutor.getBehavior("CommunicationBehavior");
8 :       procBehavior = tutor.getBehavior("ProceduralBehavior");
9 :       FIPAResult result = commBehavior.parseFipaExpression(content);
10 :
11 :      if result.isAction then
12 :         ActionNode action = commBehavior.findNextAction(learner.name);
13 :         ObjectNode objNode = action.getIncomingObjectNode();
14 :         Entity entity = procBehavior.findEntityInProcedure(objNode);
15 :
16 :         if learner.Workingmemory.hasEntity (entity) then
17 :            string res = objNode.Description;
18 :            CommunicationAction act = new CommunicationAction();
19 :            act.performative = ACLPerformative.INFORM;
20 :            act.receivers.Add(msg.Sender);
21 :            act.FipaContent = "((= (iota ? description ( slot ?";
22 :            act.FipaContent += "description ?" + entity.name + ")) ";
23 :            act.FipaContent += res + "))";
24 :            act.naturalContent = res;
25 :            act.ressources.Add(entity);
26 :            commBehavior.sendMessage(acttion);
```

que l'apprenant possède l'objet à manipuler dans sa mémoire de travail. Si ce n'est pas le cas, alors le tuteur met l'objet à manipuler en évidence, tout en énonçant son nom (Algorithme 10, lignes 6 à 12). Si malgré cette assistance, l'apprenant ne fait toujours rien, le tuteur énonce l'action à réaliser si celle si est bien présente dans la mémoire de travail de l'apprenant (Algorithme 10, lignes 14 à 19). Enfin, si ces deux assistances n'ont toujours pas aidé l'apprenant, le tuteur énonce l'action et son but (Algorithme 10, lignes 21 à 28).

Algorithme 10 Adaptation de scénario pédagogique en fonction de l'inaction.

```
1 : int nbActionTuteur = getNbActionTuteur();
2 : float currentTime = Time.time;
3 : float deltaTime = currentTime - lastInteractionTime;
4 : ActionNode knownAction = learner.WorkingMemory.getAction(action);
5 : Entity knownEntity = learner.WorkingMemory.getEntity(entity);
6 : if nbRepetition == 0 && deltaTime > TIMEOUT then
7 :     if knownEntity && knownEntity.ActiveShape then
8 :         putInEvidence(knownEntity);
9 :     else
10 :        say(knownEntity.name);
11 :        putInEvidence(knownEntity);
12 :    nbRepetition++;
13 : else
14 :    if nbRepetition == 1 && deltaTime > TIMEOUT then
15 :        say(actionToDo.Description);
16 :        if knownAction && knownAction.Incoming.Count != 0 then
17 :            knownAction.Incoming.Clear();
18 :            knownAction.goal = " ";
19 :        nbRepetition++;
20 :    else
21 :        if nbRepetition == 2 && deltaTime > TIMEOUT then
22 :            string sentence = "Pour " + actionToDo.goal + ", ";
23 :            sentence += actionToDo.Description;
24 :            say(sentence);
25 :            if knownAction && knownAction.Incoming.Count != 0 then
26 :                knownAction.Incoming.Clear();
27 :                knownAction.goal = " ";
28 :            nbRepetition++;
```

La troisième étape consiste à vérifier l'action réalisée par l'apprenant. Si l'action réalisée est une action correcte selon la procédure (la bonne opération, sur le bon objet, Algorithme 11, lignes 9 et 10), alors le tuteur laisse l'apprenant réaliser l'action (Algorithme 11, ligne 11). Comme cette action est correcte, le tuteur transfère les informations de cette action en mémoire à long terme de l'apprenant (Algorithme 11, lignes 12 et 13).

Si l'apprenant fait une erreur en essayant de manipuler le mauvais objet, le tuteur ne l'autorise pas à réaliser l'action. Il réalise une nouvelle action pédagogique (action de communication) en redonnant le nom du mauvais objet manipulé par l'apprenant et le nom de

Algorithme 11 Comportement du tuteur en cas d'action correcte de l'apprenant.

```
1 : if learner.wantedAction then
2 :    Action action = learner.wantedAction;
3 :    Action actionToDo = learnerProceduralBehavior.actionToDo();
4 :    Operation learnerOperation = action.Operation;
5 :    Operation operationToDo = actionToDo.Operation;
6 :    ObjectNode wantedObjectNode = action.getIncomingObjectNode();
7 :    ObjectNode correctObjectNode = actionToDo.getIncomingObjectNode();
8 :
9 :    if (learnerOperation == operationToDo) then
10 :      if (wantedObjectNode == correctObjectNode) then
11 :         actionToDo.start (learner);
12 :         if learner.Workingmemory.hasOperationOrAction(actionToDo) then
13 :            transferOnCorrectActionAndObject(learner, actionToDo);
14 :      else                                          ▷ voir Algorithme 12
15 :   else                                             ▷ voir Algorithme 13
```

l'objet qu'il aurait dû manipuler (Algorithme 12, lignes 8 à 10). Le tuteur vérifie que l'apprenant possède l'objet à manipuler, ainsi que sa géométrie, dans sa mémoire de travail. Si c'est le cas, le tuteur met l'objet à manipuler en évidence et retire le lien entre l'objet et sa géométrie dans la mémoire de travail (Algorithme 12, lignes 1 à 4). Ceci permet de retirer la connaissance qui fait le lien entre l'objet et sa géométrie dans les mémoires de l'apprenant. Si ce n'est pas le cas, alors le tuteur met l'objet à manipuler en évidence, tout en énonçant son nom (Algorithme 12, lignes 5 à 7).

Algorithme 12 Comportement du tuteur en cas d'erreur sur l'objet.

```
1 : Entity knownEntity = learner.WorkingMemory.getEntity(entity);
2 : if knownEntity && knownEntity.ActiveShape then
3 :    putInEvidence(knownEntity);
4 :    knownEntity.ActiveShape = null;
5 : else
6 :    say(knownEntity.name);
7 :    putInEvidence(knownEntity);
8 : string sentence = "Ceci est le " + wantedObjectNode.name;
9 : sentence += " et non le " + correctObjectNode.name;
10 : say(sentence);
```

Si l'erreur réalisée par l'apprenant est liée à l'action, le tuteur n'autorise pas l'apprenant à la réaliser. Il réalise dans ce cas une nouvelle action pédagogique (action de communication), en redonnant le nom de l'objet à manipuler et en précisant l'action à réaliser (Algorithme 13, lignes 5 à 7). Le tuteur vérifie également que l'action est stockée dans la mémoire de travail de l'apprenant. Si c'est le cas, alors le tuteur retire les flux d'entrée de l'action ainsi que son but de la mémoire de travail de l'apprenant. Ceci est réalisé parce que le tuteur considère, à ce stade, que l'apprenant ne connaît pas l'enchainement entre

l'action précédente et l'action qu'il aurait dû réaliser.

Algorithme 13 Comportement du tuteur en cas d'erreur sur l'action.

```
1 : ActionNode knownAction = learner.WorkingMemory.getAction(actionToDo);
2 : if knownAction && knownAction.Incoming.Count != 0 then
3 :    knownAction.Incoming.Clear();
4 :    knownAction.goal = " ";
5 : string sentence = "Ceci est bien le " + correctObjectNode.name;
6 : sentence += " mais tu dois " + actionToDo.Description;
7 : say(sentence);
```

3.4 Bilan

Dans ce chapitre nous avons présenté notre modèle MEMORIA, un modèle de représentation de la mémoire de l'apprenant pour les systèmes tutoriels intelligents et adaptatifs. L'apport majeur de ce modèle réside dans une formalisation et une implémentation du modèle de l'apprenant sous forme de mémoires qui stockent les informations perçues par l'apprenant dans un environnement virtuel. Ces informations portent essentiellement sur les instructions et les objets à manipuler dans le cadre de procédure sur des systèmes techniques. Nous avons choisi d'organiser la conception de notre modèle autour des quatre modèles de connaissances classiques dans les systèmes tutoriels intelligents.

Le modèle du domaine est représenté par les connaissances métiers formalisées à l'aide du méta-modèle MASCARET. Nous nous appuyons sur ce méta-modèle pour définir les trois autres modèles d'un système tutoriel intelligent.

Le modèle de l'interface est représenté par un agent conversationnel animé, afin de rendre naturelles les interactions entre le tuteur et l'apprenant. L'agent est capable de générer des communications verbales et non verbales grâce aux connaissances sur le domaine métier en fonction d'un scénario pédagogique prescrit ou d'un comportement de tuteur adaptatif. L'agent est également capable de reconnaître les communications verbales et non verbales provenant de l'apprenant. Celles-ci servent de données d'entrée au comportement adaptatif du tuteur.

Le modèle de l'apprenant est constitué de l'ensemble des connaissances acquises par l'apprenant en cours de simulation. Ces connaissances sont organisées dans trois mémoires : la mémoire sensorielle, la mémoire de travail et la mémoire à long terme. L'enjeu majeur de cette thèse porte sur la formalisation de l'encodage des informations dans ces mémoires, ainsi que le flux d'information entre celles-ci.

Le modèle de tuteur que nous proposons est centré sur la réalisation d'un comporte-

ment qui adapte l'exécution du scénario en fonction des connaissances de l'apprenant et de ses interactions avec le tuteur.

Nous considérons que ce comportement adaptatif devrait permettre une amélioration des performances d'apprentissage. Dans le chapitre suivant (Chapitre 4) nous présentons les expériences que nous avons menées afin de valider cette hypothèse.

4 Évaluation

Dans le Chapitre 3, nous avons présenté notre modèle de tuteur adaptatif pour l'apprentissage de procédure. Pour apprendre une procédure il faut la comprendre mais aussi la répéter. Durant les répétitions, le niveau de compétence des apprenants évoluent différemment. Le scénario pédagogique qui est défini par un formateur est donc sans doute pertinent lors du premier essai pour tous les participants, mais ne l'est plus durant les essais suivants. Le comportement de tuteur que nous proposons, adapte l'exécution du scénario en fonction des connaissances de l'apprenant et de ses interactions avec le tuteur. Nous considérons que cette adaptation permet d'améliorer les performances de l'apprenant lors de l'apprentissage. C'est ce que nous souhaitons valider dans ce chapitre, précisément dans l'expérience II (Section 4.3).

Le modèle que nous proposons est destiné à l'apprentissage en environnement virtuel. Ceci a guidé les choix que nous avons réalisés dans le modèle d'interface. Avant d'évaluer l'intérêt du comportement adaptatif du tuteur, nous devons tout d'abord :

- Démontrer que notre modèle s'applique à des contextes professionnels complexes qui peuvent être simuler en environnement virtuel. Ceci est l'objet de la Section 4.1.

- Identifier la situation de réalité virtuelle la plus pertinente. Ceci est présenté dans l'expérience I (Section 4.2).

Sommaire

4.1 Environnement virtuel et procédure

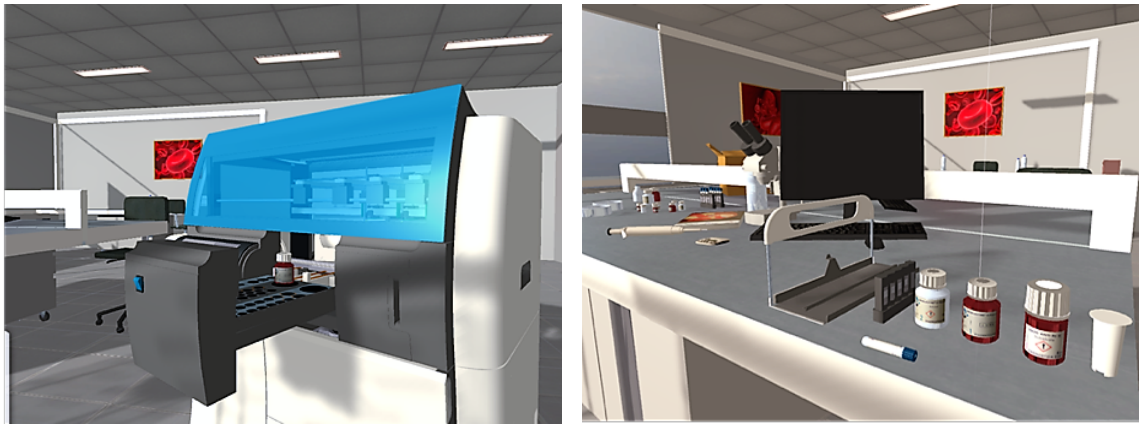
L'Environnement Virtuel pour l'Apprentissage Humain (EVAH) que nous avons utilisé pour valider notre modèle est une évolution de celui utilisé par Hoareau (Hoareau 2016). Cet environnement avait été conçu à l'origine dans le cadre de la thèse de Le Corre (Le Corre 2013). Nous avons porté le projet sous Unity 3D afin d'améliorer le réalisme graphique et la réalisation des comportements des objets de la scène et des actions de la procédure.

Cet EVAH permet de réaliser une procédure de lancement de tests de coagulation sanguine. Il représente un laboratoire d'analyses médicales dans lequel se trouvent des paillasses, des réactifs et un automate d'analyse qui permet de tester des échantillons de sang humain (Figures 4.1 et 4.2).

Figure 4.1 – Présentation de l'environnement virtuel : le laboratoire d'analyses médicales.

Dans cette simulation, le matériel à utiliser est disposé sur une paillasse : l'échantillon de sang contenu dans un tube à essai, un *rack*, un panier, différents réactifs biochimiques (neoplastine et solvant) et un *maxi-reducer* pour diminuer l'étranglement du flacon. D'autres éléments sont présents dans la scène mais n'ont pas d'utilité propre à la procédure à appliquer. Ils permettent simplement de se rapprocher d'une situation réelle (par exemple, d'autres paillasses, un évier, un ordinateur qui en situation réelle sert d'interface avec l'automate de coagulation, etc.).

La position initiale des participants dans l'EVAH a été choisie afin qu'ils puissent visualiser les objets utiles. Ils n'ont donc pas à effectuer de déplacements physiques importants (comme marcher dans la pièce) pour interagir avec les objets utiles. Une position debout, prédéfinie au début de la passation, a été fixée pour que chaque participant, quelque soit la condition, ait une vue similaire de la scène initiale.



(a) L'automate de coagulation. (b) Quelques produits utilisés.

FIGURE 4.2 – Présentation des éléments constituant la scène.

Nous avons réalisé une première expérience (Section 4.2) qui avait pour objectif d'identifier la meilleur situation immersive pour l'apprentissage d'une procédure. Dans cette expérience l'apprentissage concernait une procédure de lancement de tests sanguins sur un automate de coagulation. Cette procédure est enseignée lors de formations dispensées par du personnel spécialisé auprès des nouveaux acquéreurs d'un instrument de diagnostic en hémostase. La procédure contient 38 actions que l'apprenant doit exécuter (Annexe A). Cette procédure a également été utilisée pour la deuxième expérience (Section 4.3). Cependant, les 38 actions ont été légèrement modifiées, en supprimant et rajoutant quelques autres actions. Dans le deuxième expérience, la procédure comprenait finalement 36 actions au total (Annexe E). Dans expérience, les instructions étaient présentées sous forme sonore par un agent virtuel incarné.

4.2 Expérience I

Des travaux antérieurs (HOAREAU 2016) ont validé que la procédure de lancement de tests sanguins peut être apprises dans l'environnement virtuel présenté plus haut. Cependant l'expérience a été menée dans un contexte de *VR Desktop*, c'est-à-dire que l'environnement était présenté sur un écran d'ordinateur et les interactions se faisaient par le biais d'un clavier et d'une souris. Dans notre étude, nous souhaitons valider que cette procédure peut être apprise dans l'environnement virtuel amélioré et dans des conditions immersives plus poussées.

Le but de cette première expérience (`ExpeI`) est d'identifier la meilleure situation immersive pour l'apprentissage de cette procédure (écran, casque ou CAVE). En effet, les résultats des travaux concernant les bénéfices possibles des dispositifs visuellement immersifs sont assez contradictoires. Les travaux effectués par Sowndararajan et ses collègues (SOWNDARARAJAN et al. 2008) ont mis en évidence qu'un niveau plus élevé d'immersion peut être plus efficace dans l'apprentissage des procédures complexes car ces

procédures font référence à des emplacements spatiaux. Toutefois, Morineau (Morineau 2000), qui s'est intéressé à l'effet de l'immersion sur l'activité cognitive, a observé une dégradation des performances qui serait due à l'utilisation d'un nouveau dispositif et à l'augmentation d'informations à traiter.

4.2.1 Participants

Quarante-cinq personnes (dont 9 femmes), âgées de 14 ans à 48 ans (âge moyen : 23 ans, σ = 6.21) étaient volontaires pour participer à cette étude. L'échantillon était constitué d'étudiants provenant d'écoles d'ingénieurs, d'un collège et de chercheurs de laboratoire. Certains participants avaient déjà vu et/ou utilisé le matériel (système d'interaction et dispositif d'immersion visuelle). Des groupes indépendants ont été constitués pour éviter les effets d'apprentissage : les participants étaient répartis sur chaque dispositif de manière équivalente, en préservant l'homogénéité des caractéristiques recueillies.

4.2.2 Protocole expérimental

L'expérience s'est déroulé en passation individuelle. Chaque participant devait remplir un formulaire de consentement libre et éclairé (Annexe B) puis répondre à un questionnaire portant sur les caractéristiques individuelles (Annexe C). Les participants ont été répartis dans trois conditions expérimentales (15 participants par condition) : immersion faible, immersion moyenne et immersion élevée. Pour chacune des conditions, l'expérience était divisée en trois phases : une phase de démonstration, une phase de familiarisation et une phase de réalisation de l'expérience. Des consignes standardisées étaient données à l'oral par l'expérimentateur (Annexe D). La démonstration était réalisée par l'expérimentateur qui présentait à l'utilisateur le matériel et l'environnement virtuel à l'aide d'une procédure conçue à cet effet. Il s'agissait d'une procédure de déplacements de deux cubes de couleurs différentes. Cette procédure reprenait les actions de base de l'interaction avec l'EVAH (prendre un objet, cliquer sur un bouton, etc.) (Figure 4.3).

La phase de familiarisation s'appuyait sur cette même procédure et permettait aux participants de s'approprier les dispositifs de visualisation et d'interaction. Le but de cette phase était également de se familiariser avec le type de tâche proposé dans l'expérience. Il était possible de répéter plusieurs fois cette procédure si le participant le souhaitait. Avant la phase de réalisation de l'expérience, l'expérimentateur rappelait les différentes touches utiles (Annexe D). Il s'assurait alors de la compréhension du participant puisque par la suite il ne pouvait plus intervenir pour l'aider dans la réalisation de la procédure. La consigne donnée aux participants était de réaliser la procédure dans son intégralité un certain nombre de fois. Les participants n'étaient pas informés du nombre d'essais qu'ils allaient devoir réaliser. Cette dissimulation volontaire du nombre d'essais se basait sur les résultats de Ganier et ses collègues (Ganier et al. 2013) qui ont montré une dégradation des performances quasi-systématique lors du dernier essai réalisé par les apprenants. Une hypothèse explicative était que cette dégradation pouvait être due à une annonce de l'expérimentateur du type « Vous allez effectuer le dernier essai ».

FIGURE 4.3 – Présentation de la scène avec les cubes.

Lors de la phase de réalisation de l'expérience I, les participants devaient réaliser quatre essais de la procédure de lancement de tests de coagulation. Le choix du nombre d'essais est dû au fait que l'étude porte seulement sur le début de l'apprentissage de la procédure tel que défini par Fitts et Posner (FITTS et POSNER 1967) [1].

L'apprenant était guidé par 38 instructions pour réaliser la procédure. Pour chaque action, il devait sélectionner l'objet à manipuler, ce qui affichait un menu en 3D présentant la liste des actions possibles à réaliser sur cet objet. L'apprenant choisissait alors dans cette liste l'action qu'il souhaitait réaliser. Les instructions étaient présentées sous forme sonore. L'apprenant avait la possibilité de réécouter l'instruction correspondante à l'action à réaliser autant de fois qu'il le souhaitait. C'est seulement après que l'action correcte ait été exécutée que l'instruction suivante était disponible. Les instructions étaient des phrases courtes, comme par exemple « Prendre le flacon de neoplastine ». Une mise en saillance de l'objet concerné par l'action à réaliser était présente lors des 4 essais.

4.2.2.1 Dispositifs d'immersion visuelle

Dans cette étude, le niveau d'immersion visuelle était lié à la technologie utilisée. Le niveau d'immersion visuelle offert par les dispositifs a été défini par le degré de prégnance de l'environnement virtuel, tout en prenant en compte les caractéristiques techniques qui leurs sont inhérentes telles que la qualité des stimulations visuelles : résolution, monoscopie *versus* stéréoscopie etc. Grâce à ces considérations, nous avons définit trois niveaux d'immersion : faible, moyenne et élevée.

Pour le niveau d'« immersion faible », un écran plat de 49 pouces (62.5 cm de hauteur et 109 cm de largeur soit 124 cm de diagonale) et d'une résolution de 1920 × 1080 (60 Hz) a été utilisé (Figure 4.4). Il s'agissait d'un écran 2D monoscopique. L'image n'était pas recalculée à partir du point de vue du participant. La Figure 4.4 montre une image du

1. L'apprentissage de procédure est constitué de trois phases. La première phase étant la plus coûteuse en terme de temps et de charge cognitive, c'est précisément cette phase qui nous intéresse dans cette étude.

dispositif en cours d'utilisation et la Figure 4.5 montre la configuration, avec les différentes mesures, du dispositif. Le cercle représente la position initiale de l'utilisateur.

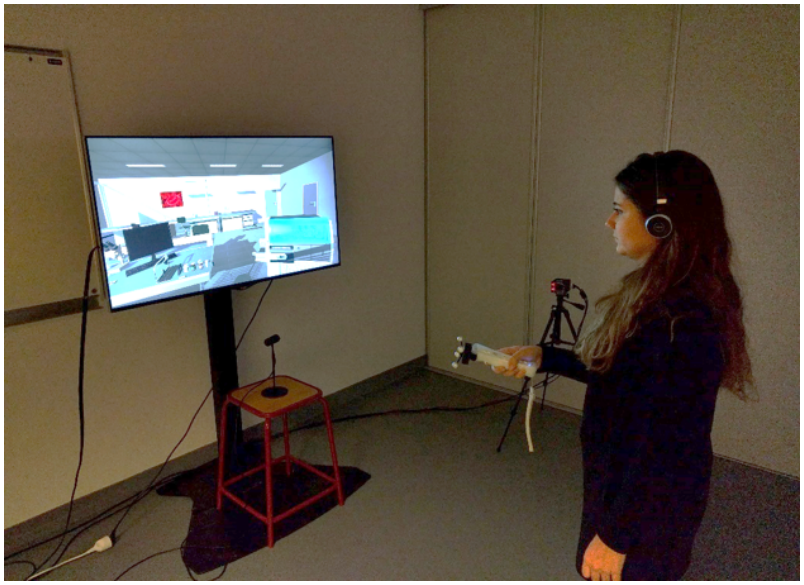


FIGURE 4.4 – Dispositif d'immersion faible : écran plat.

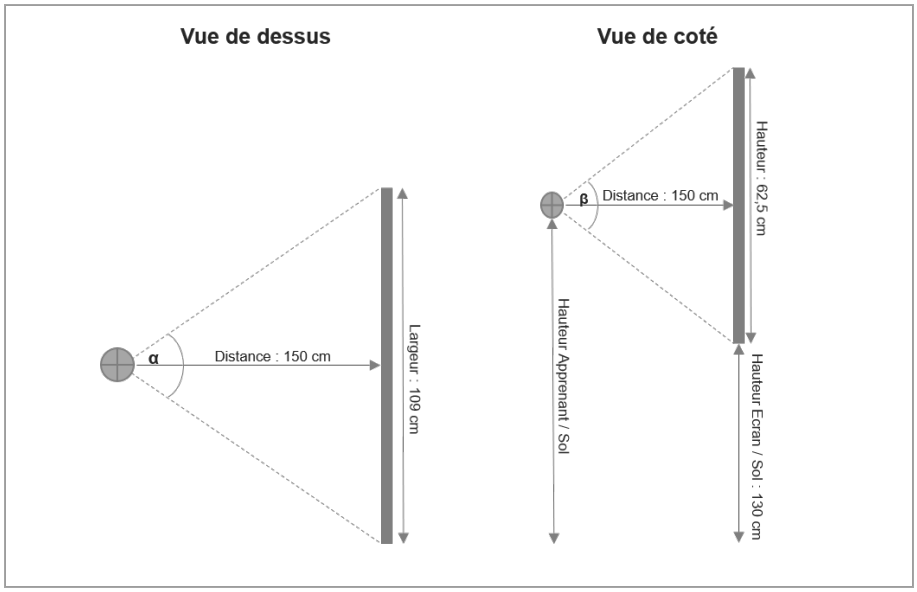


FIGURE 4.5 – Configuration du dispositif d'immersion faible et position de l'utilisateur.

Concernant le niveau d'« immersion moyen », le dispositif était un CAVE à 4 faces (dont une au sol) (Figure 4.6). Chaque face était d'une hauteur de 219.5 cm et d'une largeur de 194 cm. L'écran au sol avait une largeur de 250 cm et une longueur de 194 cm. Le CAVE était donc d'un volume d'environ 10 m^3. La résolution était de 1280 × 720 (60 Hz) par face. Les images étaient présentées en stéréo passive. Le participant portait donc des lunettes avec des filtres polarisants. Des détecteurs infrarouges suivaient les mouvements de la tête du participant grâce à des réflecteurs sphériques fixés sur les

lunettes. La Figure 4.7 représente la configuration, avec les différentes mesures, du CAVE. Le cercle représente la position initiale de l'utilisateur.

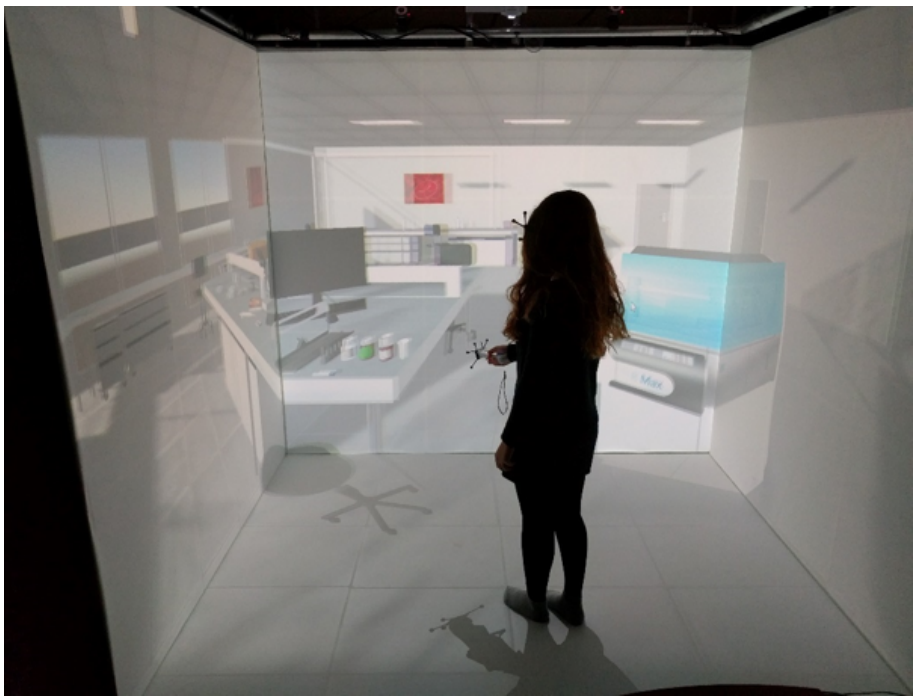


FIGURE 4.6 – Dispositif d'immersion moyenne : CAVE.

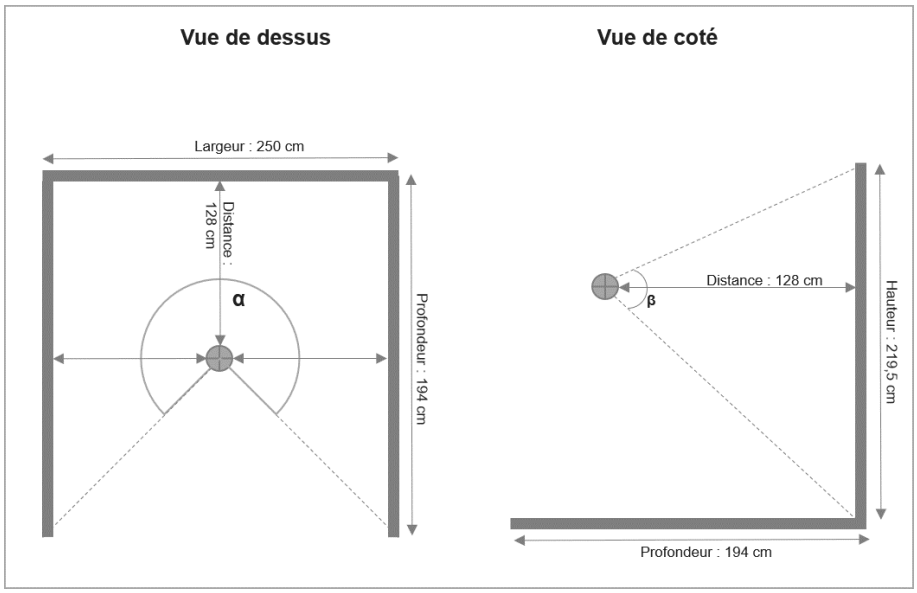


FIGURE 4.7 – Configuration du CAVE et position de l'utilisateur.

Enfin, pour la dernière configuration qui équivaut à une « immersion visuelle élevée », le dispositif utilisé était un casque *Oculus Rift*, version commerciale de 2016, (Figure 4.8), d'une résolution de 2160 × 1200 (soit 1080 × 1200 par œil) (90 Hz). Il s'agit d'un dispositif qui permet une vision 3D stéréoscopique avec captation des mouvements de la tête du

participant et qui offre un champ de vision équivalent à 110°.

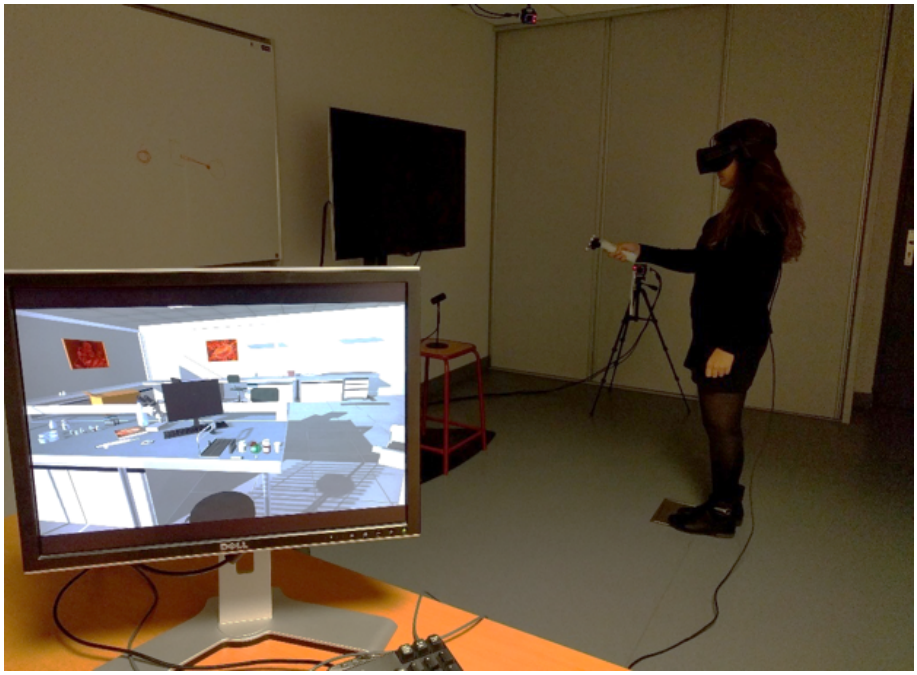


FIGURE 4.8 – Dispositif d'immersion élevée : casque *Oculus Rift*.

La Figure 4.8 montre une image du dispositif en cours d'utilisation et la Figure 4.9 montre la configuration, avec les différentes mesures, du dispositif.

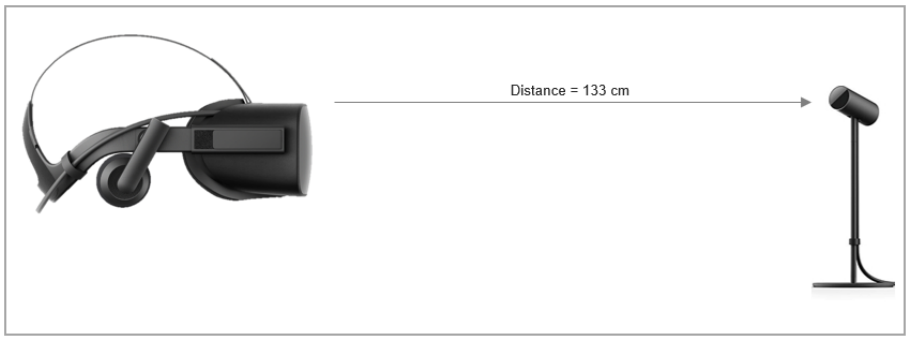


FIGURE 4.9 – Configuration du dispositif d'immersion élevée.

4.2.2.2 Système d'interaction

Afin d'éviter des biais expérimentaux, le dispositif d'interaction était identique pour chacun des dispositifs d'immersion visuelle. Le système d'interaction était une Wiimote sur laquelle un assemblage de réflecteurs sphériques pour caméra de Motion Capture (8 caméras OptiTrack pour la condition CAVE et 4 caméras ART Track pour les deux autres conditions) avait été fixé (Figure 4.10).

Le participant utilisait la Wiimote pour pointer l'objet qu'il voulait sélectionner. La position et l'orientation de la Wiimote étaient obtenues à l'aide du système de Motion Capture. Ce pointage était traduit dans l'environnement virtuel par une petite sphère jaune avec un

FIGURE 4.10 – Présentation du système d'interaction : Wiimote et caméra de Motion Capture.

degré de transparence permettant une vision de l'objet pointé. Le pointeur changeait de couleur (vert) lorsqu'il rencontrait un objet interactif, ce qui fournissait une aide préalable dans la visualisation des objets utiles lors de la procédure. Le pointeur répond donc à des critères ergonomiques d'incitation et de *feedback* immédiat (BACH 2004). L'interaction n'était pas optimale, mais ce choix provenait de l'utilisabilité de cette manette dans les 3 dispositifs.

Pour sélectionner un objet, le participant appuyait sur le bouton B correspondant à la gâchette arrière de la Wiimote (Figure 4.11). Il faisait alors apparaître un menu proposant la liste exhaustive des actions qu'il était possible de réaliser avec cet objet (Figure 4.12).

Dans ce menu, le participant passait d'une action à l'autre grâce aux flèches haut et bas de la Wiimote, et il sélectionnait celle qu'il souhaitait en appuyant de nouveau sur le bouton B. Les actions possibles sur l'objet étaient listées par ordre alphabétique et un item « Fermer Menu » permettait d'annuler la sélection de l'objet. Dans les 3 situations, le participant était guidé par des instructions sonores délivrées dans un casque audio intégré à l'*Oculus Rift* et un casque *Bluetooth* pour les deux autres dispositifs. L'aide était accessible via le bouton A de la Wiimote. Dans ce cas, l'instruction à réaliser était annoncée et l'objet à manipuler clignotait en rouge (Figure 4.13).

C'est en s'appuyant sur le principe d'aide prédictive de Wickens (WICKENS 2002) qui recommande d'apporter le guidage utile à l'apprenant avant qu'il en fasse la demande, et le principe de signalement (MAYER 2009), que le choix d'une mise en saillance des objets concernés par l'action à réaliser a été fait. En effet, l'apprenant n'avait pas à choisir entre l'instruction sonore ou la mise en saillance de l'objet, il devait simplement demander l'ins-

FIGURE 4.11 – Touches utiles pour interagir avec l'environnement (ExpeI).

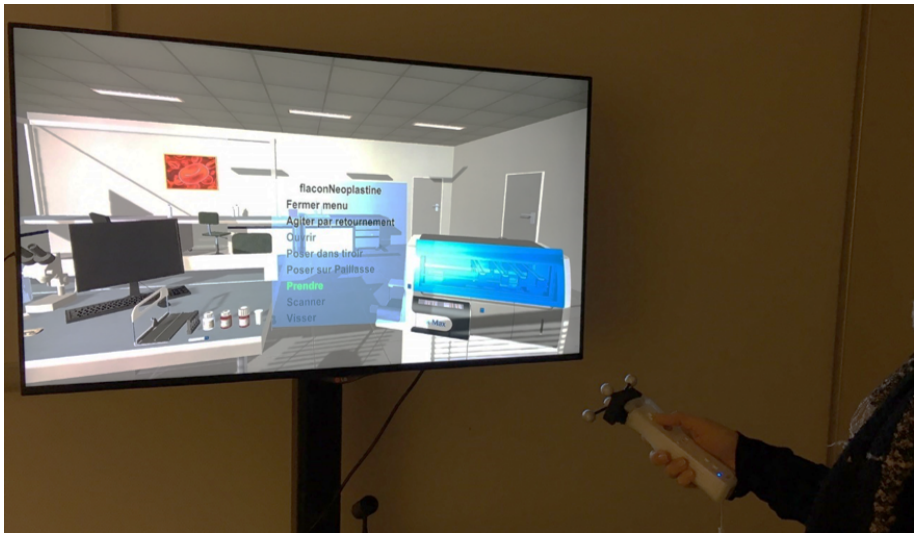


FIGURE 4.12 – Affichage du menu après sélection d'un objet (ExpeI).

truction sonore pour avoir l'aide visuelle également. La mise en saillance était destinée à orienter l'attention de l'apprenant sur l'élément auquel l'instruction faisait référence (HOAREAU 2016). Ce comportement était exécuté pour tous les participants et durant tous les essais. Dans cette expérience nous n'avons pas intégré le comportement adaptatif de notre modèle.

4.2.3 Résultats

Plusieurs indicateurs ont été utilisés pour évaluer l'impact du niveau d'immersion visuelle sur les performances d'apprentissage d'une procédure. Des mesures objectives ont été recueillies sur 4 essais. Ces mesures portent sur :

FIGURE 4.13 – Mise en saillance d'un flacon en rouge (ExpeI).

- le temps de réalisation de la tâche,
- le nombre de consultations des instructions,
- le nombre d'actions incorrectes.

4.2.3.1 Temps de réalisation de la tâche

Il s'agit du temps moyen mis pour réaliser la tâche, c'est-à-dire le temps temps d'exécution des actions ajouté au temps de consultation des instructions. Ce temps a été enregistré pour chaque essai. Au fur et à mesure des essais, il semble y avoir une diminution du temps de réalisation de la tâche (Tableau 4.1) indépendamment de la condition d'immersion visuelle. Le temps de réalisation de la tâche est élevé au premier essai et diminue au fil des répétitions. L'analyse des résultats « ANOVA » confirme que le temps de réalisation de la tâche varie en fonction des essais : $F(3,126) = 182.61$; $p < .001$. Les tests de comparaisons multiples permettent de montrer une diminution significative du temps de réalisation de la procédure pour ces essais (essai 1 contre essai 2 : $F(1,44) = 13.12$; $p < .001$; essai 2 contre essai 3 : $F(1,44) = 4.23$; $p < .001$; essai 3 contre essai 4 : $F(1,44) = 4.92$; $p < .001$). Le temps de réalisation de la tâche est donc très élevé au premier essai et diminue au fil des répétitions.

Tableau 4.1 – Temps moyen de réalisation de la tâche lors des 4 essais (en minutes) (ExpeI).

	Essai 1	**Essai 2**	**Essai 3**	**Essai 4**
m (σ)	6.23 (*1.33*)	4.68 (*1.05*)	4.18 (*1.06*)	3.59 (*0.76*)

La Figure 4.14 montre l'effet d'interaction entre les variables « Essai » et « Condition »

sur le temps de réalisation de la tâche lors des 4 essais.

Concernant l'effet de la condition d'immersion visuelle, les participants ayant utilisés le « CAVE » semblent mettre plus longtemps à réaliser la tâche (m = 4.92 min) que les participants de la condition « Oculus » (m = 4.72 min), qui eux-même mettaient plus de temps que ceux de la condition « Écran » (m = 4.36 min). Cependant, l'analyse de variance « ANOVA » ne révèle aucun effet significatif de la condition sur le temps de réalisation de la tâche : $F(2,42) = 1.20$.

L'interaction entre les variables « Essai » et « Condition » est significative : $F(6,126) = 4.94$; $p < .001$. Selon les essais, les résultats ne sont pas similaires. Lors de l'essai 1, le temps le plus court est enregistré pour la condition « Écran » (m = 5.48) et le temps le plus long pour la condition « CAVE » (m = 6.79). Les tests de comparaisons multiples montrent une différence tendanciellement significative pour l'essai 1 entre les conditions « Écran » et « CAVE » : $F(1,28) = 3.24$; $p < .1$.

Les tests de comparaisons multiples montrent des différences significatives pour toutes les conditions entre l'essai 1 et l'essai 2 (CAVE : $F(1,14) = 10.81$; $p < .001$; Oculus : $F(1,14) = 7.32$; $p < .001$; Écran : $F(1,14) = 4.60$; $p < .001$). Il existe également des différences pour la condition « Oculus » entre l'essai 2 et 3 : $F(1,14) = 3.89$; $p < .01$; et l'essai 3 et 4 : $F(1,14) = 3.50$; $p < .05$. Aucune autre différence n'est significative.

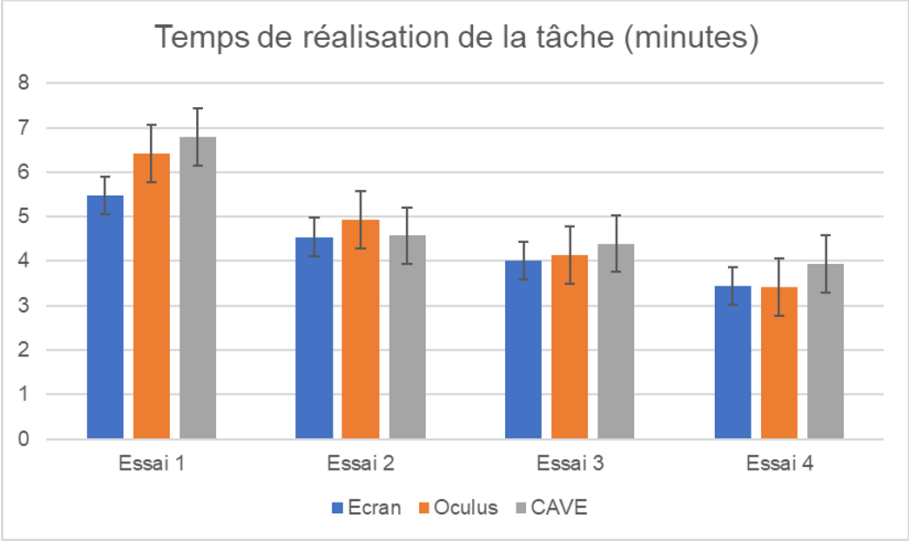


FIGURE 4.14 – Temps de réalisation de la tâche en fonction du dispositif d'immersion visuelle lors des 4 essais (en minutes) (`ExpeI`).

4.2.3.2 Nombre de consultations des instructions

Le nombre de consultations des instructions semble diminuer au fur et à mesure des essais (Tableau 4.2). Au fur et à mesure des essais, il semble y avoir une diminution du nombre de consultations des instructions indépendamment de la condition d'immersion visuelle. Le nombre de consultations des instructions est très élevé au premier essai et diminue au fil des répétitions. L'analyse des résultats « ANOVA » indique que l'effet de la

variable « Essai » sur le nombre de consultations des instructions est significatif : $F(3,126) = 261.08$; $p < .001$. Les tests de comparaisons multiples permettent de montrer un écart significatif entre tous les essais pris deux à deux (essai 1 contre essai 2 : $F(1,44) = 10.50$; $p < .001$; essai 2 contre essai 3 : $F(1,44) = 9.87$; $p < .001$; essai 3 contre essai 4 : $F(1,44) = 5.59$; $p < .001$).

Tableau 4.2 – Nombre de consultations des instructions lors des 4 essais (ExpeI).

	Essai 1	**Essai 2**	**Essai 3**	**Essai 4**
m (σ)	42.51 (*4.16*)	27.84 (*11.85*)	14.07 (*12.04*)	6.27 (*8.61*)

La Figure 4.15 montre l'effet d'interaction entre les variables « Essai » et « Condition » sur le nombre de consultations des instructions lors des 4 essais.

L'analyse de variance n'indique aucun effet significatif de la variable « Condition » sur le nombre consultations des instructions : $F(2,42) = 1.48$; et une absence d'interaction entre les variables « Essai » et « Condition » : $F < 1$.

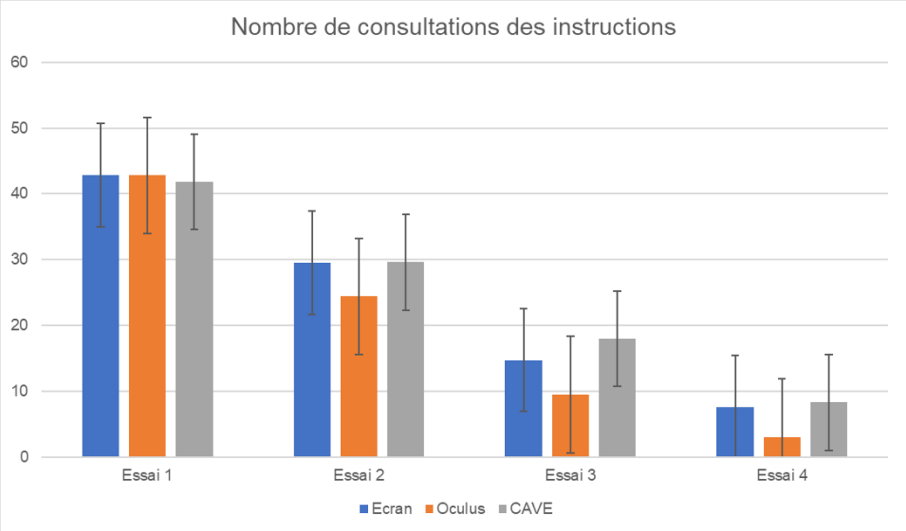


FIGURE 4.15 – Nombre de consultations des instructions pour les différentes conditions lors des 4 essais (ExpeI).

4.2.3.3 Nombre d'actions incorrectes

Le nombre d'actions incorrectes, telles que sélectionner le mauvais objet ou réaliser la mauvaise action, a été également enregistré. De manière générale, il semble qu'il y ait une fluctuation du nombre d'actions incorrectes au fur et à mesure des essais (Tableau 4.3).

La Figure 4.16 montre l'effet d'interaction entre les variables « Essai » et « Condition » sur le nombre moyen d'actions incorrectes lors des 4 essais.

Les groupes de participants ont un nombre moyen d'actions incorrectes assez similaire (Écran = 3.73 ; Oculus = 3.48 ; CAVE = 4.03). L'analyse des résultats « ANOVA »

Tableau 4.3 – Nombre d'actions incorrectes obtenu lors des 4 essais (ExpeI).

	Essai 1	**Essai 2**	**Essai 3**	**Essai 4**
m (σ)	1.68 (*2.48*)	4.02 (*4.10*)	5.18 (*4.38*)	4.11 (*4.69*)

indique que ce nombre ne varie pas en fonction de la condition : $F < 1$. Aucun autre effet n'est significatif. La Figure 4.16 représente le nombre d'actions incorrectes commises par les participants selon le dispositif utilisé lors des 4 essais.

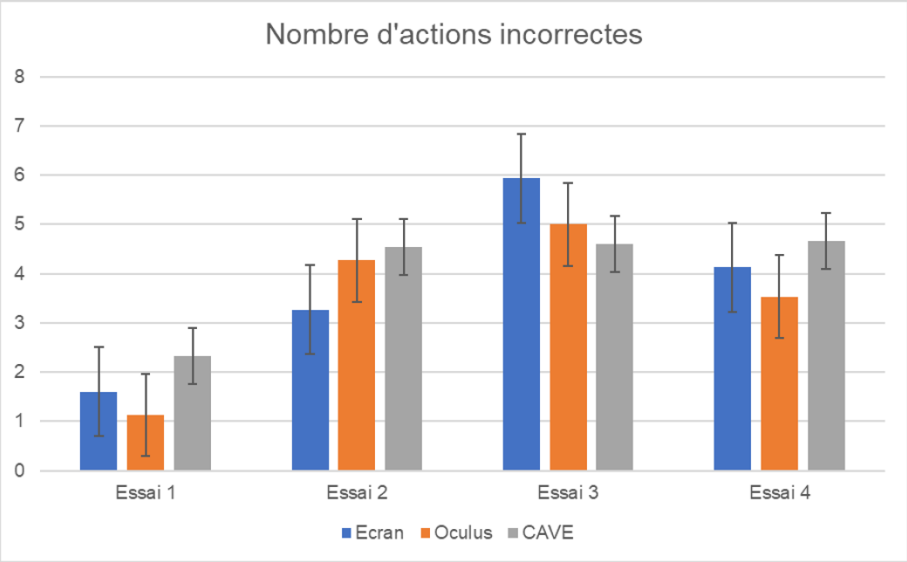


FIGURE 4.16 – Nombre d'actions incorrectes lors des 4 essais pour les trois dispositifs d'immersion visuelle (ExpeI).

4.2.4 Interprétations

Au fil des essais, il y a une diminution significative du temps de réalisation de la tâche et du nombre de consultations des instructions. Lors du premier essai, le temps et le nombre de consultations des instructions sont élevés, puis les performances enregistrées pour ces indicateurs diminuent progressivement, confirmant l'effet d'un apprentissage. La répétition de la procédure permet donc une amélioration des performances pour ces 2 indicateurs comportementaux. En revanche, les résultats obtenus quant aux nombres d'actions incorrectes sont assez inattendus. Lors de l'essai 1, les participants réalisent moins d'erreurs que l'essai 2 et 3. Il y a en effet une augmentation significative du nombre d'actions incorrectes commises par les participants entre l'essai 1 et l'essai 2. Au premier essai, le système fournit systématiquement des aides, ainsi les apprenants ont peu de risques de commettre des erreurs. Au deuxième essai, il n'y a plus d'aide systématique or les apprenants n'ont pas encore acquis toutes les actions de la procédure. Leur comportement est alors plus hasardeux et génère donc potentiellement plus d'erreurs.

Les dispositifs d'immersion visuelle n'exercent pas d'effet sur les indicateurs compor-

tementaux recueillis (temps total de réalisation de la tâche, nombre de consultations des instructions et nombre d'actions incorrectes). Néanmoins, l'interaction entre les variables « Condition » et « Essai » est significative concernant le temps total de réalisation de la procédure. Concernant le CAVE et l'écran, il y a une amélioration significative du temps total de réalisation de la tâche entre l'essai 1 et 2. Les participants ayant l'Oculus Rift ont, quant à eux, également une amélioration significative entre l'essai 1 et 2 puis, l'essai 2 et 3 et l'essai 3 et 4 en termes de vitesse de réalisation de la procédure. Enfin, concernant la comparaison entre les dispositifs essai par essai, il y a une différence tendanciellement significative lors du premier essai entre la condition « CAVE » et la condition « Écran », avec un temps de réalisation plus court pour la condition « Écran ». Les différences ne sont pas significatives pour les autres essais. Cela signifie que, quelque soit la condition immersive, les performances des apprenants s'améliorent au fur et à mesure des essais. Cependant, dans certaines conditions immersives, cette amélioration est plus marquée. Par exemple, les participants ayant utilisés l'Oculus Rift ont continué de progresser de manière significative pendant tous les essais. Tandis que, pour les participants ayant utilisés un écran ou le CAVE, leur progression est surtout marquée entre l'essai 1 et 2.

Pour conclure, nous remarquons que quelque soit le dispositif d'immersion visuelle utilisé, il y a apprentissage de la procédure. Comme précisé précédemment, nous souhaitons utiliser dans la deuxième expérience un dispositif de réalité virtuelle immersif. Nous pouvons donc en théorie utiliser un CAVE ou un casque de réalité virtuelle. Cependant, nous remarquons qu'en début d'expérience, le temps d'exécution peut être plus longs dans un CAVE qu'avec un casque. Ces résultats ont été confortés par une autre expérience que nous avons menée dans un contexte industriel avec la société Thalès. Dans cette étude les participants devaient manipuler des objets de petite taille (vis et écrou) qu'ils ont eu du mal à identifier du fait de la résolution graphique insuffisante du CAVE. De plus, nous observons dans l'expérience que nous venons de présenter, une tendance en faveur du casque sur le nombre de demandes d'aides par rapport au CAVE.

Nous considérons donc que cette procédure peut être apprise grâce à l'environnement virtuel présenté et qu'elle peut donc servir de support pour l'expérience II qui a pour objectif de valider notre modèle de tuteur adaptatif. Aux vues des performances comparées entre le casque et le CAVE, nous choisissons de mener cette seconde expérience à l'aide d'un casque de réalité virtuelle.

4.3 Expérience II

Dans la perspective de valider le modèle que nous avons développé et présenté au Chapitre 3, nous avons mené une deuxième expérience (`ExpeII`), destinée à examiner l'impact de la présence d'un tuteur virtuel adaptatif sur les performances d'un apprentissage de procédure. Il s'agissait de mesurer les performances d'apprentissage des participants lorsqu'ils interagissent avec un tuteur adaptatif, en termes de :

- temps de réalisation de la tâche,
- nombre de demandes d'aides,
- nombre d'actions incorrectes.

Nous avons émis l'hypothèse que la présence d'un tuteur qui adapte l'exécution du scénario pédagogique, en fonction des connaissances de l'apprenant et du nombre d'essai, améliorerait les performances d'apprentissage.

Hypothèse 1

La présence d'un tuteur adaptatif devrait améliorer les performances d'apprentissage.

Nous prenons en compte deux façons d'évaluer les performances d'apprentissage de l'apprenant. D'une part, nous considérons que l'adaptation du scénario pédagogique va permettre aux apprenants de ne plus demander de l'aide à l'agent ni de commettre des erreurs, plus rapidement. Nous appelons ce niveau de performance (0 demande d'aide et 0 erreur) une « performance de référence ». Ceci nous a conduit à émettre les deux sous-hypothèses suivantes :

Hypothèse 1.1

La présence d'un tuteur adaptatif devrait permettre d'atteindre les performances de références en un nombre moins élevé d'essais.

Hypothèse 1.2

La présence d'un tuteur adaptatif devrait permettre de réduire le temps d'exécution de la procédure, à partir du deuxième essai.

D'autre part, le deuxième type de performance que nous considérons, porte sur les interactions entre l'apprenant et le tuteur. Nous émettons ainsi deux autres sous-hypothèses :

Hypothèse 1.3

La présence d'un tuteur adaptatif devrait permettre de réduire le nombre de demandes d'aides à partir du deuxième essai.

Hypothèse 1.4

La présence d'un tuteur adaptatif devrait permettre de réduire le nombre d'actions incorrectes à partir du deuxième essai.

4.3.1 Participants

Quarante personnes (dont 6 femmes), âgées de 18 à 34 ans (âge moyen = 22.8, σ = 3.26) étaient volontaires et n'ont pas reçu de compensation pour participer à cette étude. L'échantillon était constitué d'étudiants (ENIB et UBO) et de chercheurs (IMT-Atlantique et IRT-BCOM). Le niveau d'étude des participants allait de Bac +1 à Bac +8. Nous avons demandé aux participants de répondre à un questionnaire afin de pouvoir contrôler des variables telles que la connaissance du matériel, de l'environnement virtuel ou encore la latéralisation et la pratique de jeux vidéo (Annexe G). Recueillir la latéralisation du participant permettait de configurer la manette utile, mais il n'y avait aucune différence concernant l'utilisabilité de la manette et l'interaction dans l'environnement virtuel. La connaissance de la procédure était un critère d'exclusion. Aucun des participants n'avait utilisé auparavant cet environnement virtuel et 90 % des participants avaient déjà vu un casque « *Oculus Rift* » et parmi ceux-ci, 40 % en avaient déjà utilisé un. Des groupes indépendants ont été constitués pour éviter les effets d'apprentissage. Les participants étaient répartis de manière équivalente au sein de deux conditions expérimentales, en préservant l'homogénéité des caractéristiques recueillies de telle manière que chaque groupe comprenait 20 participants (17 hommes et 3 femmes). Les participants ont donc été répartis dans deux conditions expérimentales :

- Condition adaptative : interaction avec un agent adaptatif.
- Condition non adaptative : interaction avec un agent non adaptatif.

L'absence de différences entre les deux groupes a été confirmée par le test de « Wilcoxon-Mann-Whitney » pour l'âge et la fréquence de pratique des jeux vidéo. Les deux groupes étaient similaires en termes d'âge (condition non adaptative : m = 22.0 ; condition adaptative : m = 22.5) et de fréquence de pratique des jeux vidéo (condition non adaptative : m = 1 ; condition adaptative : m = 1). La fréquence concernant la pratique des jeux vidéo a été recueillie à l'aide d'une affirmation présentée sous la forme d'une échelle de « Likert » en cinq degrés allant de « Jamais » à « Très fréquemment » (Annexe G). Quatre degré ont été explicité de manière plus concrète en se référant aux notions de « Semaine », « Mois » ou encore « Année » afin d'éviter des biais de compréhension de termes équivoques tels que « Fréquent » ou « Rarement » (Voir extrait ci-dessous).

Extrait du questionnaire (Annexe G) :

- ☐ Très fréquemment (plusieurs fois par semaine)
- ☐ Assez fréquemment (par exemple, une fois pas semaine)
- ☐ Rarement (par exemple, une fois par mois)
- ☐ Très rarement (par exemple une fois par an)
- ☐ Jamais

Des analyses statistiques, test de « Wilcoxon-Mann-Whitney », ont été réalisées pour examiner l'influence possible du profil du participant en tant que « Joueur » ou « Non Joueur » sur les mesures effectuées lors de l'apprentissage. Les participants classés dans la catégorie « Joueur » sont ceux ayant répondu qu'ils utilisaient une console soit « Très fréquemment » soit « Assez fréquemment ». Les participants « Non Joueur » correspondent donc à ceux ayant répondu qu'ils utilisaient une console « Rarement », « Très rarement » ou encore « Jamais ».

Dans cette étude, les groupes sont donc homogènes au niveau de la fréquence de pratique des jeux vidéo, ce qui permet de ne pas associer les variations possibles entre les conditions (non adaptative et adaptative) sur les mesures recueillies à ce facteur (i.e. au profil des participants en tant que joueur ou non).

4.3.2 Protocole expérimental

L'expérience se déroulait en passation individuelle. Chaque participant devait remplir un formulaire de consentement libre et éclairé (Annexe F) puis répondre à un questionnaire portant sur les caractéristiques individuelles (Annexe G). Pour chacune des conditions, l'expérience était divisée en 3 phases : une phase de démonstration, une phase de familiarisation et une phase de réalisation de l'expérience.

Des consignes standardisées étaient données à l'oral par l'expérimentateur (Annexe H pour la condition non adaptative et Annexe I pour la condition adaptative). La différence entre les deux consignes est dû au comportement du tuteur. La démonstration était réalisée par l'expérimentateur qui montrait l'utilisation du matériel et de l'environnement virtuel à l'aide d'une procédure conçue à cet effet. Il s'agissait de la procédure de déplacements de deux cubes utilisée dans la première expérience. Comme nous l'avons déjà dit, cette procédure reprenait les actions de base d'interaction avec l'EVAH, comme par exemple prendre un objet, cliquer sur un bouton, etc. (Figure 4.3).

La phase de familiarisation s'appuyait sur cette même procédure et permettait aux participants de s'approprier le dispositif de visualisation et d'interaction. Le but de cette phase était de se familiariser avec le matériel et l'environnement mais aussi avec la logique : demander l'instruction puis réaliser l'action. En d'autres termes, l'individu se familiarise avec le type de tâche qui sera demandée : la réalisation d'une procédure.

Il faut noter que nous ne familiarisons pas l'individu à la lecture systématique des instructions, mais seulement au cas sans lecture systématique des instructions, c'est-à-

dire de consultations libres. Ce choix provient du fait que, sinon, les participants auraient eu tendance à mémoriser davantage les instructions lors du premier essai de la phase expérimentale en se disant que la situation est identique à la phase de familiarisation. Dans la phase d'expérience, le fait que la lecture des instructions est systématique (essai 1) ou non (essai n) est annoncé dans les consignes standardisées (Annexes H et I). Les participants ne savent donc qu'une fois l'essai 1 effectué, que les essais suivants se feront avec une consultation libre des instructions.

En outre, le choix de mettre en place une lecture systématique des instructions lors du premier essai provient de résultats d'études antérieures montrant une forte consultation des instructions lors du premier essai. Pour Anderson (Anderson 1983), la première phase d'apprentissage (la phase déclarative) s'appuie fortement sur la consultation des instructions. De plus, il est préférable de fournir la ou les information(s) nécessaires à la réalisation de l'action avant que l'apprenant n'en ressente le besoin (Wickens 2002).

Il était possible de répéter plusieurs fois cette phase de familiarisation si le participant le souhaitait. Avant la phase de réalisation de l'expérience, l'expérimentateur rappelait les différentes touches utiles. Il s'assurait alors de la compréhension du participant puisque par la suite il ne pouvait plus intervenir pour l'aider dans la réalisation de la procédure. La consigne donnée aux participants était de réaliser la procédure dans son intégralité un certain nombre de fois. Pour les mêmes raisons que dans l'expérience I, les participants n'étaient également pas informés du nombre d'essais qu'ils allaient devoir réaliser (Ganier et al. 2013).

Lors de la phase expérimentale proprement dite, les participants devaient réaliser plusieurs essais de la procédure de lancement de tests de coagulation. Lorsqu'ils ne demandaient plus d'aide à l'agent et qu'ils ne commettaient plus d'actions incorrectes, ils arrêtaient de réaliser la procédure.

4.3.2.1 Matériels et système d'interaction

Le système d'interaction utilisé dans cette expérience était les manettes *Oculus Touch* du casque *Oculus Rift* (Figure 4.17). Le participant utilisait une manette *Oculus Touch* pour pointer l'objet qu'il voulait sélectionner. La position et l'orientation de la manette étaient obtenues à l'aide du système de captation de l'Oculus. Ce pointage était traduit dans l'environnement virtuel par un laser jaune avec un degré de transparence permettant une vision de l'objet pointé. Le pointeur changeait de couleur (vert) lorsqu'il rencontrait un objet préhensible, ce qui fournissait une aide préalable dans la visualisation des objets utiles lors de la procédure. Le pointeur répond donc à des critères ergonomiques d'incitation et de *feedback* immédiat (Bach 2004). Pour sélectionner un objet, le participant appuyait sur la gâchette avant de la manette. Il faisait alors apparaître un menu proposant la liste exhaustive des actions qu'il était possible de réaliser avec cet objet (Figure 4.19).

Dans ce menu, le participant passait d'une action à l'autre grâce au *thumbstick* de la manette en l'orientant vers le haut et le bas, et il sélectionnait celle qu'il souhaitait en appuyant de nouveau sur la gâchette avant (Figure 4.18). Les actions possibles sur l'objet

FIGURE 4.17 – Présentation du système d'interaction : la manette *Oculus Touch* pour *Oculus Rift* (ExpeII).

FIGURE 4.18 – Touches utiles pour interagir avec l'agent et l'environnement (ExpeII).

étaient listées par ordre alphabétique et un item « Fermer Menu » permettait d'annuler la sélection de l'objet. Dans les deux conditions, le participant était guidé par des instructions sonores délivrées dans un casque audio intégré au casque *Oculus Rift*. L'aide était accessible en pointant avec le même laser de la manette en direction de l'agent (niveau buste et tête). Dans ce cas, un menu d'aide s'affichait (Figure 4.20).

4.3.2.2 Performance de référence

Le but pour le participant étant d'apprendre la procédure de lancements de tests de coagulation sanguine, la passation s'arrêtait seulement lorsque le participant réalisait la procédure sans demande d'aide et sans action incorrecte. Un délai maximum de 45 minutes était alloué pour la passation de l'expérience (réalisation de différents essais de la procédure), avec des pauses entre chaque essai pour atténuer les effets possibles du casque de réalité virtuelle.

Avant de réaliser l'expérience proprement dite, des pré-tests et des pré-manipulations

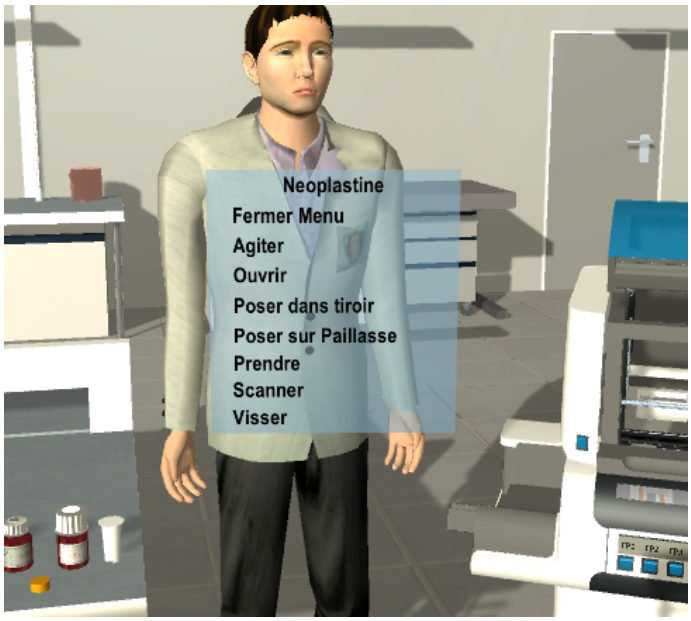


FIGURE 4.19 – Affichage du menu d'action après sélection d'un objet (ExpeII).

(a) Condition non adaptative.

(b) Condition adaptative.

FIGURE 4.20 – Affichage du menu d'aide dans les deux conditions (ExpeII).

ont servi à noter la faisabilité de la procédure et ainsi à établir une performance de référence équivalente à une performance « d'expert » de la procédure.

Concernant les mesures comportementales recueillies (temps de réalisation de la tâche, nombre de demandes d'aides et nombre d'actions incorrectes), il est attendu qu'elles diminuent au fur et à mesure des répétitions de la procédure. Par exemple, le nombre de demandes d'aides et d'actions incorrectes devraient être inexistant en fin d'apprentissage. Ces indicateurs refléteraient une récupération directe en mémoire de la procédure

sans devoir passer par une consultation des instructions comme c'est le cas pour des individus « novices » de la tâche (c'est-à-dire ayant peu ou pas de connaissances). Ces tests servaient à noter si des individus étaient capables de réaliser la procédure avec 0 aide et 0 erreur après un certain nombre de répétitions. Elles ont également servi à définir une plage concernant le nombre d'essais nécessaire pour atteindre cette performance mais aussi le temps de réalisation moyen. Ainsi, la performance de référence équivaut à 0 demande d'aide et 0 action incorrecte. Pour atteindre cette performance, il faudrait entre 4 et 10 essais à un participant et il lui faudrait environ 25 minutes. L'idée était donc de doubler ce temps pour déterminer une durée maximale de passation.

4.3.2.3 Conditions expérimentales

Les participants ont été répartis dans deux groupes. Le premier groupe interagissait avec un tuteur dont le comportement était non adaptatif et le second interagissait avec un tuteur ayant le comportement défini dans le Chapitre 3. Nous détaillons dans cette section ces deux comportements.

Le comportement du tuteur lors du premier essai était identique dans les deux conditions. Lors de cet essai, le tuteur annonçait systématiquement, avant l'action à réaliser, le nom de l'action, son but et l'objet à manipuler. L'objet était simultanément mis en évidence. La mise en évidence correspondait à faire clignoter l'objet en rouge et au pointage de cet objet par le tuteur (Figure 4.21). C'est en s'appuyant sur le principe d'aide prédictive de Wickens (Wickens 2002) qui recommande d'apporter le guidage utile à l'apprenant avant qu'il en fasse la demande, et sur le principe de signalement (Mayer 2009), que le choix d'une mise en saillance des objets concernés par l'action à réaliser a été fait.

Figure 4.21 – Mise en saillance d'un flacon (ExpeII).

Si le participant commettait une erreur, le tuteur lui indiquait que c'était une erreur et répétait l'instruction, exemple « Non c'est une erreur, Pour préparer les réactifs, ouvrir le flacon de solvant » et l'objet était mis en évidence. À tout moment, l'apprenant avait la

possibilité de demander de l'aide. Il n'avait accès qu'à un seul type d'aide, qui répétait l'instruction complète avec mise en saillance de l'objet à manipuler.

Comportement du tuteur dans la condition non adaptative

À partir du second essai, le tuteur ne faisait plus d'intervention systématique, mais il n'intervenait que sur demande d'aide de l'apprenant. Dans cette condition, l'apprenant n'avait accès qu'à un seul type d'aide « Aide ». Dans cette aide le tuteur annonçait l'action à réaliser, son but et l'objet à manipuler avec mise en saillance. En cas d'erreur de l'apprenant, le tuteur lui indiquait que c'était une erreur, répétait l'instruction et mettait en évidence l'objet.

Comportement du tuteur dans la condition adaptative

Dans cette condition le comportement du tuteur était celui présenté dans notre modèle au Chapitre 3. Lors du premier essai, ce comportement était inhibé pour s'assurer d'être exactement dans la même situation que dans la condition non adaptative. Cependant, le comportement du tuteur faisait tout de même les inférences sur le contenu des mémoires et sauvegardait l'état de la mémoire à long terme à la fin du premier essai. Cette sauvegarde était utilisée au chargement du deuxième essai, pour initialiser la mémoire à long terme du participant. À partir de ce deuxième essai, le comportement du tuteur était désinhibé. Étant donné que les participants étaient immergés à l'aide du casque de réalité virtuelle, le tuteur n'avait pas accès aux expressions faciales.

Quatre types d'aides différents étaient proposés aux participants via le menu 3D (Figure 4.20b) :

- Aide Action (AA)
- Aide But (AB)
- Aide Objet (AO)
- Aide (A)

Plus précisément, « Aide Action » correspondait à l'annonce de l'action à réaliser. L'« Aide But » était l'annonce du but de la sous-partie de la procédure que le participant était en train de réaliser. Par exemple, « Préparer les réactifs ». L'« Aide Objet » faisait référence à la mise en saillance rouge clignotante de l'objet à manipuler. Il s'agissait donc d'une aide simplement visuelle. Enfin, l'aide complète nommée « Aide » dans le menu, correspondait à l'ensemble des 3 aides citées précédemment. C'est-à-dire que l'agent virtuel annonçait le but et l'action à réaliser et pointait vers l'objet à manipuler, qui était également mis en saillance. Il est important de préciser qu'un participant pouvait demander, lors d'un même essai, plusieurs aides qu'elles soient similaires ou différentes. Il n'y avait aucune restriction. Il était donc possible également de sélectionner à la suite deux fois la même aide ou une aide différente si celle demandée précédemment n'était

pas suffisante pour restituer l'action.

Dans cette condition, les erreurs de l'apprenant ont été traitées comme présenté dans notre modèle au Chapitre 3.

4.3.3 Résultats

Les mesures comportementales étaient enregistrées par l'ordinateur. Les données recueillies sous forme de *logs* correspondaient aux mesures comportementales. Elles comprenaient le temps de réalisation de la tâche, le nombre de demandes d'aides ainsi que le nombre d'actions incorrectes commises.

Afin de réaliser des analyses statistiques pertinentes, nous avons différencié les erreurs liées à l'apprentissage des erreurs relevant d'un problème d'utilisabilité du logiciel, du système d'interaction ou de l'environnement virtuel. Ainsi, les données ont été revues et corrigées dans les deux conditions à partir de deux critères : (i) suppression des doublons correspondant à une erreur de sélection d'objet et (ii) suppression des erreurs de type sélection d'objets proches d'un point de vue de la localisation.

Pour l'analyse des données, des analyses de variances « ANOVA » et des tests de comparaisons multiples ont été réalisés sur 20 participants par condition. Pour cela nous avons utilisé le logiciel « R [2] ». Concernant l'analyse de variance « ANOVA mixte », le test de « Mauchly » a été utilisé pour tester l'homogénéité de variance et de covariance (sphéricité). Dans le cas où la sphéricité n'était pas assumée, une correction de « Greenhouse-Geisser » était appliquée. Il s'agit de tests statistiques robustes permettant de savoir s'il y a des différences significatives entre les conditions. Ces analyses ont porté essentiellement sur les 4 essais suivant la première exécution, qui pour rappel était identique pour l'ensemble des participants. Quelque soit la condition, tous les participants ont réalisé au moins 5 essais. Certains participants ont atteint les performances de référence (0 erreurs et 0 demande d'aide) à l'issue de ce cinquième essai. Ils n'ont donc pas eu besoin de faire d'essais supplémentaires. Nous réalisons donc les tests statistiques uniquement jusqu'à ce cinquième essai.

4.3.3.1 Nombre d'essais

Il s'agit du nombre d'essais nécessaire à l'apprenant pour atteindre le niveau de performance de référence sans prendre en compte le premier essai similaire dans les deux conditions. Dans la condition non adaptative, les participants ont mis en moyenne 5.75 essais (σ = 1.97) pour atteindre les performances de référence, alors que ceux de la condition adaptative ont mis 5.40 essais (σ = 1.10). L'analyse de variances ne révèle pas de différence significative entre les deux conditions.

Il faut noter que dans la condition adaptative, les participants pouvaient au minimum atteindre cette performance au quatrième essai étant donné que les essais précédents comportaient des interventions de l'agent virtuel considérées comme des aides. En effet, le comportement du tuteur, tel que présenté dans notre modèle, s'appuyait sur les

2. https://www.r-project.org/

niveaux de complexité d'encodage des instructions pour sélectionner ses interventions. Les quatre premiers niveaux de complexité nécessitent la communication d'informations supplémentaires à l'apprenant. C'est donc uniquement, au minimum après le quatrième essai que le tuteur peut arrêter d'intervenir et par conséquent que l'apprenant peut atteindre les performances de référence.

4.3.3.2 Temps de réalisation de la tâche

Le temps de réalisation de la tâche est indiqué en minutes et centièmes de minutes. Cet indicateur comportemental correspond à la somme du temps de consultation des instructions et du temps d'exécution des actions. Au fur et à mesure des essais, il y a une diminution significative du temps de réalisation de la tâche : $F(4,152) = 133.22$; $p < .001$. Les participants mettent en moyenne 4.53 minutes ($\sigma = 0.76$) lors du premier essai. Puis, lors du cinquième essai, la réalisation de la tâche est plus rapide (m = 2.82 minutes, $\sigma = 0.39$) (Tableau 4.4). Le temps est élevé lors de la première réalisation de la tâche, puis il diminue au fil des répétitions. Les tests de comparaisons par paires confirment la diminution significative de ce temps pour les essais pris deux à deux (essai 2 contre essai 3 : $F(1,39) = 7.29$; $p < .001$; essai 3 contre essai 4 : $F(1,39) = 9.22$; $p < .001$; essai 4 contre essai 5 : $F(1,39) = 4.46$; $p < .001$).

Tableau 4.4 – Temps moyen de réalisation de la tâche lors de 5 essais (en minutes) (ExpeII).

	Essai 1	**Essai 2**	**Essai 3**	**Essai 4**	**Essai 5**
m (σ)	4.53 (*0.76*)	4.32 (*0.85*)	3.80 (*0.65*)	3.14 (*0.46*)	2.82 (*0.39*)

La Figure 4.22 montre l'effet d'interaction entre les variables « Essai » et « Condition » sur le temps de réalisation de la tâche.

L'analyse de variance ne révèle pas de différences significatives entre les conditions expérimentales, mais elle montre une interaction entre les conditions et les différents essais : $F(4,152) = 18.45$; $p < .001$ (Figure 4.22). Tout d'abord, concernant la condition non adaptative, les tests de comparaisons multiples montrent des différences significatives entre l'essai 2 et 3, puis entre l'essai 3 et 4 : respectivement, $F(1,152) = 9.68$; $p < .001$ et $F(1,152) = 4.01$; $p < .005$. Il y a une diminution progressive du temps de réalisation de la tâche de l'essai 1 à 3 pour les participants de la condition non adaptative. Ensuite, pour la condition adaptative, il y a des différences significatives entre l'essai 1 et 2, puis l'essai 3 et 4 : $F(1,152) = 3.91$; $p < .01$ et $F(1,152) = 6.36$; $p < .001$, ainsi qu'une tendance entre l'essai 4 et 5b : $F(1,152) = 3.19$; $p = .053$. Il n'y a donc pas de diminution du temps de réalisation entre l'essai 2 et 3, elle n'apparait qu'à partir de l'essai 3. De plus, concernant l'essai 1, les analyses statistiques ne montrent pas de différence significative entre la condition non adaptative et la condition adaptative. Puis, à l'essai 2, il y a une différence significative entre les deux conditions, $F(1,38) = 5.72$; $p < .001$. Ce sont les participants de la condition adaptative qui réalisent la tâche le plus rapidement avec en

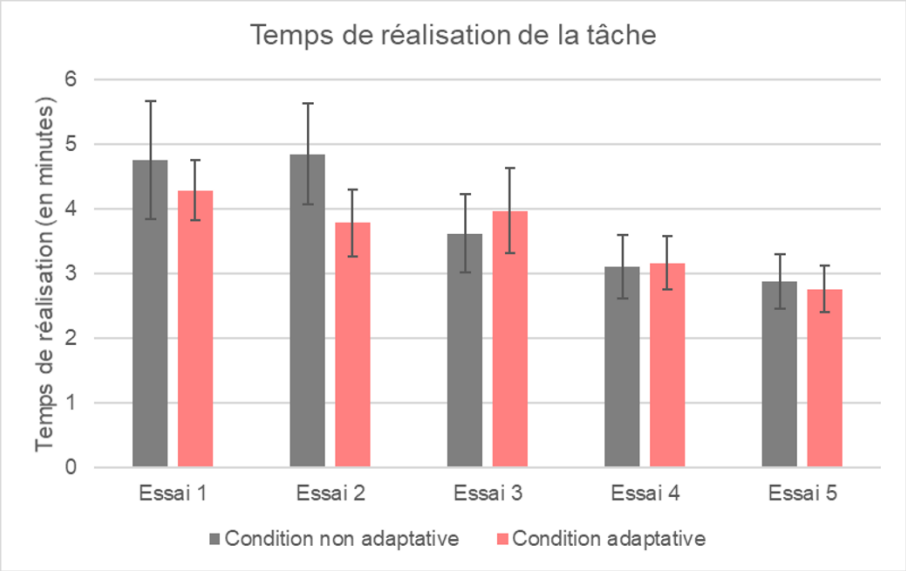


FIGURE 4.22 – Temps moyen de réalisation de la tâche lors de 5 essais par condition (en minutes) (`ExpeII`).

moyenne un temps d'environ 3.78 minutes (σ = 0.52), alors que ceux de la condition non adaptative mettent environ 4.85 minutes (σ = 0.78). Concernant les essais suivants, les tests de comparaisons multiples ne révèlent pas de différences significatives entre les conditions.

Pour rappel, le comportement du tuteur lors du premier essai était identique dans les deux conditions, et comme nous venons de le montrer, il n'y a pas de différence entre la condition adaptative et la condition non adaptative lors de cet essai en termes de temps de réalisation de la tâche. Ceci est également valable pour les autres mesures comportementales (nombre de demandes d'aides et des actions incorrectes). Ainsi, nous avons choisi, afin de simplifier la lecture des autres mesures comportementales, de ne pas présenter les résultats du premier essai.

4.3.3.3 Nombre de demandes d'aides

Il s'agit du nombre de demandes d'aides effectué par le participant et ayant abouti à une instruction sonore. En effet, les demandes étant réalisées via un menu 3D, ce nombre ne compte pas les apparitions de la fenêtre « Menu Aide » dans le cas où le participant a seulement appuyé sur « Fermer Menu ». Le nombre de demandes d'aides diminue de manière significative en fonction des essais, $F(3,114) = 19.73$; $p < .001$. Les participants demandent en moyenne 5.05 aides lors du deuxième essai (Tableau 4.5). À l'essai 5, les demandes d'aides sont inférieures à 1. Les tests de comparaisons multiples montrent un écart significatif entre tous les essais pris deux à deux, à l'exception de l'essai 4 et 5 (essai 2 contre essai 3 : $F(1,39) = 3.57$; $p < .01$ et essai 3 contre essai 4 : $F(1,39) = 2.70$; $p < .05$).

Tableau 4.5 – Nombre de demandes d'aides lors de 4 essais (ExpeII).

	Essai 2	**Essai 3**	**Essai 4**	**Essai 5**
m (σ)	5.05 (*8.32*)	2.60 (*4.10*)	0.75 (*1.51*)	0.33 (*0.86*)

L'analyse de variance révèle une différence significative selon la condition, $F(1,38) = 9.78$; $p < .01$. Les participants demandant le plus d'aides sont ceux de la condition non adaptative, avec en moyenne 3.68 demandes d'aides ($\sigma = 6.60$) contre 0.69 demandes d'aides ($\sigma = 1.71$) pour les participants de la condition adaptative.

Enfin, les tests statistiques révèlent une interaction significative entre les différents essais et les conditions, $F(3,114) = 23.47$; $p < .001$ (Figure 4.23). Tout d'abord concernant la condition non adaptative, les tests de comparaisons multiples montrent une différence significative entre l'essai 2 et 3, $F(1,19) = 7.07$; $p < .001$ mais pas entre les essais suivants. Il y a donc une diminution significative du nombre de demandes d'aides dans cette condition entre l'essai 2 et l'essai 3. Pour la condition adaptative, il n'y a pas de différences significatives entre les essais pris deux à deux, $F < 1$.

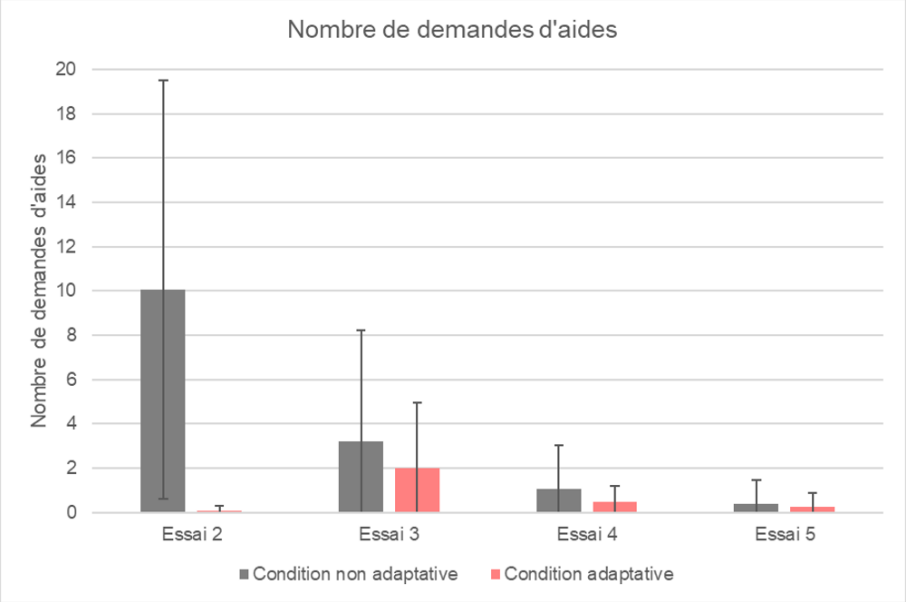


FIGURE 4.23 – Nombre de demandes d'aides par condition lors de 4 essais (ExpeII).

Puis, à l'essai 2, l'analyse révèle une différence significative entre la condition non adaptative et la condition adaptative, $F(1,38) = 7.86$; $p < .001$. Ce sont les participants de la condition adaptative qui réalisent la tâche en demandant le moins d'aides à l'essai 2 (m = 0.05, $\sigma = 0.22$). Les participants de la condition non adaptative demandent en moyenne 10.05 aides ($\sigma = 9.46$) lors de l'essai 2.

Concernant les essais suivants, les tests de comparaisons multiples ne révèlent pas de différences significatives entre les conditions. Il est possible de noter une légère augmentation du nombre de demandes d'aides à partir de l'essai 3 pour la condition adap-

tative (essai 3 : m = 2.00 (σ = 2.94) ; essai 4 : m = 0.45 (σ = 0.76) ; essai 5 : m = 0.25 (σ = 0.64)). Alors que pour les participants de la condition non adaptative, les demandes d'aides baissent fortement entre l'essai 2 et 3, passant de 10.05 (σ = 9.46) à 3.20 (σ = 5.01) demandes d'aides en moyenne.

Il est important de noter que certains participants cliquaient sur des objets puis parcouraient le menu offrant une liste d'actions exhaustives, ce qui pouvait leur fournir une aide. Par exemple, dans la condition non adaptative, si le participant sélectionnait le solvant à la place du desorbU, il pouvait remarquer que l'action « Scanner » n'était pas disponible et pouvait ainsi fermer ce menu et aller sélectionner le bon objet et la bonne action. Ainsi, de possibles différences entre les conditions peuvent être dues à cette possibilité d'affichage du menu d'action d'un objet dans la condition non adaptative qui est impossible dans la condition adaptative. En effet, lors de la sélection d'un mauvais objet, le menu d'action ne s'affichait pas et l'agent virtuel informait que ce n'était pas le bon objet. Cette différence entre les conditions est valable aussi bien pour le nombre de demandes d'aides que pour le nombre d'actions incorrectes.

Détail du nombre d'aides selon leur type

Comme nous l'avons déjà mentionné, quatre types d'aides étaient proposés aux participants de la condition adaptative : Aide Action (AA), Aide But (AB), Aide Objet (AO) et Aide (A). Pour la condition adaptative, nous avons réalisé des analyses statistiques sur les différences entre ces types d'aides. Il est possible de remarquer que les aides nommées « Aide But » et « Aide » n'ont pas été utilisés par les participants (Figure 4.24). L'analyse de variance met en évidence une différence significative entre les types d'aides demandées, $F(3,304) = 9.19$; $p < .001$. Les tests de comparaisons multiples montrent un écart significatif entre l'« Aide Action » et toutes les autres (AA contre A : $F(1,159) = 4.50$; $p < .001$; AA contre AB : $F(1,159) = 4.50$; $p < .001$; AA contre AO : $F(1,159) = 3.62$; $p < .01$). L'« Aide Action » est donc la plus demandée par les participants (m = 0.58, σ = 1.67).

En outre, l'analyse de variance révèle qu'il y a une interaction significative entre le type d'aides et les essais, $F(9,304) = 4.61$; $p < .001$ (Figure 4.24). Les tests de comparaisons multiples montrent des différences significatives concernant les demandes d'« Aide Action » entre l'essai 1 et l'essai 2, $F(1,80) = 6.97$; $p < .001$, puis entre l'essai 2 et l'essai 3, $F(1,79) = 5.37$; $p < .001$. Il y a une augmentation à l'essai 2 du nombre de demandes d'aides concernant l'action à réaliser (m = 1.75, σ = 2.95). Puis, à l'essai 3, ce nombre de demandes d'aides diminue (m = 0.4, σ = 0.75). Enfin, les analyses statistiques montrent également des différences entre l'« Aide Action » et les 3 autres types d'aides, mais ceci à l'essai 2 seulement (AA contre A : $F(1,79) = 6.97$; $p < .001$; AA contre AB : $F(1,79) = 6.97$; $p < .001$; AA contre AO : $F(1,79) = 5.97$; $p < .001$). Ainsi, à l'essai 2, c'est majoritairement l'« Aide Action » qui est utilisé par l'ensemble des participants de la condition adaptative. Les tests ne révèlent pas d'autres différences significatives.

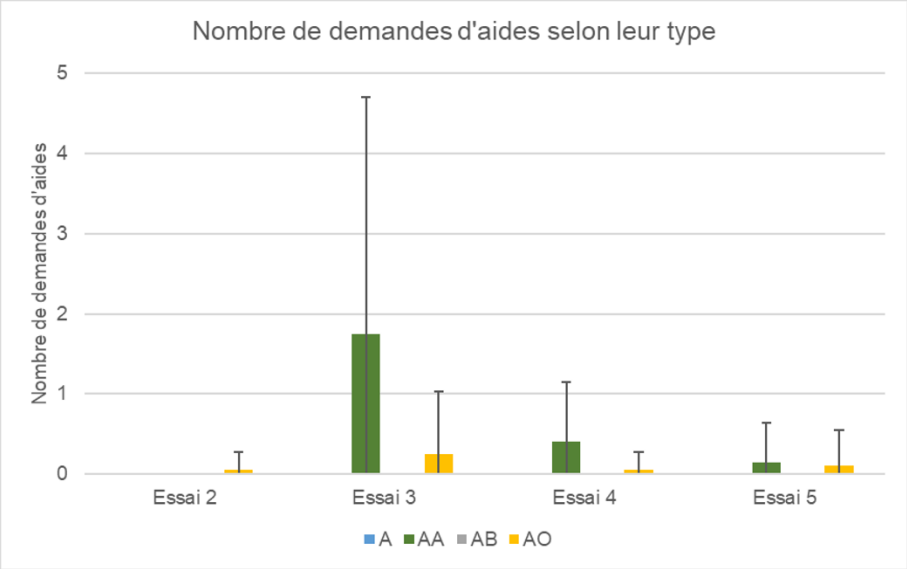


FIGURE 4.24 – Nombre de demandes d'aides selon leur type lors de 4 essais (ExpeII).

4.3.3.4 Nombre d'actions incorrectes

Le nombre d'actions incorrectes correspond au nombre de manipulations d'objets ne correspondant pas à l'instruction en cours et pour lesquelles le système renvoyait un message. Plus précisément, dans la condition non adaptative, une erreur correspond à la sélection de la mauvaise action, que ce soit sur le bon ou le mauvais objet. Dans la condition adaptative, une erreur peut correspondre soit à la sélection d'un mauvais objet, soit à la sélection d'une mauvaise action sur le bon objet, soit aux deux en même temps. Comme nous l'avons précédemment indiqué, les conditions sont plus strictes et les erreurs sont tout de suite identifiées, ne laissant pas la possibilité de s'aider du menu d'actions. En outre, dans cette condition, les *feedbacks* sont personnalisés. En effet, selon le type d'erreur (mauvais objet ou mauvaise action sur le bon objet) l'agent virtuel fournit un message sonore différent, permettant de préciser où se situe l'erreur. Alors que dans la condition non adaptative, le message d'erreur est toujours le même « Non c'est une erreur », ne donnant pas de précision sur ce qui n'est pas correct, vu que le comportement est non adaptatif. Ensuite l'agent annonce de nouveau à l'apprenant l'action qu'il doit réaliser, son but et l'objet à manipuler. En parallèle de cet énoncé, l'objet à manipuler est mis en saillance.

Tableau 4.6 – Nombre moyen d'actions incorrectes obtenu lors de 4 essais (ExpeII).

	Essai 2	**Essai 3**	**Essai 4**	**Essai 5**
m (σ)	3.98 (*3.75*)	4.23 (*2.43*)	3.08 (*2.03*)	1.80 (*1.77*)

De manière générale, il semble qu'il y ait une fluctuation du nombre d'erreurs au fur et à mesure des essais. En effet, il est possible de constater que le nombre d'erreurs entre l'essai 2 et l'essai 3 est sensiblement le même (essai 2 : m = 3.98 (σ = 3.75) ; essai

3 : m = 4.23 (σ = 2.43)). Puis, le nombre d'erreurs commises diminue au fur et à mesure des exécutions suivantes. L'analyse de variance montre une différence significative selon les essais, $F(3,114) = 12.42$; $p < .001$. Les tests de comparaisons multiples révèlent des différences significatives entre l'essai 3 et 4, $F(1,39) = 2.61$; $p < .05$, puis entre l'essai 4 et 5, $F(1,39) = 2.90$; $p < .05$.

Ensuite, l'analyse de variance met en évidence une différence significative entre les conditions pour le nombre d'erreurs commises. Les participants qui commettent le moins d'erreurs sont ceux de la condition adaptative avec en moyenne 2.75 erreurs (σ = 2.25), contre 3.79 erreurs pour la condition non adaptative (σ = 3.11).

Enfin, l'ANOVA montre une interaction significative entre les conditions et les essais pour le nombre d'erreurs, $F(3,114) = 16.68$; $p < .001$ (Figure 4.25). Les tests de comparaisons multiples montrent une différence significative entre l'essai 2 et 3 que ce soit pour la condition non adaptative, $F(1,19) = 3.70$; $p < .01$ ou la condition adaptative, $F(1,19) = 4.50$; $p < .001$. Il y a donc une légère hausse du nombre d'erreurs pour les participants de la condition adaptative, alors qu'il y a une diminution de ce nombre pour ceux de la condition non adaptative. Il n'y a pas de différences significatives entre les autres essais. Enfin, à l'essai 2, l'analyse révèle une différence significative entre la condition non adaptative et la condition adaptative, $F(1,38) = 6.58$; $p < .001$. Ce sont les participants de la condition adaptative qui réalisent la tâche en commettant le moins d'erreurs à l'essai 2 (m = 1.55, σ = 1.88). Les participants de la condition non adaptative commettent en moyenne 6.40 erreurs (σ = 3.60) lors de l'essai 2. Concernant les essais suivants, les tests de comparaisons multiples ne révèlent pas de différences significatives entre les conditions.

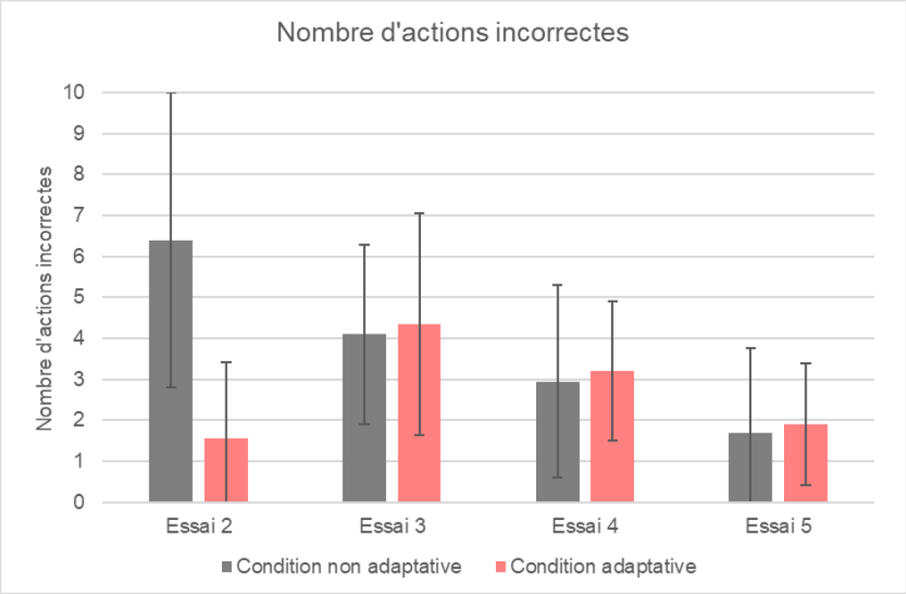


FIGURE 4.25 – Nombre moyen d'actions incorrectes lors de 4 essais par condition (`ExpeII`).

Détail du nombre d'erreurs selon leur type

Dans les mesures recueillies, deux types d'erreurs étaient différenciés pour la condition adaptative, notamment en raison des *feedbacks* qu'elles engendraient. En d'autres termes, il était facile de différencier ces erreurs, puisque l'agent virtuel corrigeait le participant de manière différente et à des moments différents selon le type d'erreur. En effet, une erreur de type objet « Erreur Objet », c'est-à-dire une mauvaise sélection d'un objet avait pour conséquence le non-affichage du menu 3D et l'agent virtuel annonçait que c'était le mauvais objet, puis identifiait le bon objet (localisation et nomination). Par exemple « ceci est le solvant et non le panier ». La deuxième erreur identifiée était de type « Erreur Action ». Le participant après avoir sélectionné le bon objet avait accès à un menu 3D livrant la liste exhaustive des actions à réaliser sur cet objet. Il pouvait donc également se tromper à ce moment-là et l'agent virtuel lui annonçait alors que c'était bien le bon objet et l'action correcte à réaliser. Par exemple : « ceci est bien le solvant, mais tu dois le prendre ».

Les analyses statistiques se sont centrées sur les différences entre les types d'erreurs de façon générale, puis entre les 4 essais. L'analyse de variance montre une différence significative entre les types d'erreurs commises, $F(1,152) = 53.35$; $p < .001$. Les participants de la condition adaptative commettent plus d'« Erreur Objet » (m = 2.13, σ = 1.80) que d'« Erreur Action » (m = 0.63, σ = 0.92).

FIGURE 4.26 – Nombre d'erreurs selon leur type lors de 4 essais (`ExpeII`).

De plus, l'analyse de variance montre un effet d'interaction significatif entre les variables essais et types d'erreurs, $F(3,152) = 3.53$; $p < .05$. Tout d'abord, les tests de comparaisons multiples montrent une différence significative entre l'essai 2 et l'essai 3 pour les erreurs de type « Erreur Objet », $F(1,39) = 5.24$; $p < .001$. Les participants commettent plus d'erreurs de type « Erreur Objet » à l'essai 3 (m = 3.20, σ = 1.79) par rapport

à l'essai 2 (m = 1.05, σ = 1.64). Ensuite, les tests révèlent des différences significatives entre les types d'erreurs à l'essai 3, $F(1,39) = 4.99$; $p < .001$ et à l'essai 4, $F(1,39) = 5.36$; $p < .001$. Il y a également une différence tendanciellement significative à l'essai 5 entre les types d'erreurs, $F(1,39) = 2.92$; $p < .08$. Lors de chacun de ces essais, c'est une erreur de type « Erreur Objet » qui est commise un nombre de fois plus important qu'une erreur de type « Erreur Action » (Figure 4.26). Il n'y a pas d'autres différences significatives.

4.3.3.5 Nombre d'interventions de l'agent virtuel

D'après nos hypothèses formulées à partir du modèle de la mémoire humaine (Atkinson et Shiffrin 1968) et de l'apprentissage d'habiletés complexes (Anderson 1983 ; Anderson 2014), nous avons caractérisé les interventions que l'agent virtuel réalise dans le cadre du comportement que nous avons défini au Chapitre 3. En effet, l'agent virtuel interviendrait également si le participant restait inactif pendant un certain temps (8 secondes). L'analyse de ces interventions est tout aussi importante que celles concernant les mesures recueillies sur le comportement de l'apprenant, puisqu'elles font partie intégrante de la condition adaptative. Elles permettent donc de se positionner quant à la validité du modèle et à la pertinence des hypothèses formulées.

Une analyse de variance a été réalisée concernant le nombre d'interventions de l'agent virtuel, c'est-à-dire des interventions préalablement définies par le modèle et celles illustrant une inaction du participant. L'analyse de variance révèle qu'il y a une différence significative des interventions selon les essais, $F(3,76) = 11081$; $p < .001$. Les tests de comparaisons multiples montrent des différences significatives entre l'essai 2 et l'essai 3, $F(1,39) = 120.29$; $p < .001$, ainsi qu'entre l'essai 3 et l'essai 4 $F(1,39) = 23.27$; $p < .001$ et entre l'essai 4 et l'essai 5, $F(1,39) = 23.96$; $p < .001$. Il y a donc une diminution des interventions de l'essai 2 à 3, 3 à 4 et 4 à 5.

Dans le but de savoir si l'agent virtuel fournit suffisamment ou non d'aides avant que l'apprenant soit en difficulté (i.e. inactif), nous avons réalisé une comparaison entre deux types d'interventions possibles : les interventions dues au comportement du tuteur et les interventions dues à une inaction.

Interventions dues au comportement du tuteur

Ce type d'intervention concerne les interventions de l'agent virtuel. Nous n'avons pas effectué d'analyses statistiques pour ce type d'intervention puisque les résultats de la Figure 4.27 montrent que le comportement effectif du tuteur lors des expériences est identique au comportement nominal défini dans le Chapitre 3. Ce que l'on peut observer par le nombre d'« Intervention » en bleu sur la Figure 4.27. Nous détaillons ici les interventions dues au comportement nominal du tuteur. Tout d'abord, à l'essai 2, il y a 36 interventions de l'agent virtuel prévues dont 5 interventions complètes (la procédure est composée de 5 buts à atteindre). Ce type d'intervention reprenait le but et l'action à effectuer avec mise en saillance de l'objet concerné. Il y a également 20 interventions portant sur des actions et 11 interventions sur des actions et les objets correspondant mis en saillance.

Ces interventions apparaissent lors de la première nomination d'un objet dans cet essai. Autrement dit, dès que le nom de cet objet est répété, donc dès qu'une deuxième action doit être effectuée sur cet objet, il n'est pas mis en rouge clignotant. À l'essai 3, les buts et la mise en saillance ne font plus partie des interventions définies. Il y a seulement des interventions sur les actions qui sont situées aux actions « frontières », i.e. délimitant un but d'une sous-partie de la procédure. En d'autres termes, il s'agit de la première et de la dernière action d'une sous-partie. À l'essai 4, seul les 5 buts sont donnés aux participants. Enfin, à l'essai 5, l'agent virtuel n'intervient plus sauf en cas d'inaction ou de demande d'aide.

Interventions dues aux inactions

Concernant les interventions dues aux inactions du participant, elles pouvaient être de trois types : Intervention Inaction Objet, Intervention Inaction Action et Intervention Inaction But. Les « Intervention Inaction Objet » correspondent à la mise en saillance de l'objet sur lequel doit être réalisé l'action en cours. Elles apparaissent dans un délai de 8 secondes si le participant ne fait rien (c'est-à-dire, ne demande pas d'aide ou ne sélectionne pas un objet). Les « Intervention Inaction Action » correspondent à l'annonce de l'action à réaliser et apparaissent 8 secondes après les « Intervention Inaction Objet », si le participant n'a toujours rien fait. Les « Intervention Inaction But » correspondent à l'énoncé de l'action et du but et apparaissent 8 secondes après les « Intervention Inaction Action », si le participant n'a toujours rien fait. Lors des 4 essais de cette procédure de lancements de tests de coagulation sanguine, il n'y a eu aucune « Intervention Inaction Action » ni « Intervention Inaction But ». Tandis que les « Intervention Inaction Objet », il y en a eu en moyenne 0.33.

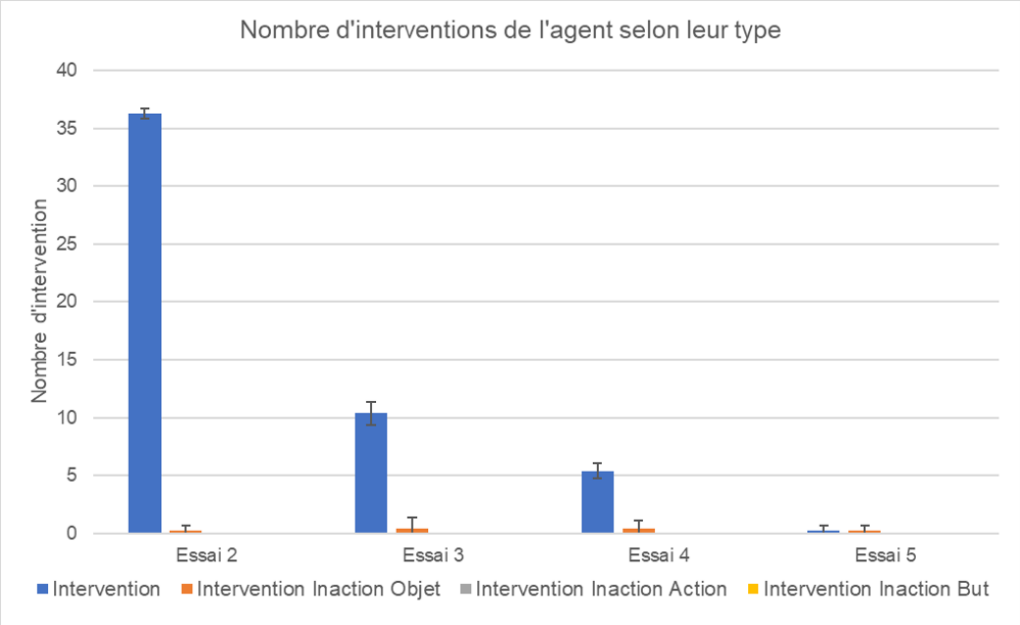


FIGURE 4.27 – Nombre d'interventions de l'agent virtuel lors de 4 essais (`ExpeII`).

4.3.4 Interprétations

Notre Hypothèse 1.1 est que la présence d'un tuteur adaptatif devrait permettre de réduire le nombre d'essais pour atteindre les performances de référence. Les résultats présentés en Section 4.3.3.1 montrent qu'il n'y a pas de différence significative sur le nombre d'essais entre la condition non adaptative et la condition adaptative. Notre hypothèse n'est donc pas vérifiée. Cependant, nous remarquons un nombre moyen d'essais inférieur pour la condition adaptative, ce qui marque une certaine tendance.

En ce qui concerne le temps de réalisation de la tâche, nous observons une réduction au fur et à mesure des essais quelque soit la condition. Ceci tend à démontrer que dans les deux conditions il y a apprentissage de la procédure. Il n'y a pas de différence significative sur les temps de réalisation entre les deux conditions. Cependant, il y a une différence significative sur le temps de réalisation lors du deuxième essai. Les participants de la condition adaptative mettent significativement moins de temps à réaliser la procédure que les participants de la condition non adaptative. Ceci signifie que les interventions calculées par notre comportement de tuteur sont pertinentes et valident donc partiellement l'Hypothèse 1.2.

Au delà du nombre d'essai et du temps pour atteindre les performances de référence, nous avons également souhaité mesurer les performances liées aux interactions entre l'apprenant et le tuteur. La première mesure que nous avons effectuée concerne le nombre de demandes d'aides. Les résultats présentés en Section 4.3.3.3 montrent une différence significative sur le nombre de demandes d'aides sur la totalité des essais. Les participants de la condition adaptative ont significativement demandé moins d'aide que les participants de la condition non adaptative. Ceci signifie encore que les interventions du tuteur adaptatif sont pertinentes, ce qui valide l'Hypothèse 1.3. L'étude détaillée sur les types d'aides demandés montre que les participants de la condition adaptative, n'ont jamais demandé d'aide portant sur le but des actions mais uniquement sur les actions et les objets. Ces interventions sont minimes, à l'exception de l'essai 3, dans lequel les participants de la condition adaptative ont demandés un peu plus d'aides sur l'action à réaliser. Cela signifie que certaines sous-parties de la procédure ont été mémorisées, mais pas toutes. Il serait intéressant de mener des études plus approfondies afin d'identifier dans quel contexte les demandes d'aides ont été réalisées et ainsi d'améliorer le comportement de notre tuteur.

Enfin, nous avons également mesuré le nombre d'erreurs commises par les participants. D'après les résultats présentés en Section 4.3.3.4, ce nombre diminue au fur et à mesure des essais et tend à valider que la procédure est acquise. Nous observons une différence significative entre les conditions sur le nombre d'erreurs commises. Les participants de la condition adaptative commettent significativement moins d'erreurs que les participants de la condition non adaptative. Ceci valide notre Hypothèse 1.4.

Ces résultats montrent que le comportement de tuteur que nous avons défini améliore les performances d'apprentissage. Nous considérons que les participants font moins d'erreurs et moins de demandes d'aides quand ils interagissent avec ce tuteur. Nous avons également examiné si le participant restait actif lors de l'apprentissage. Nous avons observé très peu d'interventions du tuteur sur l'inaction du participant. Lorsque le tuteur intervenait, une seule intervention était nécessaire (mise en évidence de l'objet à manipuler) pour que l'apprenant se rappelle de l'action à réaliser.

4.4 Bilan

Dans ce chapitre nous avons voulu valider l'impact du comportement du tuteur, tel que nous l'avons défini dans le Chapitre 3 pour l'apprentissage d'une procédure en environnement virtuel.

Avant de valider notre modèle, nous avons réalisé une première expérience (Section 4.2) ayant comme objectif de s'assurer que la procédure pouvait réellement être apprise en environnent virtuel et d'identifier la configuration de réalité virtuelle la plus adaptée à cet apprentissage. Les résultats de cette expérience ont montré que cette procédure pouvait être apprise. Les résultats nous ont également permis d'identifier que le casque de réalité virtuelle était la configuration immersive la plus adaptée. C'est donc dans ces conditions que nous avons réalisé notre deuxième expérience pour valider notre modèle.

Dans la seconde expérience, nous avons comparé les performances d'apprentissage de la procédure entre une condition avec notre tuteur adaptatif et une condition avec un scénario pédagogique figé. Les données recueillies permettent de conclure quant à l'efficacité de notre modèle pour un apprentissage de procédure.

III

Conclusion

5 Conclusion et Perspectives

Dans ce travail de recherche, nous avons présenté un modèle de représentation de la mémoire de l'apprenant pour les systèmes tutoriels intelligents et adaptatifs. Dans ce chapitre, nous faisons un bilan de ce travail et nous ouvrons de nouvelles perspectives.

Sommaire

5.1 Conclusion

Dans ce mémoire, nous avons présenté notre modèle MEMORIA. La contribution principale de ce modèle réside dans une formalisation et une implémentation du modèle de l'apprenant sous forme de mémoires qui stockent les informations perçues par l'apprenant dans un environnement virtuel et les instructions émises par le tuteurs.

La conception de notre modèle est basée sur les systèmes tutoriels intelligents, qui sont classiquement composés de quatre composantes : le modèle du domaine, le modèle de l'interface, le modèle de l'apprenant et le modèle du tuteur.

Le modèle du domaine est représenté par les connaissances métiers formalisées à l'aide du méta-modèle MASCARET. Afin de rendre naturelles les interactions entre le tuteur et l'apprenant, nous avons choisi de représenter le modèle de l'interface par l'intermédiaire d'un agent conversationnel animé à l'aide de la plate-forme GRETA. Le modèle de l'apprenant est constitué de l'ensemble des connaissances acquises par l'apprenant en cours de simulation. Ces connaissances sont organisées dans trois mémoires : la mémoire sensorielle, la mémoire de travail et la mémoire à long terme. L'enjeu majeur de

cette thèse porte sur la formalisation de l'encodage des informations dans ces mémoires, ainsi que le flux de données entre celles-ci. Cette formalisation est basée sur la théorie de la mémoire humaine proposée par Atkinson et Shiffrin et inspirée de l'architecture cognitive ACT-R. Le modèle de tuteur que nous proposons est centré sur la réalisation d'un comportement qui adapte l'exécution du scénario en fonction des connaissances de l'apprenant et de ses interactions avec le tuteur.

Dans l'étude expérimentale que nous avons menée pour valider ce modèle, nous avons émis l'hypothèse générale suivante : le comportement adaptatif devrait permettre une amélioration des performances d'apprentissage de l'apprenant, en termes de temps de réalisation de la tâche, du nombre de demandes d'aides et du nombre d'actions incorrectes. Nous avons comparé deux groupes de participants. Dans le premier groupe, nous avons intégré un tuteur adaptatif utilisant notre modèle MEMORIA, qui adapte l'exécution du scénario pédagogique et dans le second groupe, un tuteur non adaptatif qui réalise un scénario pédagogique figé. Les participants de la condition adaptative mettent significativement moins de temps à réaliser la procédure que les participants de la condition non adaptative. De plus, les participants de la condition adaptative ont significativement demandé moins d'aide que les participants de la condition non adaptative. Ceci signifie que les interventions du tuteur adaptatif sont pertinentes. Enfin, les participants de la condition adaptative commettent significativement moins d'erreurs que les participants de la condition non adaptative. L'ensemble de ces résultats valide notre hypothèse générale.

Les travaux que nous avons réalisé sont en cours d'application dans le cadre d'une prestation avec un industriel, équipementier dans le domaine militaire et aéronautique. L'objet de cette prestation est la conception d'un outil de formation aux procédures de maintenance sur des systèmes militaires tels que présentés dans la Figure 5.1. Plusieurs scénarios pédagogiques ont été définis et notre modèle est intégré pour adapter l'exécution de ces scénarios en fonction de l'évolution des apprenants (maintenanciers aéronautiques).

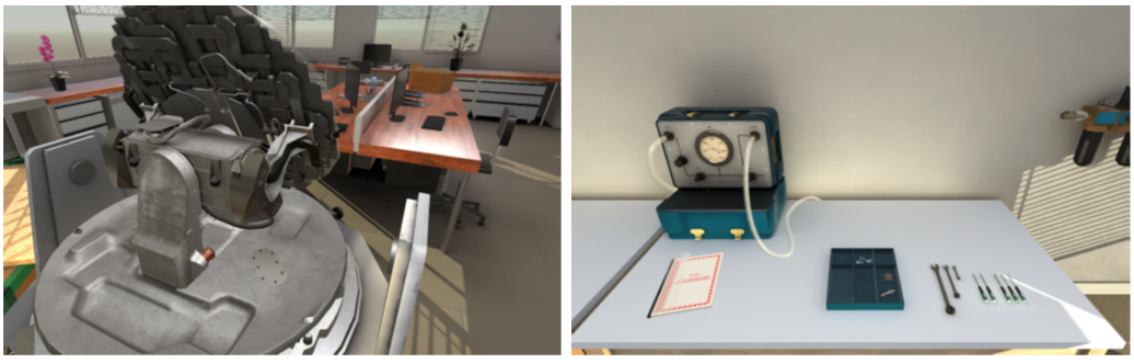


FIGURE 5.1 – Exemple de situation pédagogique pour la maintenance aéronautique.

5.2 Perspectives

Nous avons pu valider notre modèle par une expérience et par son intégration dans une application industriel concrète. Cependant, nous identifions dans cette section des perspectives à nos travaux. Dans un premier temps, nous identifions des pistes d'amélioration et d'extension de notre modèle, puis nous proposons de l'appliquer à un autre domaine que la formation.

5.2.1 Extension de notre modèle

Dans le modèle de l'apprenant que nous avons proposé, nous nous sommes appuyés sur deux hypothèses en psychologie cognitive. La première hypothèse porte sur la notion de *chunking* qui regroupe des informations de plus en plus complexes au fur et à mesure des répétitions au cours de l'apprentissage. Puisque notre sujet de recherche concerne l'apprentissage de procédure, nous avons proposé des niveaux de complexité pour l'encodage des instructions et des objets à manipuler. À notre connaissance, il n'existe pas de travaux dans la littérature qui permettent de valider cette hypothèse. Nous proposons donc comme première perspective de collaborer avec des collègues en psychologie cognitive afin de définir des protocoles expérimentaux pour étudier plus précisément la hiérarchisation des connaissances liées aux instructions dans l'apprentissage de procédure. Les résultats obtenus par ces études pourront facilement être intégrés dans notre modèle, parce que nous nous appuyons sur une représentation formelle de ces instructions.

La deuxième hypothèse en psychologie cognitive sur laquelle nous nous sommes appuyés pour concevoir notre modèle, concerne le transfert et la récupération des informations en mémoire à long terme. Nous avons considéré que lorsque l'apprenant réalise correctement l'action, alors il y a transfert des informations portant sur cette action de la mémoire de travail vers la mémoire à long terme. De plus, lorsque le tuteur annonce une instruction, les informations de la mémoire à long terme portant sur cette instruction sont automatiquement transférées dans la mémoire de travail. Ces hypothèses restent encore à valider. Ce travail est à réaliser par des collègues en psychologie cognitive mais notre architecture pourrait servir pour la construction des protocoles expérimentaux.

Dans le modèle MEMORIA, nous avons essentiellement travaillé sur les connaissances procédurales. Les seules connaissances déclaratives que nous avons prises en compte dans notre modèle de mémoire sont les connaissances liées au nom et à la position des objets. Le méta-modèle MASCARET permet de représenter toutes les connaissances déclaratives autour d'un métier, par exemple la sémantique d'une propriété d'un composant, la description d'un rôle dans une procédure, etc. Afin de permettre à l'apprenant d'acquérir de compétences supérieures lui permettant de raisonner par lui-même sur le système et les procédures, il serait intéressant de pouvoir en tenir compte dans notre modèle. Ceci impliquerait de définir une formalisation spécifique de ces informations pour leur enco-

dage dans les différentes mémoires.

Nous avons proposé d'incarner le tuteur par un agent conversationnel animé afin de rendre plus naturelles leurs interactions. L'apprenant et le tuteur sont par exemple capables de poser des questions l'un à l'autre et d'y répondre. Cependant, nous n'avons pas pris en compte cette capacité dans le comportement adaptatif du tuteur. Par exemple, l'apprenant peut poser une question au tuteur. Le tuteur peut lui répondre et selon sa réponse, celle-ci sera encodée dans la mémoire de l'apprenant. Par contre, le fait même que l'apprenant ait posé une question n'est pas pris en compte dans le comportement du tuteur. À l'inverse, le tuteur fait des inférences sur le contenu de la mémoire de l'apprenant, mais ceux ne sont que des inférences, le tuteur pourrait en cas de doute poser une question à l'apprenant afin de les confirmer. Il serait intéressant de prendre en compte cette interaction bidirectionnelle dans le comportement du tuteur. Ce comportement peut devenir de plus en plus complexe et il est actuellement conçu par un informaticien. Afin de respecter la démarche de MASCARET, qui propose d'expliciter les connaissances liées à la pédagogie, pour que le formateur puisse lui-même concevoir les situations pédagogiques, nous proposons de rendre explicite l'ensemble des éléments sur lesquels s'appuient le comportement du tuteur. Ceci permettra au formateur de concevoir lui-même le comportement du tuteur à l'aide du langage UML utilisé dans MASCARET.

5.2.2 Autre domaine d'application

Notre modèle peut être appliqué dans le cadre du projet VITAAL [1] (Vaincre l'Isolement par les TIC pour l'« Ambient Assisted Living »). L'objectif de ce projet est de lutter contre l'isolement des personnes âgées grâce à l'exploitation des nouvelles technologies et des services du numérique dans un contexte de santé et de bien-être. Le projet s'intéresse en particulier à l'isolement de personnes âgées, handicapées, ou souffrant de pathologies diverses et qui sont éloignées des centres de soins. Les outils numériques sont alors destinés à améliorer le confort, à virtualiser une relation qui n'est pas possible physiquement et à permettre les soins ou l'intervention de personnels de santé à distance.

La maladie d'Alzheimer est une des pathologies qui peut toucher ces personnes. Le nombre de personnes touchées par ces troubles est de plus en plus important. Selon France Alzheimer [2], ce nombre pourrait atteindre 1 275 000 français en 2020, et 3 millions de personnes si l'on prend en compte les proches aidants. Cela représente donc un enjeu majeur à étudier. « La maladie d'Alzheimer est un type de démence qui provoque des troubles de la mémoire, de la pensée et du comportement » [3]. Plus précisément, cette maladie a un impact et une évolution progressive sur les différentes mémoires et le comportement des personnes touchées. Cet impact affecte leur vie quotidienne et leur

1. https://www.imt-atlantique.fr/fr/rechercheihsevhaalprojetsvitaal
2. https://www.francealzheimer.org/
3. https://www.alz.org/fr/quest-ce-que-la-maladie-d-alzheimer.asp

autonomie fonctionnelle (McKhann et al. 2011).

Des débats scientifiques sont en cours par rapport à la prise en charge des personnes souffrant de démence de type Alzheimer. Ce débat tourne autour de deux solutions. La premier solution, est une approche biomédicale qui propose un prise en charge médicamenteuses, alors que la seconde solution, consiste en une approche psychosociale de l'accompagnement de ces personnes (Van der Linden et Van der Linden 2015).

Dans ce contexte, notre modèle, permettant l'explicitation du contenu de la mémoire humaine, pourrait être appliqué à l'étude et à la remédiation de cette maladie, ce qui rejoint l'approche psychosociale.

Des travaux ont été menés utilisant la réalité virtuelle (Figure 5.2) pour le ré-apprentissage des activités de la vie quotidienne telles que préparer du café (Foloppe et al. 2018). Un extrait d'une telle activité est présenté ci-dessous. Classiquement, c'est ce type d'activité qui devient difficile au fur et à mesure de la progression de la maladie.

Extrait de la liste d'actions à réaliser utilisée dans (Foloppe et al. 2018) :
Coffee task : make a coffee using the coffee machine and serve a cup of coffee with sugar.

1. Open the lid of the coffee-maker
2. Put the coffee pot under the tap
3. Turn on the tap
4. Turn off the tap
5. Put the water of the coffee pot in the coffee-maker
6. Put the coffee pot in its place
7. Put a coffee filter in the coffee-maker
8. Put the ground coffee in the filter
9. Close the lid of the coffee-maker
10. Press the red button
11. Wait for the machine to stop
12. Put some coffee into the coffee cup
13. Put a sugar lump into the coffee cup

Les activités de la vie quotidienne sont souvent formalisées sous forme de procédure manipulant des objets du monde réel. Ce formalisme est compatible avec le méta-modèle Mascaret que nous avons utilisé pour représenter les procédures sur des systèmes industriels. Notre modèle Memoria pourrait donc être utilisé de deux manières dans le cadre du projet VITAAL et plus particulièrement à propos de la maladie d'Alzheimer.

Tout d'abord, notre modèle peut servir à réaliser des expériences étudiants les mécanismes mnésiques qui feraient défauts dans cette maladie. Ces situations expérimentales pourraient avoir lieu dans des situations de réalité virtuelle simulant ces activités quotidiennes, ce qui permettrait d'obtenir des résultats scientifiques plus écologiques et potentiellement plus valides.

FIGURE 5.2 – Environnement virtuel pour le ré-apprentissage d'une activité de la vie quotidienne.

Ensuite, notre modèle pourrait également servir, non plus comme un tuteur, mais comme un compagnon qui assiste les personnes malades au quotidien sur les activités de la vie quotidienne. Grâce à notre modèle, ce compagnon pourrait inférer les compétences résiduelles de ces personnes et le contenu de leur mémoire lors de la réalisation d'une tâche quotidienne. Ainsi, une aide adaptée pourrait alors leur être fournies. Ce compagnon adaptatif pourrait permettre le maintien à domicile des personnes malades et de favoriser leur autonomie fonctionnelle, ce qui pourrait retarder leur entrée en institution spécialisée.

Annexes

A

Procédure (ExpeI)

N°	Objet	Action	Instruction sonore
1	flacon neoplastine	Ouvrir	Ouvrir le flacon de neoplastine.
2	flacon solvant	Ouvrir	Ouvrir le flacon de solvant.
3	flacon neoplastine	Prendre	Prendre le flacon de neoplastine.
4	flacon solvant	Prendre	Prendre le flacon de solvant.
5	flacon solvant	Verser	Verser le solvant dans le neoplastine.
6	flacon solvant	Poser	Poser le flacon de solvant sur la paillasse
7	flacon neoplastine	Visser	Visser le flacon de neoplastine.
8	flacon neoplastine	Agiter	Agiter le flacon de neoplastine.
9	flacon neoplastine	Poser	Poser le flacon de neoplastine sur la paillasse.
10	flacon neoplastine	Ouvrir	Ouvrir le flacon de neoplastine.
11	flacon desorbU	Ouvrir	Ouvrir le flacon de desorbU.
12	maxi-reducer	Prendre	Prendre le maxi-reducer.
13	maxi-reducer	Insérer	Insérer le maxi-reducer dans le desorbU.
14	bouton tiroir	Appuyer	Appuyer sur le bouton du tiroir.
15	flacon neoplastine	Prendre	Prendre le flacon de neoplastine.
16	flacon neoplastine	Scanner	Scanner le flacon de neoplastine.
17	flacon neoplastine	Poser	Poser le flacon de neoplastine dans le tiroir.
18	flacon desorbU	Prendre	Prendre le flacon de desorbU.
19	flacon desorbU	Scanner	Scanner le flacon de desorbU.
20	flacon desorbU	Poser	Poser le flacon de desorbU dans le tiroir.
21	bouton tiroir	Appuyer	Appuyer sur le bouton du tiroir.
22	tube	Prendre	Prendre le tube à essai.
23	tube	Poser	Poser le tube à essai dans le rack.
24	rack	Prendre	Prendre le rack.
25	rack	Poser	Poser le rack dans le panier.
26	panier	Prendre	Prendre le panier.
27	panier	Poser	Poser le panier sur l'automate.
28	bouton démarrage	Appuyer	Appuyer sur le bouton de lancements des tests.
29	panier	Prendre	Prendre le panier.
30	panier	Poser	Poser le panier sur la paillasse.

31	rack	Prendre	Prendre le rack.
32	rack	Poser	Poser le rack sur la paillasse.
33	bouton tiroir	Appuyer	Appuyer sur le bouton du tiroir.
34	flacon neoplastine	Prendre	Prendre le flacon de neoplastine.
35	flacon neoplastine	Poser	Poser le flacon de neoplastine sur la paillasse.
36	flacon desorbU	Prendre	Prendre le flacon de desorbU.
37	flacon desorbU	Poser	Poser le flacon de desorbU sur la paillasse.
38	bouton tiroir	Appuyer	Appuyer sur le bouton du tiroir.

B

Formulaire de consentement (ExpeI)

Formulaire de consentement libre et éclairé
Expérience de psychologie cognitive ergonomique
Université de Bretagne Occidentale, Faculté des Lettres et Sciences Humaines
20 rue Duquesne, 29200 Brest
Centre Européen de Réalité Virtuelle
25 rue Claude Chappe, 29280 Plouzané

Je, soussigné ..... déclare accepter librement, et de façon éclairée, de participer à l'étude intitulée : Comparaison de dispositifs d'immersion visuelle.

Réalisée sous la direction de : Frank Ganier, Ronan Querrec et Joanna Taoum.

Investigateurs principaux : Anaïs Raison et Joanna Taoum.

Objectif de l'étude : comparer différents dispositifs de réalité virtuelle dans le domaine de l'apprentissage humain.

Risques éventuels liés à l'expérience : Maux de tête, étourdissement en enlevant le casque de réalité virtuelle ou en sortant du CAVE.

J'accepte volontairement de participer à cette étude. Je comprends que ma participation n'est pas obligatoire, et que je peux stopper ma participation à tout moment sans encourir aucune responsabilité.

Mon consentement ne décharge pas les organisateurs de l'étude de leurs responsabilités et je conserve tous mes droits garantis par la loi.

Je comprends que les informations recueillies au cours de cette étude sont strictement

confidentielles et à usage exclusif des investigateurs concernés.

Les données recueillies lors de l'étude sont et resteront anonymes. En aucun cas le nom d'un participant ou tout élément susceptible de l'identifier n'apparaîtra dans un rapport ou une publication.

J'accepte que les données enregistrées à l'occasion de cette étude puissent être conservées dans une base de données et faire l'objet d'un traitement informatisé non nominatif. J'ai bien noté que le droit d'accès prévu par la loi « informatique et libertés » s'exerce à tout moment.

Les investigateurs sont à ma disposition pour toutes informations complémentaires (Tél. : 02 98 05 89 60).

Ce formulaire a été établi en deux exemplaires, dont une copie sera conservée par moi-même et l'autre par le(s) chercheur(s) investigateur(s).

Fait à …… le …… en 2 exemplaires.
Signatures (précédée de la mention « Lu et approuvé ») :

Le participant

L'investigateur principal

C

Questionnaire (ExpeI)

Questionnaire sur les caractéristiques des participants

Dans le cadre de ce travail, il nous faudra prendre en compte un ensemble de facteurs individuels présentés ci-dessous :

Condition (à remplir par l'investigateur) : ……
N° participant (à remplir par l'investigateur) : ……
Date : ……
Nom, Prénom : ……
Sexe : ……
Âge : ……
Lieu d'étude : ……
Niveau d'étude : ……
Cursus d'étude : ……

Connaissance du matériel :
Déjà Vu OUI □ NON □
Déjà Utilisé OUI □ NON □
Si OUI, à quelle occasion ? ……

Connaissance de la procédure de lancement de tests de coagulation sanguine :
Déjà Vu OUI □ NON □
Déjà Utilisée OUI □ NON □
Si OUI, à quelle occasion ? ……

Connaissance de l'environnement virtuel pour l'apprentissage humain :
Déjà Vu OUI □ NON □
Déjà Utilisé OUI □ NON □

Si OUI, à quelle occasion ? ……

Pratique des jeux vidéo :
Joueur OUI □ NON □
Quelle(s) console(s) utilisez-vous ? ……

Vous utilisez une console :
□ Très fréquemment (plusieurs fois par semaine)
□ Assez fréquemment (par exemple, une fois par semaine)
□ Rarement (par exemple, une fois par mois)
□ Très rarement (par exemple, une fois par an)
□ Jamais

A quel(s) type(s) de jeux jouez-vous principalement parmi les types cités dans le tableau suivant ? (Classez les types choisis par ordre de fréquence d'utilisation, exemple : 1, le jeu ayant la fréquence la plus élevée, etc.)

□ Action/Aventure	
□ Jeux de rôle (MMORPG* ou MMO*, RPG* etc.)	
□ FPS*	
□ Sport / Course	
□ Jeux de simulation	
□ *Serious game**	
□ Plate-forme*	
□ Jeux de stratégie (wargames...)	
□ Jeux de réflexion (puzzles, objets cachés, labyrinthes etc.)	
□ Jeux de société	
□ Autres Lesquels ? ……	

Jouez-vous à des jeux où il est nécessaire de s'impliquer physiquement (exemple : *Guitar Hero* ou *Just Dance*, etc.) ? OUI □ NON □

* MMORPG ou MMO : *Massively Multiplayer Online Role Playing Game* (jeu de rôle en ligne, massivement multi joueur).

* RPG : *Role Playing Game* (jeu de rôle).

* FPS : *First-Person Shooter* (jeu de tir où l'on voit à travers les « yeux » du personnage).

* *Serious game* : est une application qui combine des aspects « sérieux » (enseignement, apprentissage, information …) avec des aspects ludiques issus du jeu vidéo.

* Plate-forme : jeu vidéo dans lequel le joueur doit faire franchir à son personnage différent niveaux, en sautant de plate-forme en plate-forme.

D

Consignes (ExpeI)

Phase de démonstration et familiarisation

Voici un environnement virtuel représentant une pièce où se trouvent deux cubes de couleurs différentes disposés sur une paillasse. Le but est d'interagir avec ces cubes. Pour cela, vous disposez d'un :

— écran plat / CAVE (+ lunettes) / Oculus Rift
— casque audio (dans le cas de l'Oculus Rift, le casque audio est intégré)
— une manette Wiimote

Vos actions sont traduites dans l'environnement virtuel par une petite sphère jaune qui devient verte au contact d'objet interactif.

Touches utiles pour interagir :

— bouton B (gâchette arrière) = permet de sélectionner un objet et de sélectionner l'action choisie dans le menu
— flèches (haut et bas) = pour se déplacer dans le menu d'actions
— bouton A = pour écouter une instruction

Vous avez la possibilité de réécouter les instructions aussi souvent que vous le souhaitez tant que vous n'avez pas finalisé l'action en cours.

Condition expérimentale

Voici un environnement virtuel qui représente un laboratoire d'analyses médicales. Je vais vous demander de réaliser une procédure de lancement de tests de coagulation sanguine. Pour réaliser cette procédure, vous disposez d'un :

— écran plat / CAVE (+ lunettes) / Oculus Rift
— casque audio (dans le cas de l'Oculus Rift, le casque audio est intégré)
— une manette Wiimote

Comme précédemment, vos actions sont traduites dans l'environnement virtuel par une petite sphère jaune qui devient verte au contact d'objet interactif.

Rappel touches utiles pour interagir :

— bouton B (gâchette arrière) = permet de sélectionner un objet et de sélectionner l'action choisie dans le menu
— flèches (haut et bas) = pour se déplacer dans le menu d'actions
— bouton A = pour écouter une instruction

Vous avez la possibilité de réécouter les instructions aussi souvent que vous le souhaitez tant que vous n'avez pas finalisé l'action en cours.

Vous devrez réaliser cette procédure plusieurs fois.

Avez-vous des questions ?

À partir de maintenant, je ne peux plus intervenir.

E

Procédure (ExpeII)

N°	Objet	Action	Instruction sonore
But 1 : Pour préparer le neoplastine :			
1	flacon neoplastine	Ouvrir	Ouvrir le flacon de neoplastine.
2	flacon solvant	Ouvrir	Ouvrir le flacon de solvant.
3	flacon neoplastine	Prendre	Prendre le flacon de neoplastine.
4	flacon solvant	Prendre	Prendre le flacon de solvant.
5	flacon solvant	Verser	Verser le solvant dans le neoplastine.
6	flacon solvant	Poser	Poser le flacon de solvant sur la paillasse
7	flacon neoplastine	Visser	Visser le flacon de neoplastine.
8	flacon neoplastine	Agiter	Agiter le flacon de neoplastine.
9	flacon neoplastine	Poser	Poser le flacon de neoplastine sur la paillasse
10	flacon neoplastine	Ouvrir	Ouvrir le flacon de neoplastine.
But 2 : Pour préparer le desorbU :			
11	flacon desorbU	Ouvrir	Ouvrir le flacon de desorbU.
12	aimant	Insérer	Insérer l'aimant dans le desorbU
13	bouteille eau distillé	Ouvrir	Ouvrir la bouteille d'eau d'instillé.
14	pipette	Prélever	Prélever de l'eau distillée avec la pipette.
15	pipette	Verser	Verser l'eau distillée dans le desorbU avec la pipette.
16	maxi-reducer	Insérer	Insérer le maxi-reducer dans le desorbU.
But 3 : Pour charger les produits :			
17	bouton tiroir	Appuyer	Appuyer sur le bouton du tiroir.
18	flacon neoplastine	Scanner	Scanner le flacon de neoplastine.
19	flacon neoplastine	Poser	Poser le flacon de neoplastine dans le tiroir.
20	flacon desorbU	Scanner	Scanner le flacon de desorbU.
21	flacon desorbU	Poser	Poser le flacon de desorbU dans le tiroir.
22	bouton tiroir	Appuyer	Appuyer sur le bouton du tiroir.
23	tube	Poser	Poser le tube dans le rack.
24	rack	Poser	Poser le rack dans le panier.
25	panier	Poser	Poser le panier sur l'automate.
But 4 : Pour réaliser les tests :			
26	capot	Fermer	Fermer le capot.

N°	Objet	Action	Instruction sonore
But 1 : Pour préparer le neoplastine :			
27	bouton TP1	Sélectionner	Sélectionner le bouton TP1.
28	bouton TP3	Sélectionner	Sélectionner le bouton TP3.
29	bouton démarrage	Appuyer	Appuyer sur le bouton de démarrage.
30	capot	Ouvrir	Ouvrir le capot.
But 5 : Pour décharger les produits :			
31	panier	Poser	Poser le panier sur la paillasse.
32	rack	Poser	Poser le rack sur la paillasse.
33	bouton tiroir	Appuyer	Appuyer sur le bouton du tiroir.
34	flacon neoplastine	Poser	Poser le flacon de neoplastine sur la paillasse.
35	flacon desorbU	Poser	Poser le flacon de desorbU sur la paillasse.
36	bouton tiroir	Appuyer	Appuyer sur le bouton du tiroir.

F

Formulaire de consentement (ExpeII)

Formulaire de consentement libre et éclairé
Expérience de psychologie cognitive ergonomique
Université de Bretagne Occidentale, Faculté des Lettres et Sciences Humaines
20 rue Duquesne, 29200 Brest
Centre Européen de Réalité Virtuelle
25 rue Claude Chappe, 29280 Plouzané

Je, soussigné ..... déclare accepter librement, et de façon éclairée, de participer à l'étude intitulée : Comparaison de dispositifs d'immersion visuelle.

Réalisée sous la direction de : Frank Ganier, Ronan Querrec et Joanna Taoum.

Investigateurs principaux : Anaïs Raison et Joanna Taoum.

Objectif de l'étude : examiner l'apprentissage en environnement virtuel.

Risques éventuels liés à l'expérience : Maux de tête, étourdissement en enlevant le casque de réalité virtuelle ou en sortant du CAVE.

J'accepte volontairement de participer à cette étude. Je comprends que ma participation n'est pas obligatoire, et que je peux stopper ma participation à tout moment sans encourir aucune responsabilité.

Mon consentement ne décharge pas les organisateurs de l'étude de leurs responsabilités et je conserve tous mes droits garantis par la loi.

Je comprends que les informations recueillies au cours de cette étude sont strictement confidentielles et à usage exclusif des investigateurs concernés.

Les données recueillies lors de l'étude sont et resteront anonymes. En aucun cas le nom d'un participant ou tout élément susceptible de l'identifier n'apparaîtra dans un rapport ou une publication.

J'accepte que les données enregistrées à l'occasion de cette étude puissent être conservées dans une base de données et faire l'objet d'un traitement informatisé non nominatif. J'ai bien noté que le droit d'accès prévu par la loi « informatique et libertés » s'exerce à tout moment.

Les investigateurs sont à ma disposition pour toutes informations complémentaires (Tél. : 06 71 99 43 70).

Ce formulaire a été établi en deux exemplaires, dont une copie sera conservée par moi-même et l'autre par le(s) chercheur(s) investigateur(s).

Fait à …… le …… en 2 exemplaires.
Signatures (précédée de la mention « Lu et approuvé ») :

Le participant

L'investigateur principal

G

Questionnaire (ExpeII)

Questionnaire sur les caractéristiques des participants

Dans le cadre de ce travail, il nous faudra prendre en compte un ensemble de facteurs individuels présentés ci-dessous :

Condition (à remplir par l'investigateur) : ……
N° participant (à remplir par l'investigateur) : ……
Date : ……
Heure début (à remplir par l'investigateur) : ……
Heure de fin (à remplir par l'investigateur) : ……
Nom, Prénom : ……
Mail : ……
Sexe : ……
Âge : ……
Latéralisation : ……
Port de lunettes OUI □ NON □
Lieu d'étude : ……
Niveau d'étude : ……
Cursus d'étude : ……

Connaissance du matériel (Casque et Manette de l'Oculus Rift) :
Déjà Vu OUI □ NON □
Déjà Utilisé OUI □ NON □
Si OUI, à quelle occasion ? ……

Connaissance de la procédure de lancement de tests de coagulation sanguine :
Déjà Vu OUI □ NON □
Déjà Utilisée OUI □ NON □

Si OUI, à quelle occasion ? .....

Connaissance de l'environnement virtuel pour l'apprentissage humain (laboratoire d'analyse médicale) :

Déjà Vu OUI □ NON □

Déjà Utilisé OUI □ NON □

Si OUI, à quelle occasion ? .....

Pratique des jeux vidéo :

Joueur OUI □ NON □

Quelle(s) console(s) utilisez-vous ? .....

Vous utilisez une console :

□ Très fréquemment (plusieurs fois par semaine)

□ Assez fréquemment (par exemple, une fois par semaine)

□ Rarement (par exemple, une fois par mois)

□ Très rarement (par exemple, une fois par an)

□ Jamais

A quel(s) type(s) de jeux jouez-vous principalement parmi les types cités dans le tableau suivant ? (Classez les types choisis par ordre de fréquence d'utilisation, exemple : 1, le jeu ayant la fréquence la plus élevée, etc.)

□ Action/Aventure	
□ Jeux de rôle (MMORPG* ou MMO*, RPG* etc.)	
□ FPS*	
□ Sport / Course	
□ Jeux de simulation	
□ *Serious game**	
□ Plate-forme*	
□ Jeux de stratégie (wargames...)	
□ Jeux de réflexion (puzzles, objets cachés, labyrinthes etc.)	
□ Jeux de société	
□ Autres Lesquels ?	

Jouez-vous à des jeux où il est nécessaire de s'impliquer physiquement (exemple : *Guitar Hero* ou *Just Dance*, etc.) ? OUI □ NON □

* MMORPG ou MMO : *Massively Multiplayer Online Role Playing Game* (jeu de rôle en ligne, massivement multi joueur).

* RPG : *Role Playing Game* (jeu de rôle).

* FPS : *First-Person Shooter* (jeu de tir où l'on voit à travers les « yeux » du personnage).

* *Serious game* : est une application qui combine des aspects « sérieux » (enseignement, apprentissage, information …) avec des aspects ludiques issus du jeu vidéo.

* Plate-forme : jeu vidéo dans lequel le joueur doit faire franchir à son personnage différent niveaux, en sautant de plate-forme en plate-forme.

H

Consignes de la condition non-adaptative (ExpeII)

Ne pas oublier de placer le participant correctement dans la pièce. Régler le casque Oculus Rift à la bonne taille pour éviter une gêne chez le participant, s'assurer qu'il ne voit pas double, floue (régler la molette en dessous du casque). Penser à faire passer le câble de l'Oculus derrière le participant.

Phase de démonstration et familiarisation

Vous allez être dans un environnement virtuel représentant une pièce où se trouvent deux cubes de couleurs différentes disposés sur une paillasse (table) et un agent virtuel qui est là pour vous aider si besoin. Le but est d'interagir avec ces cubes. Pour cela, vous disposez :

1. D'un Casque de réalité virtuelle Oculus Rift avec un casque audio intégré
2. Une Manette Oculus Rift (attention droitier ou gaucher)

Vos actions sont traduites dans l'environnement virtuel par un laser.

1. Pour interagir avec les objets, vous utilisez les touches suivantes :
   (a) Pointage avec le laser de la manette sur les objets
   (b) Gâchette Avant pour sélectionner l'objet à manipuler
      i. Suite à la sélection, un menu d'action s'affiche
         A. Le menu a un titre (pensez à bien vérifier si c'est le bon objet sélectionné)
         B. Possibilité de « Fermer Menu » si pas le bon objet sélectionné
   (c) *Joystick* pour naviguer dans le menu avec possibilité d'orienter le *joystick* vers le haut et le bas (Pour atteindre la dernière action, ne pas forcément descendre toute la liste des actions)
   (d) Gâchette Avant de nouveau pour sélectionner l'action à réaliser

   Comme je vous l'ai dit, un agent sera présent dans l'environnement virtuel pour vous aider si besoin.

2. Pour demander de l'aide à l'agent :
   (a) Pointage avec le laser de la manette en direction de l'agent (niveau buste – tête)
   (b) Gâchette Avant pour sélectionner l'agent
      i. Suite à la sélection, un menu d'Aide s'affiche
      ii. Possibilité de « Fermer Menu » si pas besoin d'aide
   (c) *Joystick* pour naviguer dans le menu d'Aide avec possibilité d'orienter le *joystick* vers le haut et le bas comme dans le menu d'action
   (d) Gâchette Avant de nouveau pour sélectionner l'aide Vous avez la possibilité de redemander de l'aide à l'agent aussi souvent que vous le souhaitez tant que vous n'avez pas finalisé l'action en cours. L'agent vous informera si vous faites une erreur et il vous indiquera lorsque la procédure est finie.
3. Avez-vous des questions ? *(faire plusieurs fois la phase de familiarisation et s'assurer que les boutons sont appris)*

ⓘ Je vous dirais quand vous pourrez commencer, ne pas appuyer sur une touche avant.

Condition expérimentale : Essai 1

Vous allez être dans un environnement virtuel qui représente un laboratoire d'analyses médicales. Comme précédemment, il y a un agent virtuel qui est là pour vous aider si besoin. Je vais vous demander de réaliser une procédure de lancement de tests de coagulation sanguine. Pour réaliser cette procédure, vous disposez :

1. D'un Casque de réalité virtuelle Oculus Rift avec un casque audio intégré
2. Une Manette Oculus Rift (attention droitier ou gaucher)

Vos actions sont traduites dans l'environnement virtuel par un laser.

1. Pour interagir avec les objets, vous utiliserez les touches qu'on a vu dans la phase de familiarisation (rappel de touches) :
   (a) Pointage avec le laser de la manette sur les objets
   (b) Gâchette Avant pour sélectionner l'objet à manipuler
      i. Suite à la sélection, un menu d'action s'affiche
         A. Le menu a un titre (pensez à bien vérifier si c'est le bon objet sélectionné)
         B. Possibilité de « Fermer Menu » si pas le bon objet sélectionné
   (c) *Joystick* pour naviguer dans le menu avec possibilité d'orienter le *joystick* vers le haut et le bas (Pour atteindre la dernière action, ne pas forcément descendre toute la liste des actions)
   (d) Gâchette Avant de nouveau pour sélectionner l'action à réaliser

   Comme je vous l'ai dit, un agent sera présent dans l'environnement virtuel pour vous aider si besoin.
2. Pour demander de l'aide à l'agent :

(a) Pointage avec le laser de la manette en direction de l'agent (niveau buste – tête)
(b) Gâchette Avant pour sélectionner l'agent
    i. Suite à la sélection, un menu d'Aide s'affiche
    ii. Possibilité de « Fermer Menu » si pas besoin d'aide
(c) *Joystick* pour naviguer dans le menu d'Aide avec possibilité d'orienter le *joystick* vers le haut et le bas comme dans le menu d'action
(d) Gâchette Avant de nouveau pour sélectionner l'aide

Lorsque je vous informe que vous pouvez commencer, l'agent virtuel vous annoncera systématiquement (automatiquement) l'instruction que vous devriez réaliser, contrairement à ce qu'on à vue dans la phase de familiarisation.
Vous avez la possibilité de redemander de l'aide à l'agent aussi souvent que vous le souhaitez tant que vous n'avez pas finalisé l'action en cours.
L'agent vous informera si vous faites une erreur et il vous indiquera lorsque la procédure est finie.
Vous devrez réaliser cette procédure plusieurs fois, en la répétant.
Je vous informerai quand on aura fini et on enlèvera le casque entre chaque essai.

3. Avez-vous des questions ?

Pour le bon déroulement de l'expérimentation, à partir de maintenant, je ne peux plus intervenir sur tout ce qui concerne la procédure.

ⓘ Je vous dirais quand vous pourrez commencer, ne pas appuyer sur une touche avant.

Condition expérimentale : ESSAIS N+1 (à dire entre essai 1 et essai 2)

Comme précédemment, vous allez être dans un environnement virtuel qui représente un laboratoire d'analyses médicales. Il y a toujours un agent virtuel qui est là pour vous aider si besoin. Je vais vous demander de réaliser une procédure de lancement de tests de coagulation sanguine. Pour réaliser cette procédure, vous disposez :

1. D'un Casque de réalité virtuelle Oculus Rift avec un casque audio intégré
2. Une Manette Oculus Rift (attention droitier ou gaucher)

Vos actions sont traduites dans l'environnement virtuel par un laser.

1. Pour interagir avec les objets, vous utiliserez les touches qu'on a vu dans la phase de familiarisation (rappel de touches) :
   (a) Pointage avec le laser de la manette sur les objets
   (b) Gâchette Avant pour sélectionner l'objet à manipuler
       i. Suite à la sélection, un menu d'action s'affiche
          A. Le menu a un titre (pensez à bien vérifier si c'est le bon objet sélectionné)
          B. Possibilité de « Fermer Menu » si pas le bon objet sélectionné
   (c) *Joystick* pour naviguer dans le menu avec possibilité d'orienter le *joystick* vers le haut et le bas (Pour atteindre la dernière action, ne pas forcément descendre

toute la liste des actions)

(d) Gâchette Avant de nouveau pour sélectionner l'action à réaliser

Comme je vous l'ai dit, un agent sera présent dans l'environnement virtuel pour vous aider si besoin.

2. Pour demander de l'aide à l'agent :
   (a) Pointage avec le laser de la manette en direction de l'agent (niveau buste – tête)
   (b) Gâchette Avant pour sélectionner l'agent
      i. Suite à la sélection, un menu d'Aide s'affiche
      ii. Possibilité de « Fermer Menu » si pas besoin d'aide
   (c) *Joystick* pour naviguer dans le menu d'Aide avec possibilité d'orienter le *joystick* vers le haut et le bas comme dans le menu d'action
   (d) Gâchette Avant de nouveau pour sélectionner l'aide

   Lorsque je vous informe que vous pouvez commencer, l'instruction ne sera pas annoncée systématiquement, il faudra demander de l'aide à l'agent virtuel quand vous en avez besoin (comme dans la phase de familiarisation).

   Vous avez la possibilité de redemander l'aide aussi souvent que vous le souhaitez tant que vous n'avez pas finalisé l'action en cours.

   L'agent vous informera si vous faites une erreur et il vous indiquera lorsque la procédure est finie.

   Vous devrez réaliser cette procédure plusieurs fois, en la répétant. Je vous informerai quand on aura fini et on enlèvera le casque entre chaque essai.

   Avez-vous des questions ?

Pour le bon déroulement de l'expérimentation, à partir de maintenant, je ne peux plus intervenir sur tout ce qui concerne la procédure.

ⓘ Je vous dirais quand vous pourrez commencer, ne pas appuyer sur une touche avant.

Fin de passation : dire au participant qu'il ne doit pas raconter ce qu'il a fait pour éviter de biaiser les résultats ET insister sur précautions à prendre (éviter de conduire tout de suite après utilisation du casque).

I

Consignes de la condition adaptative (ExpeII)

Ne pas oublier de placer le participant correctement dans la pièce. Régler le casque Oculus Rift à la bonne taille pour éviter une gêne chez le participant, s'assurer qu'il ne voit pas double, floue (régler la molette en dessous du casque). Penser à faire passer le câble de l'Oculus derrière le participant.

Phase de démonstration et familiarisation

Vous allez être dans un environnement virtuel représentant une pièce où se trouvent deux cubes de couleurs différentes disposés sur une paillasse (table) et un agent virtuel qui est là pour vous aider si besoin.

Le but est d'interagir avec ces cubes. Pour cela, vous disposez :

1. D'un Casque de réalité virtuelle Oculus Rift avec un casque audio intégré
2. Une Manette Oculus Rift (attention droitier ou gaucher)

Vos actions sont traduites dans l'environnement virtuel par un laser.

1. Pour interagir avec les objets, vous utilisez les touches suivantes :
   (a) Pointage avec le laser de la manette sur les objets
   (b) Gâchette Avant pour sélectionner l'objet à manipuler
      i. Suite à la sélection, un menu d'action s'affiche
         A. Le menu a un titre (pensez à bien vérifier si c'est le bon objet sélectionné)
         B. Possibilité de « Fermer Menu » si pas le bon objet sélectionné
   (c) *Joystick* pour naviguer dans le menu avec possibilité d'orienter le *joystick* vers le haut et le bas (Pour atteindre la dernière action, ne pas forcément descendre toute la liste des actions)
   (d) Gâchette Avant de nouveau pour sélectionner l'action à réaliser

   Comme je vous l'ai dit, un agent sera présent dans l'environnement virtuel pour vous aider si besoin.

2. Pour demander de l'aide à l'agent :
   (a) Pointage avec le laser de la manette en direction de l'agent (niveau buste – tête)
   (b) Gâchette Avant pour sélectionner l'agent
      i. Suite à la sélection, un menu d'Aide s'affiche
         — Aide Action correspond à l'annonce de l'action à réaliser (*donner exemple*)
         — Aide But correspond à l'annonce du but de la sous-partie de la procédure que vous êtes en train de réaliser (*donner exemple*)
         — Aide Objet correspond à la mise en saillance (rouge clignotant) de l'objet à manipuler (*donner exemple*)
         — Aide correspond à une aide globale, c'est-à-dire, annonce du but et de l'action et à la mise en saillance de l'objet (*donner exemple*)
      ii. Possibilité de « Fermer Menu » si pas besoin d'aide
   (c) *Joystick* pour naviguer dans le menu d'Aide avec possibilité d'orienter le *joystick* vers le haut et le bas comme dans le menu d'action
   (d) Gâchette Avant de nouveau pour sélectionner l'aide

   Vous avez la possibilité de redemander de l'aide à l'agent aussi souvent que vous le souhaitez tant que vous n'avez pas finalisé l'action en cours.
   L'agent vous informera si vous faites une erreur et il vous indiquera lorsque la procédure est finie.
3. Avez-vous des questions ? *(faire plusieurs fois la phase de familiarisation et s'assurer que les boutons sont appris)*

ⓘ Je vous dirais quand vous pourrez commencer, ne pas appuyer sur une touche avant.

Condition expérimentale : Essai 1

Vous allez être dans un environnement virtuel qui représente un laboratoire d'analyses médicales. Comme précédemment, il y a un agent virtuel qui est là pour vous aider si besoin. Je vais vous demander de réaliser une procédure de lancement de tests de coagulation sanguine. Pour réaliser cette procédure, vous disposez :

1. D'un Casque de réalité virtuelle Oculus Rift avec un casque audio intégré
2. Une Manette Oculus Rift (attention droitier ou gaucher)

Vos actions sont traduites dans l'environnement virtuel par un laser.

1. Pour interagir avec les objets, vous utiliserez les touches qu'on a vu dans la phase de familiarisation (rappel de touches) :
   (a) Pointage avec le laser de la manette sur les objets
   (b) Gâchette Avant pour sélectionner l'objet à manipuler
      i. Suite à la sélection, un menu d'action s'affiche

A. Le menu a un titre (pensez à bien vérifier si c'est le bon objet sélectionné)
B. Possibilité de « Fermer Menu » si pas le bon objet sélectionné

(c) *Joystick* pour naviguer dans le menu avec possibilité d'orienter le *joystick* vers le haut et le bas (Pour atteindre la dernière action, ne pas forcément descendre toute la liste des actions)
(d) Gâchette Avant de nouveau pour sélectionner l'action à réaliser

Comme je vous l'ai dit, un agent sera présent dans l'environnement virtuel pour vous aider si besoin.

2. Pour demander de l'aide à l'agent :
   (a) Pointage avec le laser de la manette en direction de l'agent (niveau buste – tête)
   (b) Gâchette Avant pour sélectionner l'agent
      i. Suite à la sélection, un menu d'Aide s'affiche
         — Aide Action correspond à l'annonce de l'action à réaliser (*donner exemple*)
         — Aide But correspond à l'annonce du but de la sous-partie de la procédure que vous êtes en train de réaliser (*donner exemple*)
         — Aide Objet correspond à la mise en saillance (rouge clignotant) de l'objet à manipuler (*donner exemple*)
         — Aide correspond à une aide globale, c'est-à-dire, annonce du but et de l'action et à la mise en saillance de l'objet (*donner exemple*)
      ii. Possibilité de « Fermer Menu » si pas besoin d'aide
   (c) *Joystick* pour naviguer dans le menu d'Aide avec possibilité d'orienter le *joystick* vers le haut et le bas comme dans le menu d'action
   (d) Gâchette Avant de nouveau pour sélectionner l'aide

   Lorsque je vous informe que vous pouvez commencer, l'agent virtuel vous annoncera systématiquement (automatiquement) l'instruction que vous devrez réaliser, contrairement à ce qu'on à vue dans la phase de familiarisation.
   Vous avez la possibilité de redemander de l'aide à l'agent aussi souvent que vous le souhaitez tant que vous n'avez pas finalisé l'action en cours.
   L'agent vous informera si vous faites une erreur et moi, je vous dirai quand la procédure sera finie.
   Vous devrez réaliser cette procédure plusieurs fois, en la répétant. Je vous informerai quand on aura fini et on enlèvera le casque entre chaque essai.

3. Avez-vous des questions ?
   Pour le bon déroulement de l'expérimentation, à partir de maintenant, je ne peux plus intervenir sur tout ce qui concerne la procédure.

Pour le bon déroulement de l'expérimentation, à partir de maintenant, je ne peux plus intervenir sur tout ce qui concerne la procédure.

ⓘ Je vous dirais quand vous pourrez commencer, ne pas appuyer sur une touche avant.

Condition expérimentale : ESSAIS N+1 (à dire entre essai 1 et essai 2)

Comme précédemment, vous allez être dans un environnement virtuel qui représente un laboratoire d'analyses médicales. Il y a toujours un agent virtuel qui est là pour vous aider si besoin. Je vais vous demander de réaliser de nouveau, la même procédure de lancement de tests de coagulation sanguine. Pour réaliser cette procédure, vous disposez :

1. D'un Casque de réalité virtuelle Oculus Rift avec un casque audio intégré
2. Une Manette Oculus Rift (attention droitier ou gaucher)

Vos actions sont traduites dans l'environnement virtuel par un laser.

1. Pour interagir avec les objets, vous utiliserez les touches qu'on a vu dans la phase de familiarisation (rappel de touches) :
   (a) Pointage avec le laser de la manette sur les objets
   (b) Gâchette Avant pour sélectionner l'objet à manipuler
      i. Suite à la sélection, un menu d'action s'affiche
         A. Le menu a un titre (pensez à bien vérifier si c'est le bon objet sélectionné)
         B. Possibilité de « Fermer Menu » si pas le bon objet sélectionné
   (c) *Joystick* pour naviguer dans le menu avec possibilité d'orienter le *joystick* vers le haut et le bas (Pour atteindre la dernière action, ne pas forcément descendre toute la liste des actions)
   (d) Gâchette Avant de nouveau pour sélectionner l'action à réaliser

   Comme je vous l'ai dit, un agent sera présent dans l'environnement virtuel pour vous aider si besoin.
2. Pour demander de l'aide à l'agent :
   (a) Pointage avec le laser de la manette en direction de l'agent (niveau buste – tête)
   (b) Gâchette Avant pour sélectionner l'agent
      i. Suite à la sélection, un menu d'Aide s'affiche
         — Aide Action correspond à l'annonce de l'action à réaliser (*donner exemple*)
         — Aide But correspond à l'annonce du but de la sous-partie de la procédure que vous êtes en train de réaliser (*donner exemple*)
         — Aide Objet correspond à la mise en saillance (rouge clignotant) de l'objet à manipuler (*donner exemple*)
         — Aide correspond à une aide globale, c'est-à-dire, annonce du but et de l'action et à la mise en saillance de l'objet (*donner exemple*)
      ii. Possibilité de « Fermer Menu » si pas besoin d'aide

(c) *Joystick* pour naviguer dans le menu d'Aide avec possibilité d'orienter le *joystick* vers le haut et le bas comme dans le menu d'action

(d) Gâchette Avant de nouveau pour sélectionner l'aide

Lorsque je vous informe que vous pouvez commencer, si vous avez besoin d'aide et qu'elle n'est pas fournie par l'agent virtuel, il faudra lui demander.
Vous avez la possibilité de redemander de l'aide aussi souvent que vous le souhaitez tant que vous n'avez pas finalisé l'action en cours.
L'agent vous informera si vous faites une erreur et moi, je vous dirai quand la procédure sera finie.
Vous devrez réaliser cette procédure plusieurs fois, en la répétant.
Je vous informerai quand on aura fini et on enlèvera le casque entre chaque essai.

3. Avez-vous des questions ?
   Pour le bon déroulement de l'expérimentation, à partir de maintenant, je ne peux plus intervenir sur tout ce qui concerne la procédure.

Ⓘ Je vous dirais quand vous pourrez commencer, ne pas appuyer sur une touche avant.

Fin de passation : dire au participant qu'il ne doit pas raconter ce qu'il a fait pour éviter de biaiser les résultats ET insister sur précautions à prendre (éviter de conduire tout de suite après utilisation du casque).

Liste des tableaux

Table des figures

Références bibliographiques

Aleven, V., B. M. McLaren, J. Sewall et K. R. Koedinger (2006). “The Cognitive Tutor Authoring Tools (CTAT) : Preliminary Evaluation of Efficiency Gains”. In : *Intelligent Tutoring Systems*. Sous la dir. de M. Ikeda, K. D. Ashley et T.-W. Chan. Berlin, Heidelberg : Springer Berlin Heidelberg, p. 61–70. isbn : 978-3-540-35160-3 (cf. p. 35).

Anderson, J. R. (1983). “A spreading activation theory of memory”. In : *Journal of verbal learning and verbal behavior* 22.3, p. 261–295 (cf. p. 4, 81, 109, 122).

Anderson, J. R. (1996). “ACT : A simple theory of complex cognition.” In : *American Psychologist* 51.4, p. 355 (cf. p. 61).

Anderson, J. R. (2014). *Rules of the mind*. Psychology Press (cf. p. 122).

Anderson, J. R., D. Bothell, M. D. Byrne, S. Douglass, C. Lebiere et Y. Qin (2004). “An integrated theory of the mind.” In : *Psychological review* 111.4, p. 1036 (cf. p. 35, 60).

Atkinson, R. C. et R. M. Shiffrin (1968). “Human memory : A proposed system and its control processes.” In : *In K. W. Spence and J. T. Spence (Eds.), The Psychology of learning and motivation : Advances in research and theory (vol. 2).* P. 89–105 (cf. p. 5, 46, 60, 62, 72, 122).

Azevedo, R., F. Bouchet, J. M. Harley, R. Feyzi-Behnagh, G. Trevors, M. Duffy, M. Taub, N. Pacampara, L. Agnew, S. Griscom et al. (2011). “MetaTutor : An Intelligent Multi-Agent Tutoring System Designed to Detect, Track, Model Foster Self-Regulated Learning”. In : *Proceedings of the Fourth Workshop on Self-Regulated Learning in Educational Technologies* (cf. p. 30).

Bach, C. (2004). “Development and validation of Ergonomic Criteria for Human Virtual-Environments Interactions”. Thése. Université de Metz (cf. p. 99, 109).

Bailenson, J., N. Yee, J. Blascovich, A. Beall, N. Lundblad et M. Jin (2008). “The Use of Immersive Virtual Reality in the Learning Sciences : Digital Transformations of Teachers, Students, and Social Context”. In : *Journal of the Learning Sciences 17(1)* 17.1, p. 102–141 (cf. p. 1).

Baylor, A., J. Ryu et E. Shen (2003). “The effects of pedagogical agent voice and animation on learning, motivation and perceived persona”. In : *EdMedia : World Conference on Educational Media and Technology*. Association for the Advancement of Computing in Education (AACE), p. 452–458 (cf. p. 37).

Benabbou, A. (2018). “Génération dynamique de situations critiques en environnements virtuels : dilemme et ambiguité”. Thése. Université de Technologie, Compiègne (cf. p. 21).

Benabbou, A., D. Lourdeaux et D. Lenne (2017). “Génération dynamique de dilemmes en environnement virtuel à partir de modèles de connaissances”. In : *8ème Conférence sur les Environnements Informatiques pour l’Apprentissage Humain (EIAH)* (cf. p. 12).

Blandin, B. et R. Querrec (2014). “Quelle méthode pour concevoir un environnement virtuel pour apprendre une activité ? Une tentative de réponse : le projet EAST”. In : *3e Colloque International de Didactique Professionnelle* (cf. p. 3).

Bloom, B. S., M. D. Englehart, E. J. Furst, W. H. Hill et D. R. Krathwohl (1956). “Taxonomy of educational objectives. Vol. 1 : Cognitive domain”. In : *New York : McKay*, p. 20–24 (cf. p. 3).

Bourdeau, J. et M. Grandbastien (2010). “Modeling tutoring knowledge”. In : *Advances in intelligent tutoring systems*. Springer, p. 123–143 (cf. p. 27, 30).

Bouville Berthelot, R., V. Gouranton, T. Boggini, F. Nouviale et B. Arnaldi (2015). “#FIVE : High-Level Components for Developing Collaborative and Interactive Virtual Environments”. In : *Software Engineering and Architectures for Realtime Interactive Systems (SEARIS), 2015 IEEE 8th Workshop on*. IEEE, p. 33–40 (cf. p. 15, 16).

Burleson, W. (2006). “Affective Learning Companions : strategies for empathetic agents with real-time multimodal affective sensing to foster meta-cognitive and meta-affective approaches to learning, motivation, and perseverance”. Thèse de doct. Massachusetts Institute of Technologies (cf. p. 38).

Busso, C., Z. Deng, S. Yildirim, M. Bulut, C. M. Lee, A. Kazemzadeh, S. Lee, U. Neumann et S. Narayanan (2004). “Analysis of Emotion Recognition Using Facial Expressions, Speech and Multimodal Information”. In : *Proceedings of the 6th International Conference on Multimodal Interfaces*. ICMI ’04. State College, PA, USA : ACM, p. 205–211 (cf. p. 53).

Cassell, J., T. Bickmore, M. Billinghurst, L. Campbell, K. Chang, H. Vilhjálmsson et H. Yan (1999). “Embodiment in conversational interfaces : Rea”. In : *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*. ACM, p. 520–527 (cf. p. 39).

Cassell, J., H. H. Vilhjálmsson et T. Bickmore (2004). “Beat : the behavior expression animation toolkit”. In : *Life-Like Characters*. Springer, p. 163–185 (cf. p. 39).

Chevaillier, P., T. H. Trinh, M. Barange, F. Devillers, J. Soler, P. De Loor et R. Querrec (2012). “Semantic modelling of virtual environments using Mascaret”. In :

Proceedings of the Fourth Workshop on Software Engineering and Architectures for Realtime Interactive Systems, IEEE VR, Singapore, p. 1–8 (cf. p. 5, 21).

Claude, G. (2016). “Séquencement d’actions en environnement virtuel collaboratif”. Thèse. INSA de Rennes (cf. p. 18).

Claude, G., V. Gouranton, R. B. Berthelot et B. Arnaldi (2014). “Short Paper : #SEVEN, a Sensor Effector Based Scenarios Model for Driving Collaborative Virtual Environment”. In : *ICAT-EGVE, International Conference on Artificial Reality and Telexistence, Eurographics Symposium on Virtual Environments*. Bremen, Germany, p. 1–4 (cf. p. 5).

Claude, G., V. Gouranton, R. Bouville Berthelot et B. Arnaldi (2014). “#SEVEN, a Sensor Effector Based Scenarios Model for Driving Collaborative Virtual Environment”. In : *ICAT-EGVE, International Conference on Artificial Reality and Telexistence, Eurographics Symposium on Virtual Environments*. Bremen, Germany, p. 1–4 (cf. p. 16, 17).

Conati, C., A. Gertner et K. Vanlehn (2002). “Using Bayesian Networks to Manage Uncertainty in Student Modeling”. In : *User Modeling and User-Adapted Interaction* 12, p. 371–417. issn : 0924-1868. doi : 10.1023/A:1021258506583 (cf. p. 32).

Conway, A. R., M. J ; Kane, M. F. Bunting, D. Z. Hambrick, O. Wilhelm et R. W. Engle (2005). “Working memory span tasks : A methodological review and user’s guide”. In : *Psychonomic bulletin & review* 12.5, p. 769–786 (cf. p. 68).

Cormier, J., D. Pasco, C. Syllebranque et R. Querrec (2011). “VirTeaSy a haptic simulator for dental education”. In : *The 6th International Conference on Virtual Learning*. T. 156, p. 61–68 (cf. p. 12).

Craik, F. I. M. et R. S. Lockhart (1972). “Levels of processing : A framework for memory research”. In : *Journal of verbal learning and verbal behavior* 11.6, p. 671–684 (cf. p. 62).

de Sevin, E., R. Niewiadomski, E. Bevacqua, A-.M. Pez, M. Mancini et C. Pelachaud (2010). “Greta, une plateforme d’agent conversationnel expressif et interactif”. In : *Technique et science informatiques* 29.7, p. 751 (cf. p. 40–42).

DeVault, D., R. Artstein, G. Benn, T. Dey, E. Fast, A. Gainer, K. Georgila, J. Gratch, A. Hartholt, M. Lhommet et al. (2014). “SimSensei Kiosk : A virtual human interviewer for healthcare decision support”. In : *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*. International Foundation for Autonomous Agents et Multiagent Systems, p. 1061–1068 (cf. p. 44).

DeVault, D., J. Mell et J. Gratch (2015). “Toward natural turn-taking in a virtual human negotiation agent”. In : *AAAI Spring Symposium on Turn-taking and Coordination in Human-Machine Interaction. AAAI Press, Stanford, CA*. Palo Alto, California, p. 2–9 (cf. p. 44).

Dibie-Barthélemy, J., O. Haemmerlé et E. Salvat (2006). “A semantic validation of conceptual graphs”. In : *Knowledge-Based Systems* 19.7, p. 498–510 (cf. p. 28).

Ding, Y., K. Prepin, J. Huang, C. Pelachaud et T. Artières (2014). “Laughter animation synthesis”. In : *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*. International Foundation for Autonomous Agents et Multiagent Systems, p. 773–780 (cf. p. 42).

Ekman, P. et W. V. Friesen (1976). “Measuring facial movement”. In : *Environmental psychology and nonverbal behavior* 1.1, p. 56–75 (cf. p. 53, 54).

Fitts, P. M. et M. I. Posner (1967). “Human performance.” In : (cf. p. 95).

Flavell, J. H. (1976). “Metacognitive aspects of problem solving”. In : *The nature of intelligence*, p. 231–235 (cf. p. 30).

Foloppe, D. A., P. Richard, T. Yamaguchi, F. Etcharry-Bouyx et P. Allain (2018). “The potential of virtual reality-based training to enhance the functional autonomy of Alzheimer’s disease patients in cooking activities : A single case study”. In : *Neuropsychological rehabilitation* 28.5, p. 709–733 (cf. p. 131).

Fricoteaux, L., I. Thouvenin et D. Mestre (2014). “GULLIVER : a decision-making system based on user observation for an adaptive training in informed virtual environments”. In : *Engineering Applications of Artificial Intelligence* 33, p. 47–57 (cf. p. 12).

Gagné, E. D. (1985). *The cognitive psychology of school learning*. Little, Brown (cf. p. 62).

Ganier, F. (2004). “Factors Affecting the Processing of Procedural Instructions : Implications for Document Design”. In : *IEEE Transactions on Professional Communication* 47.1, p. 15–26. issn : 03611434 (cf. p. 80).

Ganier, F., C. Hoareau et F. Devillers (2013). “Évaluation des performances et de la charge de travail induits par l’apprentissage de procédures de maintenance en environnement virtuel”. In : *Le travail humain* 76.4, p. 335–363 (cf. p. 94, 109).

Glanzer, M. et A. R. Cunitz (1966). “Two storage mechanisms in free recall”. In : *Journal of verbal learning and verbal behavior* 5.4, p. 351–360 (cf. p. 68).

Graesser, A. C. et S. D’Mello (2012). “Emotions during the learning of difficult material”. In : 57, p. 183–225 (cf. p. 37).

Graesser, A. C., Xiangen Hu et al. (2018). “ElectronixTutor : an intelligent tutoring system with multiple learning resources for electronics”. In : *International Journal of STEM Education* 5.1, p. 15. issn : 2196-7822. doi : 10.1186/s40594-018-0110-y (cf. p. 37).

Graesser, A. C., H. Li et C. Forsyth (2014). “Learning by Communicating in Natural Language with Conversational Agents”. In : *Current Directions in Psychological Science* 23.51, p. 374–380. doi : 10.1177/0963721414540680 (cf. p. 37).

Grafsgaard, J. F., J. B. Wiggins, K. E. Boyer, E. N. Wiebe et J. C. Lester (2013). “Automatically recognizing facial indicators of frustration : a learning-centric analysis”. In : *Proceedings of the Humaine Association Conference on Affective Computing and Intelligent Interaction*. IEEE, p. 159–165 (cf. p. 36).

Hartholt, A., D. Traum, S. C. Marsella, A. Shapiro, G. Stratou, A. Leuski, L.-P. Morency et J. Gratch (2013). “All Together Now : Introducing the Virtual Human Toolkit”. In : *13th International Conference on Intelligent Virtual Agents*. Edinburgh, UK (cf. p. 42, 43).

Hoareau, C. (2016). “Élaboration et évaluation de recommandations ergonomiques pour le guidage de l’apprenant en EVAH”. Thése. Université de Bretagne occidentale, Brest (cf. p. 92, 93, 100).

Hotte, R., H. Godinet et J.-P. Pernin (2007). “Scénariser l’apprentissage, une activité de modélisation”. In : *International Journal of Technologies in Higher Education* 4, p. 2 (cf. p. 3).

Jewett, A. E., L. S. Jones, S. M. Luneke et S. M. Robinson (1971). “Educational change through a taxonomy for writing physical education objectives”. In : *Quest* 15.1, p. 32–38 (cf. p. 3).

Johnson, W., J. W. Rickel et J. C. Lester (2000). “Animated pedagogical agents : Face-to-face interaction in interactive learning environments”. In : *International Journal of Artificial intelligence in education* 11.1, p. 47–78 (cf. p. 37).

Johnson, W., P. Rizzo, W. Bosma, S. Kole, M. Ghijsen et H. van Welbergen (2004). “Generating Socially Appropriate Tutorial Dialog”. In : *ISCA Workshop on Affective Dialogue Systems*. Berlin, Heidelberg, p. 254–264 (cf. p. 51).

Jones, H. et N. Sabouret (2013). “TARDIS - A simulation platform with an affective virtual recruiter for job interviews”. In : *IDGEI : Intelligent Digital Games for Empowerment and Inclusion* (cf. p. 38).

Kautzmann, T. R., T. Carlotto et P. A. Jaques (2016). “Adaptive Training of the Metacognitive Skill of Knowledge Monitoring in Intelligent Tutoring Systems”. In : *Proceedings of the 13th International Conference on Intelligent Tutoring Systems - Volume 9684*. Zagreb, Croatia : Springer-Verlag New York, Inc., p. 301–306 (cf. p. 31).

Kautzmann, T. R. et P. A. Jaques (2017). “Improving the Metacognitive Ability of Knowledge Monitoring in Computer Learning Systems”. In : *Researcher Links Workshop : Higher Education for All*. Springer, p. 124–140 (cf. p. 32).

Kay, J., S. Kleitman et R. Azevedo (2013). “Empowering teachers to design learning resources with metacognitive interface elements”. In : *R Luckin, S Puntambekar, P oodyear, B Grabowski, J Underwood, N Winters (Eds.) : Handbook of Design in Educational Technology. New York : Routledge*, p. 124–134 (cf. p. 31).

Kokane, A., H. Singhal, S. Mukherjee et G.R.M. Reddy (2014). “Effective E-learning using 3D Virtual Tutors and webRTC Based Multimedia Chat”. In : *International Conference on Recent Trends in Information Technology (ICRTIT)*, p. 1–6 (cf. p. 37, 51).

Koper, R. et R. Van Es (2004). “Modelling units of learning from a pedagogical perspective”. In : *Online Education Using Learning Objects* 40, p. 40–52 (cf. p. 3, 25, 49).

Kopp, S., B. Jung, N. Lessmann et I. Wachsmuth (2003). “Max-a multimodal assistant in virtual reality construction.” In : *KI* 17.4, p. 11 (cf. p. 39, 40).

Kopp, S., B. Krenn, S. C. Marsella, A. N. Marshall, C. Pelachaud, H. Pirker, K. R. Thórisson et H. H. Vilhjálmsson (2006). “Towards a common framework for multimodal generation : The behavior markup language”. In : *Proceedings of 6th International Conference on Intelligent Virtual Agents*. T. 4133. LNCS. Springer, p. 205–217 (cf. p. 40).

Krathwohl, D. R., B. S. Bloom et B. B. Masia (1964). “Taxonomy of educational objectives, handbook II : affective domain. New York : David McKay Company”. In : 1 (cf. p. 3).

Le Corre, F. (2013). “CHRYSAOR : un système tutoriel intelligent pour les environnements virtuels d’apprentissage humain. Application à la formation au matériel de laboratoire en hémostase”. Thése. Université de Bretagne occidentale, Brest (cf. p. 92).

Lee, J. et S. Marsella (2006). “Nonverbal behavior generator for embodied conversational agents”. In : *International Workshop on Intelligent Virtual Agents*. Springer, p. 243–255 (cf. p. 43).

Leon, E., G. Clarke, V. Callaghan et F. Sepulveda (2007). “A User-independent Real-time Emotion Recognition System for Software Agents in Domestic Environments”. In : *Eng. Appl. Artif. Intell.* 20.3, p. 337–345 (cf. p. 53).

Lester, J. C., S. A. Converse, S. E. Kahler, T. Barlow, B. A. Stone et R. S. Bhogal (1997). “The persona effect : affective impact of animated pedagogical agents”. In : *Proceedings of the ACM SIGCHI Conference on Human factors in computing systems*. ACM, p. 359–366 (cf. p. 37, 51).

Lim, K. C. (2018). “Using the xAPI to Track Learning”. In : *Innovations in Open and Flexible Education*. Singapore : Springer Singapore, p. 233–242. isbn : 978-981-10-7995-5. doi : 10.1007/978-981-10-7995-5_21 (cf. p. 34).

Loup-Escande, E., L. Dominjon, D. Perret, S. Erhel, E. Jamet, N. Michinov, C. Andriot, P. Gravez et M. Ragot (2013). “La démarche de Conception Centrée-Utilisateur en Réalité Virtuelle : l’exemple du projet VirtualiTeach”. In : *8èmes journées de l’Association Française de Réalité Virtuelle, Augmentée, Mixte et d’Intéraction 3D* (cf. p. 11).

Lourdeaux, D., A. Benabbou, L. Huguet et R. Lacaze-Labadie (2017). “HUMANS : suite logicielle pour la scénarisation d’environnements virtuels pour la formation à des situations socio-techniques complexes”. In : *3e Conférence Nationale sur les Applications Pratiques de l’Intelligence Artificielle (APIA 2017)*. Caen, France, p. 61–68 (cf. p. 5, 18–22).

Luengo, V., A. Larcher et J. Tonetti (2011). “Design and Implementation of a Visual and Haptic Simulator in a Platform for a TEL System in Percutaneuos Orthopedic Surgery”. In : Studies in Health Technology and Informatics 163. Sous la dir. de James D. Westwood & al., p. 324–328. doi : 10.3233/978-1-60750-706-2-324 (cf. p. 12).

Mayer, R. E. (2009). *Multimedia Learning*. 2nd. New York, NY, USA : Cambridge University Press (cf. p. 99, 112).

Mayer, R. E. et C. S. DaPra (2012). “An embodiment effect in computer-based learning with animated pedagogical agents.” In : *Journal of Experimental Psychology : Applied* 18.3, p. 239 (cf. p. 37).

Mayer, R. E. et R. Moreno (2002). “Aids to computer-based multimedia learning”. In : *Learning and instruction* 12.1, p. 107–119 (cf. p. 71).

McKhann, G. M., D. S. Knopman, H. Chertkow, B. T. Hyman, C. R. J. J., C. H. Kawas, W. E. Klunk, W. J. Koroshetz, J. J. Manly, R. Mayeux et al. (2011). “The

diagnosis of dementia due to Alzheimer's disease : Recommendations from the National Institute on Aging-Alzheimer's Association workgroups on diagnostic guidelines for Alzheimer's disease". In : *Alzheimer's & dementia* 7.3, p. 263–269 (cf. p. 131).

Miller, G. A. (1956). *The Magical Number Seven, Plus or Minus Two : Some Limits on Our Capacity for Processing Information* (cf. p. 61, 67).

Mojarad, S., A. Essa, S. Mojarad et R. S. Baker (2018). "Data-Driven Learner Profiling Based on Clustering Student Behaviors : Learning Consistency, Pace and Effort". In : *14th International Conference on Intelligent Tutoring Systems*. Cham : Springer International Publishing, p. 130–139. isbn : 978-3-319-91464-0 (cf. p. 33).

Moreno, R., R. E. Mayer, H. A. Spires et J. C. Lester (2001). "The case for social agency in computer-based teaching : Do students learn more deeply when they interact with animated pedagogical agents ?" In : *Cognition and instruction* 19.2, p. 177–213 (cf. p. 37).

Morineau, T. (2000). "Context effect on problem solving during a first immersion in a virtual environment". In : *Cahiers de Psychologie cognitive* 19.5-6, p. 533–555 (cf. p. 94).

Nakhal, B. (2017). "Generation of communicative intentions for virtual agents in an intelligent virtual environment : application to virtual learning environment". Thése. Université de Bretagne occidentale - Brest (cf. p. 25, 56).

Niewiadomski, R., E. Bevacqua, M. Mancini et C. Pelachaud (2009). "Greta : an interactive expressive ECA system". In : *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems-Volume 2*, p. 1399–1400 (cf. p. 55).

Nkambou, R. (2010). "Modeling the domain : An introduction to the expert module". In : *Advances in intelligent tutoring systems*. Springer, p. 15–32 (cf. p. 27, 28).

Nkambou, R., C. Frasson, G. Gauthier et K. Rouane (2001). "An authoring model and tools for knowledge engineering in intelligent tutoring systems". In : *Journal of Interactive Learning Research* 12.4, p. 323 (cf. p. 29).

Nkambou, R., R. Mizoguchi et J. Bourdeau (2010). "Advances in Intelligent Tutoring Systems". In : 1st. T. 308. Springer Science & Business Media (cf. p. 5, 27, 58).

Noël, B. (1997). *La métacognition*. Paris, Bruxelles : De Boeck Univesité. (cf. p. 30).

Ochs, M., G. De Montcheuil, J.-M. Pergandi, J. Saubesty, C. Pelachaud, D. Mestre et P. Blache (2017). "An architecture of virtual patient simulation platform to train doctor to break bad news". In : *Conference on Computer Animation and Social Agents (CASA)* (cf. p. 42).

Paquette, G. (2007). "Graphical Ontology Modeling Language for Learning Environments." In : *Technology, Instruction, Cognition & Learning* 5.2 (cf. p. 29).

Patil, J. V. et P. Bailke (2016). "Real time facial expression recognition using RealSense camera and ANN". In : *International Conference on Inventive Computation Technologies (ICICT)*. T. 2. IEEE, p. 1–6 (cf. p. 53).

Querrec, R., J. Taoum, B. Nakhal et B. Bevacqua (2018). "Model for Verbal Interaction between an Embodied Tutor and a Learner in Virtual Environments". In : *Intelligent*

Virtual Agents : 18th International Conference, IVA 2018, Sydney, NSW, Australia, November 5-8, 2018 (cf. p. 52, 82).

Rao, A. S. et M. P. Georgeff (1991). “Modeling rational agents within a BDI-architecture.” In : *KR* 91, p. 473–484 (cf. p. 39).

Razzaq, L. M. et N. T. Heffernan (2009). “To Tutor or Not to Tutor : That is the Question.” In : *AIED*, p. 457–464 (cf. p. 30).

Richir, S., P. Fuchs, D. Lourdeaux, D. Millet, C. Buche et R. Querrec (2015). “How to design compelling Virtual Reality or Augmented Reality experience ?” In : *International Journal of Virtual Reality (IJVR)* 15, p. 35–47 (cf. p. 51).

Rodriguez, F. G. et D. L. Scapin (1997). “Editing MAD* task descriptions for specifying user interfaces, at both semantic and presentation levels”. In : *Design, Specification and Verification of Interactive Systems ’97*. Vienna : Springer Vienna, p. 193–208. isbn : 978-3-7091-6878-3 (cf. p. 20).

Rowe, J. P., S. W. McQuiggan, B. W. Mott et J. C. Lester (2007). “Motivation in narrative-centered learning environments”. In : *Proceedings of the workshop on narrative learning environments, AIED*, p. 40–49 (cf. p. 37, 51).

Roy, M. (2014). “Sentiment de présence et réalité virtuelle pour les langues–Une étude de l’émergence de la présence et de son influence sur la compréhension de l’oral en allemand langue étrangère”. In : *Alsic. Apprentissage des Langues et Systèmes d’Information et de Communication* 17 (cf. p. 11).

Saunier, J., M. Barange, B. Blandin, R. Querrec et J. Taoum (2016). “Designing Adaptable Virtual Reality Learning Environments”. In : p. 5 (cf. p. 22).

Sowndararajan, A., R. Wang et D. A. Bowman (2008). “Quantifying the benefits of immersion for procedural training”. In : *Proceedings of the 2008 workshop on Immersive projection technologies/Emerging display technologiges*. ACM, p. 2 (cf. p. 93).

Taoum, J., R. Querrec, J. Saunier et B. Blandin (2015). “EAST : Environnements d’Apprentissage Scientifiques et Techniques”. In : *Les journées de l’Association Française de Réalité Virtuelle, Augmentée, Mixte et d’Intéraction 3D*. Bordeaux, France (cf. p. 22).

Thiebaux, M., S. C. Marsella, A. N. Marshall et M. Kallmann (2008). “Smartbody : Behavior realization for embodied conversational agents”. In : *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems-Volume 1*, p. 151–158 (cf. p. 43).

Thórisson, K. R., T. List, C. Pennock et J. DiPirro (2005). “Whiteboards : Scheduling blackboards for semantic routing of messages & streams”. In : *AAAI-05 Workshop on Modular Construction of Human-Like Intelligence*, p. 8–15 (cf. p. 41).

Tobias, S. et H. Everson (1996). “Assessing Metacognitive Knowledge Monitoring. Report No. 96-01.” In : *College Entrance Examination Board* (cf. p. 31).

Vail, A. K., J. F. Grafsgaard, K. E. Boyer, E. N. Wiebe et J. C. Lester (2016). “Predicting Learning from Student Affective Response to Tutor Questions”. In : *Proceedings*

of the 13th International Conference on Intelligent Tutoring Systems. Cham : Springer International Publishing, p. 154–164 (cf. p. 36, 37).

Van der Linden, M. et A.-C. Van der Linden (2015). “Penser autrement le vieillissement et la maladie d’Alzheimer”. In : *Revista E-Psi* 5.1, p. 4–22 (cf. p. 131).

Vanlehn, K., C. Lynch, K. Schulze, J. A. Shapiro, R. Shelby, L. Taylor, D. Treacy, A. Weinstein et M. Wintersgill (2005). “The Andes Physics Tutoring System : Lessons Learned”. In : *Int. J. Artif. Intell. Ed.* 15.3, p. 147–204. issn : 1560-4292 (cf. p. 32).

Vilhjálmsson, H. H., N. Cantelmo, J. Cassell, N. Ech Chafai, M. Kipp, S. Kopp, M. Mancini, S. C. Marsella, A. N. Marshall, C. Pelachaud, Z. Ruttkay, K. R. Thórisson, H. van Welbergen et R. J. van der Werf (2007). “The Behavior Markup Language : Recent Developments and Challenges”. In : *Proceedings of 7th International Conference on Intelligent Virtual Agents*. T. 4722. LNCS. Paris, France : Springer, p. 99–111 (cf. p. 25, 40, 55).

Vygotsky, L. S. (1978). *Mind in society*. Cambridge, MA : Harvard University Press (cf. p. 19).

Wickens, C. D. (2002). “Situation Awareness and Workload in Aviation”. In : *Current Directions in Psychological Science* 11.4, p. 128–133 (cf. p. 99, 109, 112).

Woolf, B. P. (1992). “Building knowledge based tutors”. In : *Computer Assisted Learning*. Sous la dir. d’Ivan Tomek. Berlin, Heidelberg : Springer Berlin Heidelberg, p. 46–60 (cf. p. 26, 48, 82).

Woolf, B. P. (2010). “Student Modeling”. In : *Advances in Intelligent Tutoring Systems*. Berlin, Heidelberg : Springer Berlin Heidelberg, p. 267–279. isbn : 978-3-642-14363-2. doi : 10.1007/978-3-642-14363-2_13 (cf. p. 34).

Yngve, V. H. (1970). “On getting a word in edgewise”. In : p. 567–578 (cf. p. 41).

Publications

Le travail de recherche que nous avons mené dans cette thèse nous a permis de publier plusieurs articles dans des conférences nationales et internationales, citées ci-dessous :

- **au niveau international :**
  - **conférences :**
    - Querrec R., **Taoum J.**, Nakhal B. & Bevacqua E. : Model for Verbal Interaction between an Embodied Tutor and a Learner in Virtual Environments, 18th ACM International Conference on Intelligent Virtual Agents, (IVA'18), Sydney, Australia, November 2018.
    - **Taoum J.**, Bevacqua E. & Querrec R. : Adaptive Virtual Tutor based on the Inference of the Student's Memory Content, 14th International Conference on Intelligent Tutoring Systems, (ITS'18) Montreal, Canada, June 2018.
    - **Taoum J.**, Nakhal B., Bevacqua E. & Querrec R. : A Design Proposition for Interactive Virtual Tutors in an Informed Environment, 16th International Conference on Intelligent Virtual Agents, (IVA'16), pp. 341–350, Los Angeles, California, September 2016.
  - **workshop :**
    - **Taoum J.**, Raison, A., Bevacqua E. & Querrec R. : An Adaptive Tutor to Promote Learners' Skills Acquisition during Procedural Learning, Workshop eliciting Adaptive Sequences for Learning (WeASeL), in ITS'18 Montreal, Canada, June 2018.

- **au niveau national :**
  - **conférences :**
    - **Taoum J.**, Nakhal B., Bevacqua E. & Querrec R. : Une proposition de conception pour les tuteurs virtuels interactifs dans un environnement informé, Les

Journées de l'Association Française de Réalité Virtuelle, Augmentée, Mixte et d'Interaction 3D, (AFRV'16) Brest, France, Octobre 2016.

► **workshop :**

— **Taoum J.**, Bevacqua E. & Querrec R. : Feedback Cognitif pour les Agents Conversationnels Animés, Workshop Affect, Compagnon Artificial, Interaction, (WACAI'16) Brest, France, June 2016.

► **autres publications :**

— Saunier J., Barange M., Blandin B., Querrec R. & **Taoum J.** : Designing Adaptable Virtual Reality Learning Environments, Proceedings of the 2016 Virtual Reality International Conference, (VRIC'16) Laval, France, March 2016.

— **Taoum J.**, Querrec R., Saunier J. & Blandin B. : EAST - Environnement d'Apprentissage Scientifiques et Technique, Les Journées de l'Association Française de Réalité Virtuelle, Augmentée, Mixte et d'Interaction 3D, (AFRV'15) Bordeaux, France, Octobre 2015.

Titre : MEMORIA, un modèle de représentation de la mémoire de l'apprenant pour les systèmes tutoriels intelligents et adaptatifs

Mots clés : Environnement Virtuel pour l'Apprentissage Humain, Adaptation, Tuteur Intelligent, Agent Pédagogique Incarné.

Résumé : Dans cette thèse, nous présentons MEMORIA, un modèle de représentation de la mémoire de l'apprenant pour les systèmes tutoriels intelligents et adaptatifs. La contribution principale de ce modèle est une formalisation et une implémentation du modèle de l'apprenant sous forme de mémoires qui stockent les informations perçues par l'apprenant dans un environnement virtuel et les instructions émises par le tuteur. La conception de notre modèle est basée sur les quatre composantes classiques d'un système tutoriel intelligent. Le modèle du domaine est représenté par les connaissances métiers formalisées à l'aide de MASCARET. Afin de rendre naturelles les interactions entre le tuteur et l'apprenant, nous représentons le modèle de l'interface par l'intermédiaire d'un agent conversationnel animé à l'aide de la plate-forme Greta. Le modèle de l'apprenant est constitué de l'ensemble des connaissances acquises par l'apprenant en cours de simulation. Ces connaissances sont organisées dans trois mémoires : la mémoire sensorielle, la mémoire de travail et la mémoire à long terme. Notre enjeu majeur porte sur la formalisation de l'encodage des informations dans ces mémoires, ainsi que le flux de données entre celles-ci. Cette formalisation est basée sur la théorie de la mémoire humaine proposée par Atkinson et Shiffrin et inspirée de l'architecture cognitive ACT-R. Le modèle de tuteur que nous proposons est centré sur la réalisation d'un comportement qui adapte l'exécution du scénario en fonction des connaissances de l'apprenant et de ses interactions avec le tuteur. Une étude expérimentale a été menée pour valider notre modèle. Nous avons comparé deux groupes de participants. Dans le premier groupe, nous avons intégré un tuteur adaptatif utilisant notre modèle, qui adapte l'exécution du scénario pédagogique et dans le second groupe, un tuteur non adaptatif qui réalise un scénario pédagogique figé. Les résultats de cette étude permettent de conclure quant à l'efficacité de notre modèle pour un apprentissage de procédure.

Title: MEMORIA, a model of the learner's memory representation for adaptive and intelligent tutoring systems

Keywords: Virtual Environment for Training, Adaptation, Intelligent Tutor, Embodied Pedagogical Agent.

Abstract: In this thesis, we present MEMORIA, a model of the learner's memory representation for adaptive and intelligent tutoring systems. The main contribution of this model is a formalization and an implementation of the learner's model using memories that store the information perceived by the learner in a virtual environment and the instructions given by the tutor. The design of our model is based on the four classic components of an intelligent tutorial system. The domain model is represented by the domain knowledge that is formalized using MASCARET. In order to make the interactions between the tutor and the learner natural, we represent the interface model through an embodied conversational agent using GRETA. The learner's model is made of all the knowledge acquired by the learner during the simulation. This knowledge is organized into three memories: sensory memory, working memory, and long-term memory. Our major challenge is to formalize the encoding of information in these memories, as well as the data flow between them. This formalization is based on the theory of human memory proposed by Atkinson and Shiffrin and inspired by the cognitive architecture ACT-R. Our proposed tutor model focuses on the realization of a behavior that adapts the execution of the pedagogical scenario according to the learner's knowledge and the interactions with the tutor. An experimental study was conducted to validate our model. We compared two groups of participants. In the first group, we integrated an adaptive tutor using our model which adapts the execution of the pedagogical scenario and in the second group, a non-adaptive tutor who applied a fixed pedagogical scenario. The results of this study allow us to conclude on the effectiveness of our model for procedural learning.